IntechOpen

Industrial Robotics: Programming, Simulation and Applications

Edited by Low Kin Huat

Industrial Robotics

Programming, Simulation and Applications

Edited by

Kin-Huat Low

Industrial Robotics: Programming, Simulation and Applications
http://dx.doi.org/10.5772/40
Edited by Low Kin Huat

First published in Croatia, 2006 by INTECH d.o.o.
eBook (PDF) Published by IN TECH d.o.o.
Place and year of publication of eBook (PDF): Rijeka, 2019.
IntechOpen is the global imprint of IN TECH d.o.o.
Printed in Croatia

Legal deposit, Croatia: National and University Library in Zagreb

Additional hard and PDF copies can be obtained from orders@intechopen.com

Industrial Robotics: Programming, Simulation and Applications
Edited by Low Kin Huat

p. cm.

ISBN 3-86611-286-6

eBook (PDF) ISBN 978-953-51-5808-0

Contents

Preface

About 86 years ago, in 1921, Karel Capek introduced the word “robot” in his book, Rossum’s Universal Robots. The term was then coined and first used by Isaac Asimov in a short story published in 1942. Subsequently, after World War II, automatic machines were designed to increase productivity, and machine-tool manufacturers made numerically-controlled (NC) machines to enable manufacturers to produce better products. At the end, a marriage between the NC capability of machine tools and the manipulators created a simple robot. The early robot was sold in 1961 by Unimation, the first robot company established by Engelberger. This happened 40 years after the word “robot” was introduced.

Today, after another 40 year, robotic arms are widely used in industrial manufacturing. There is no doubt that robots have increased in capability and performance through improved mechanisms, controller, software development, sensing, drive systems, and materials.

The ARS presents the state-of-the-art developments in this book on “Industrial Robotics: Programming, Simulation and Applications,” which collects 35 chapters of recent robotics topics in different applications. The articles are contributed by research scientists, computer scientists, professors, and engineers from 18 countries in different regions of the world. It contains a good representation of today’s advanced robotics, maintaining a good balance between practical and research scientific focus.

The works presented in this book can be divided into several groups:

Sensor Issues	Chapters 1 and 2
Welding Technology	Chapters 3, 15, 27, and 30
Robotics System Planning, Networking, Scheduling, Programming, and Reworking	Chapters 4 and Chapters 16-19
Computer Vision, Visual Study, and Applications	Chapters 5, 14, 22, and 24
Robotics Calibration	Chapters 6-9
Internet-based Robotics	Chapters 10 and 11
Motion Study and Path Planning	Chapters 12 and 13
Assembly for Various Industrial Applications	Chapters 28, 29, and 35
Cover Issues of Automation Assembly and Robotics Cells	Chapters 20, 21, 25, and 31
Special Topics: Tele-robotics, Flexible Robots, Gonio-reflectometer, Surveillance and Climbing Robots	Chapter 23, 26, and Chapters 32-34

This book is dedicated to robotics researchers, practicing engineers, and R&D managers, who wish to enhance and broaden their knowledge and expertise on the fundamentals, technologies and applications of industrial robots.

Providing an overview of contemporary research with advanced robotic systems, the above articles provide up-to-date advances and solutions to some theoretical and application problems encountered in various areas of industrial robots. By no means, however, we can cover the whole area of industrial robots in this single book. The editor is indebted to all authors for their valuable contributions to this book. Special thanks to Editors in Chief of International Journal of Advanced Robotic Systems for their effort in making this book possible.

Kin-Huat Low (K. H. Low)
School of Mechanical and Aerospace Engineering
Nanyang Technological University
Singapore 639798
mkhlow@ntu.edu.sg

1

A Design Framework for Sensor Integration in Robotic Applications

Dimitrios I. Kosmopoulos
National Centre for Scientific Research "Demokritos"
Institute of Informatics and Telecommunications
153 10 Agia Paraskevi Greece

1. Introduction

The benefits of open robot controllers' architecture have not been ignored by the robot constructors industry and in the recent years the closed proprietary architectures were partially modified to allow access for the integration of external sensors. However, the application integrators are still not able to exploit to the full extend this openness and installations that employ sensors are still the exception in most industrial plants, mainly due to the development difficulties. What is needed to the developers of robotic solutions is a generic method for rapid development and deployment of sensors and robots to enable quick responses to the market needs. This need is mainly dictated by the fact that many such applications sharing common functionality are required in industry. By sharing and reusing design patterns there is a possibility to reduce costs significantly. Therefore, a careful domain analysis that will identify the similarities and differences between the various types of applications and which will provide the infra structure for the development of new ones is of obvious importance.

A major hindrance in the development of sensor-guided robotic applications is the complexity of the domain, which requires knowledge from the disciplines of robot vision, sensor-based control, parallel processing and real time systems. A framework defines how the available scientific knowledge can be used to create industrial systems by embodying the theory of the domain; it clarifies the core principles and methods and the algorithmic alternatives. Therefore new algorithms may be easily integrated through the addition or substitution of modular components.

A design framework should provide the means for the integration of the entailed components and give an overview of their collaboration. The technological advances may then be easily adopted provided that the same cooperation rules will be applied.

In the past many academic and commercial frameworks have been presented, covering parts of the disciplines entailed in sensor-guided robotic applications. However, the available frameworks that can be used for sensor integration in robotic applications were either limited in scope or were not developed targeting to that general goal. From the point of view of the developer of sensor-guided robots the available frameworks treated the general problem in a rather fragmented manner.

This work presents a design framework that defines the basic components that comprise robotic sensor guided systems. It also shows how these components interact to implement the most common industrial requirements and provides some application examples.

The rest of this work is structured as follows: the following section defines the context of this research; section 3 summarizes the contribution of this work; section 4 identifies the commonalities and differences between various industrial applications; section 5 describes the user and the design requirements; section 6 describes the basic component, while section 7 shows how these components interact to implement the basic use cases; section 8 demonstrates the validity of the approach by presenting two applications: a sunroof fitting robot and a depalletizing robot. The chapter ends with section 9, which presents the conclusions and the future work.

2. Related work

The literature regarding integration of robotic systems is quite rich. On the one hand there are many middleware systems, which are quite general, treat sensor data processing and the control issues separately and do not address the issue of their interaction in typical use cases. On the other hand some methodologies handle the issue of the cooperation between sensors and actuators but are limited to certain sensor types, control schemes or application-specific tasks. Thus methods that will bridge this gap are needed.
The middleware systems that have been proposed implement the interfaces to the underlying operating system and hardware and they provide programming interfaces that are more familiar to control application builders than the real-time operating system primitives. Some the most interesting of these frameworks are the OSACA (Sperling & Lutz 1996) and the Orca (Orca 2006). The scope of these frameworks is to provide the infra structure and not to specify what the application will functionally be able to do, what data it will accept and generate, and how the end users and the components will interact. They can be used in combination with our framework, which specifies the functionality and the behavior for sensor-guided applications. Some other popular middleware platforms for mobile robots are the Player/Stage/Gazebo (Vaughan et al. 2003), the CARMEN (Montemerlo et al. 2003), the MIRO (Utz et al. 2002), and the CLARAty (Nesnas et al. 2006).
Many other attempts have been made to implement open control objects through the introduction of class or component libraries. Such libraries providing motion control functionality or the related components are the Orca (Orca 2006), the SMART (Anderson 1993), the START (Mazer et al. 1998). The integration of sensors for robot guidance is addressed by libraries such as the Servomatic (Toyama et al. 1996), XVision (Hager & Toyama 1998) but these libraries are limited to certain sensors, control schemes or application-specific tasks thus they are inappropriate for general use.
For object detection in sensory data a large number of frameworks has been presented by researchers and industry. Examples of such frameworks are the Image Understanding Environment (ECV-Net –IUE 2000) and the vxl (VXL 2006). A large number of libraries are available such as the Open CV (Intel 2006), the Halcon (MVTec 2006), etc. Such libraries or frameworks may be used in combination to the proposed methodology.

3. Contribution

The proposed design framework capitalizes on the increased “openness” of modern robot controllers, by extending the capabilities provided by the manufacturer, through intelligent handling and fusion of acquired information. It provides a modular and systematic way in order to encompass different types of sensory input; obviously, this approach can be fully profitable, when the control system architecture allows for this information exchange to take place. The presented work proposes how to develop software that will undertake the robot control at the end-effector level by profiting of the sensory data. More specifically, it presents:

- A unified approach for implementing sensor guided robot applications considering the restrictions posed by the controller manufacturers and the capabilities of the available control and vision methods.
- A generalization of the available approaches to enable the integration of the popular sensors to any open controller using a wide variety of control schemes.
- Patterns of use including open loop and closed loop control schemes.
- Concrete examples of the development of industrial applications facilitated by the presented framework.

The proposed design is independent of the organizational layer (procedure management, user interface, knowledge base) and execution levels (actuators, sensors) of the application and of the rest layers of the robot controller (integration platform, operating system and hardware).

4. Domain description

A unified approach for the integration of industrial sensors in robotic systems through a common object-oriented framework is reasonable because the necessary abstractions are present for the sensors, for the robotic actuators and for their interactions. This will be clarified through a domain description which will lead to the requirements.

4.1 Sensors

The sensors that are frequently used for industrial robotic tasks may be categorized to contact and non-contact ones. Typical contact sensors that are used in industry are force-torque sensors and tactile sensors. Popular non-contact sensors are cameras, laser sensors, ultrasound sensors which give similar output and inductive, capacitive or Hall sensors for detection of the presence of an object. The processing of the sensory data aims to the identification of features, which in this context normally means the parts of the image with local properties (e.g., points, edges etc.), from which useful information about the environment can be extracted. The use of sensors in most cases presupposes that the inverse sensor model is available, which allows calculation of the environment state from the sensory data.

The sensors can be treated in a unified manner in design due to the fact that the sensory data can be described as vectors or 2-d arrays and what differs is the type of the sensory data (bits, bytes, double precision numbers etc.). Examples of vector images (one-dimensional) are the outputs of the force sensors and the laser sensors; examples of two dimensional images are the outputs of the monochrome cameras, the tactile sensors and the arrays of "binary" sensors; example of image with higher dimensions is the color image, which consists of three monochrome channels (e.g., red, green and blue).

4.2 Robot controllers

The robot controller is the module that determines the robot movement that is the pose, velocity and acceleration of the individual joints or the end-effector. This is performed through motion planning and motion control. Motion planning includes the trajectory definition considering its requirements and restrictions, as well as the manipulator dynamic features. The robot operates in the joint space but the motion is normally programmed in the

Cartesian space due to easier comprehension for humans. Therefore the controller has to solve the inverse kinematic problem (the calculation of the joint states from the end-effector state) to achieve a desired state in the Cartesian space for the end-effector; then the controller must calculate the proper current values to drive the joint motors in order to produce the desired torques in the sampling moments. The industrial robot controllers are designed to provide good solutions to this problem within certain subspaces.
In closed architectures the user is able to program the robot movement by giving the target states and eventually by defining the movement type, e.g., linear, circular etc. The controller then decides how to reach these states by dividing the trajectory into smaller parts defined by interpolation points. The time to perform the movement through a pair of interpolation points is defined by the interpolation frequency; in order to reach the interpolation points closed loop control at the joint level is performed with much higher frequency. Generally the control at the joint level is not open to the programmer of industrial robots.

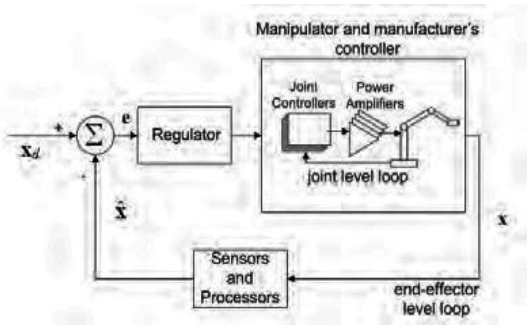


Fig. 1. Closed loop control scheme at the end-effector level. The end–effector's state $\mathbf{x}$ is measured by the sensing devices and the corresponding measurement $\hat{\mathbf{x}}$ is given as feedback; then $\hat{\mathbf{x}}$ is compared to the desired state $\mathbf{x}_d$, the difference is fed to the regulator and the regulation output is sent to the robot controller to move the manipulator. The joint-level loop is also displayed.

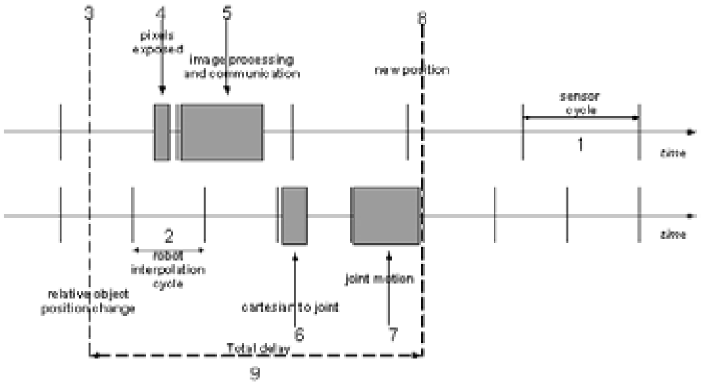


Fig. 2 The cycles of robot and sensor and their synchronization for compensating for a target movement. The total delay from the initial target movement to the new robot pose to compensate for this movement depends mainly on the robot and sensor cycle (adapted from (Coste-Maniere & Turro 1997)).

For tasks requiring low accuracy a "blind" trajectory execution with closed loop control at the joint level is acceptable. However, for the tasks that require high accuracy, closed-loop control at the end-effector level is necessary, during which the next interpolation points have to be calculated in each interpolation cycle using external input; this need is satisfied by the open architecture robot controllers, which unlike the typical closed architecture controllers, permit the user or an external system to define precisely the interpolation points and oblige the end-effector to strictly pass through them. However this restriction may produce oscillations at the end-effector level depending partially on the coarseness of the trajectory. These oscillations may be avoided if an additional regulator is used. The control scheme presented in Fig. 1 is then applied.

4.3 Use of sensors by robots

The sensors can be used for robot guidance at the end-effector level to compensate for the environmental uncertainties. Open and closed loop configurations are then applied. The open loop configurations are the simplest, since the desired robot state is sent to the robot controller and executed by the manipulator without concern about the actual end-effector state. In the closed loop configurations the current end-effector state is compared to the desired one in each interpolation cycle and corrected accordingly in the next cycle.
To identify the commonalities and the differences between the various closed loop control schemes we present the various schemes that may be applied. The schemes identified in (Hutchinson et al. 1996) for visual sensors are generally applicable also to the other sensors types if their type permits it. For the extraction of the desired robot state the "Endpoint Open Loop" scheme assumes perception of only the target's state using non-contact sensors. The "Endpoint Closed Loop" scheme assumes that both the end-effector and target state are perceived and can be applied using contact and non-contact sensors. According to another categorization the closed loop control schemes are divided to the "Position-based", which require extraction of the position for the perceived entities (end-effector and/or target), while the "Image-based" schemes extract the desired robot state by exploiting directly the image features.
In order to implement the aforementioned schemes various methods may be employed according to the application needs. For position-based tasks the physical state of all the observed objects (target, end-effector) is calculated; it is done through optimization methods that use the 3D measurements (or image measurements) in combination to object and sensor models and "fit" the model to the measurements. In image-based systems generally no object or sensor model is used but only an equation that provides the current state from the image measurements, provided that a Jacobian matrix is known (for more details the reader may refer to (Hutchinson et al. 1996)). The Jacobian matrix may be calculated online or (more frequently) offline through a training procedure.
In many applications a single sensor is not adequate for state calculation and in these cases tools such such as Kalman or particle filters may be employed. From the measurement input the system state is calculated, assuming that the system dynamics are linear around the small region where the task is to be executed (for more details the reader may refer to (Valavanis & Saridis 1992)). The filters provide also estimation about the next system state and can also be used for iterative solution of optimization problems in position-based applications.
Another common issue for the closed-loop control systems is that the ideal case in every moment that the robot has to move to reach a desired state this state has to be based on

sensory data that depict the current environmental situation. However, this is usually not possible due to differences in the sensor and robot cycles and due to the time required for processing and transmission of the sensory data (see Fig. 2). Therefore synchronization schemes are necessary that may use prediction for the calculation of the currently required state, based on previous data. For the open loop systems this problem does not exist because the robot moves only after reading and processing the sensory data.

4.4 Application layered architecture

The architecture of a typical robotic application is illustrated in Fig. 3. The left part is adapted from (Valavanis & Saridis 1992), where the levels of organization, processing and execution are presented where intelligence increases from bottom to top. The organization level includes the procedure manager that defines the robotic tasks execution sequence (usually by using a robot programming language), the knowledge base that includes the system's knowledge about the environment and the user interface. The processing level includes the controller, which defines the robot motion by using the processed input from internal and external sensors. The execution level includes the actuators, which perform the requested tasks and the sensors that provide raw sensory data to the system. The right part in Fig. 3 presents the layered architecture of the open robot controllers as presented in OSACA (Sperling & Lutz 1996). The upper layers use the services provided by the lower ones. The hardware layer includes the processing units that are used for the software execution. The operating system controls and synchronizes the use of the resources and acts as the intermediate layer between the hardware and the application software. The communication implements the information exchange between the subsystems. The configuration allows the installation and parameterization of the desired subsystems. The control objects are the subsystems that provide the continuous processing functionality in the application. They are interfaced to the software and hardware system layers through the software- and hardware - API layer, which should provide plug and play capabilities.

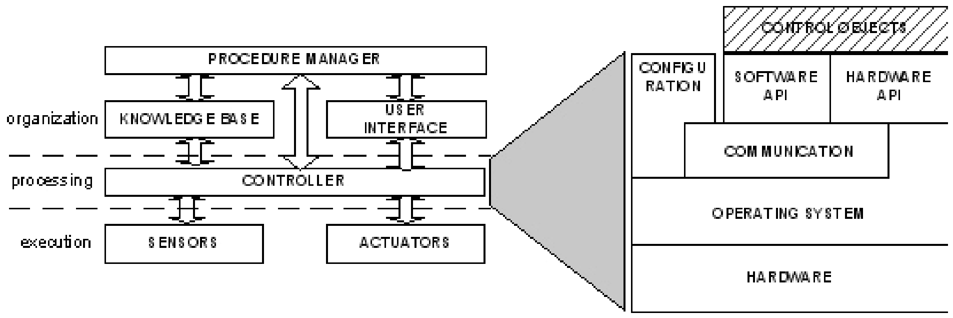


Fig. 3. Typical layered architecture of robotic applications. The organizational level includes the Procedure Manager, the Knowledge Base and the User Interface; the execution level includes the Sensors and the Actuators; the processing level includes the robot controller, which may be analyzed to the open architecture layers.

5. Requirements

5.1 Functional and non functional

A generic framework has to consider some functional and non-functional requirements derived from the description about the sensors, the robot controllers and the control schemes using sensors. These requirements as seen from the user perspective are briefly presented in the following.

As regards the basic functional requirements, the framework should provide the infra structure and patterns of use for modelling and interfacing of the sensors, should allow sensor integration with industrial robotic controllers in open loop control and closed loop control schemes, should use multiple sensors of the same or different type, should allow object recognition from the sensory data and should use virtual sensory data and manipulators.

The basic non-functional requirements are related to the software quality factors. The composed systems must be characterized by reliability, efficiency, integrity, usability, maintainability, flexibility, testability, reusability, portability and interoperability.

5.2 Use cases

The tasks that the user requires to be executed (use cases) are more or less common for the typical industrial applications and have to be covered by the framework. These are:

- *Manual movement* (offline). The user moves the robot manually using a user interface module by specifying the target state.
- *State reading*. (offline-online). Some sections of the robot trajectory can be executed without strict precision requirements. In these cases the system reads the desired state from the organization level. The consecutive states, from which the robot trajectory is composed, may be defined at organizational level through a robotic language (where high-level tasks are defined) or may be pre-recorded during state teaching.
- *State teaching* (offline). Trajectory sections (set-points) are recorded, while the user moves manually the robot.
- *Parameterization* (offline). During parameterization the data needed for sensor-based control are manually defined and stored. These data are essential for the operation of system and they are retrieved each time a sensor-guided task has to be executed. For each task the system must be parameterized separately. The parameterization may regard the end-effector, the sensor, the regulator and the processing.
- *Training* (offline). Some special system parameters are automatically calculated e.g., inverse Jacobian matrices resulting from feature measurements while the robot executes movement at predetermined steps for each degree of freedom.
- *Sensor-guided movement* (online). The system executes a task in real time. The task includes movement with and without sensory feedback at the end-effector level. When the robot moves without sensory feedback it reads from a database the trajectory that has been stored during teaching. When sensory feedback is required the movement is calculated online using the parameters that have been fixed during parameterization; the parameters are loaded from a database.

5.3 Design requirements and fulfillment

There are several requirements in designing a reusable framework for sensor integration in industrial robotic systems and they are dictated by the user requirements presented in the previous section. The most important design requirements are presented next, followed by the decisions for their fulfillment.

The composing elements have to be modular and decoupled to cover the requirements for reusability, reliability, testability, usability, flexibility, integrity and maintainability. The semantic decoupling is achieved by grouping functionalities according to general needs and not according to application specific requirements. The interconnection decoupling is achieved by specifying the control objects layer (capsules, described in next section) in a modular way. Each of them has well-specified responsibilities and attributes and communicates with the other ones only through ports using the related protocols (described in next section). By forcing capsules to communicate solely through ports, it is possible to fully de-couple their internal implementations from any direct knowledge about the environment and to make them highly reusable. Virtual sensors and robots can also be used keeping the same protocols. The separation of objects is achieved through the use of the object - oriented C++ language. Interface objects are used wherever possible for component decoupling based on the different roles of the elements.

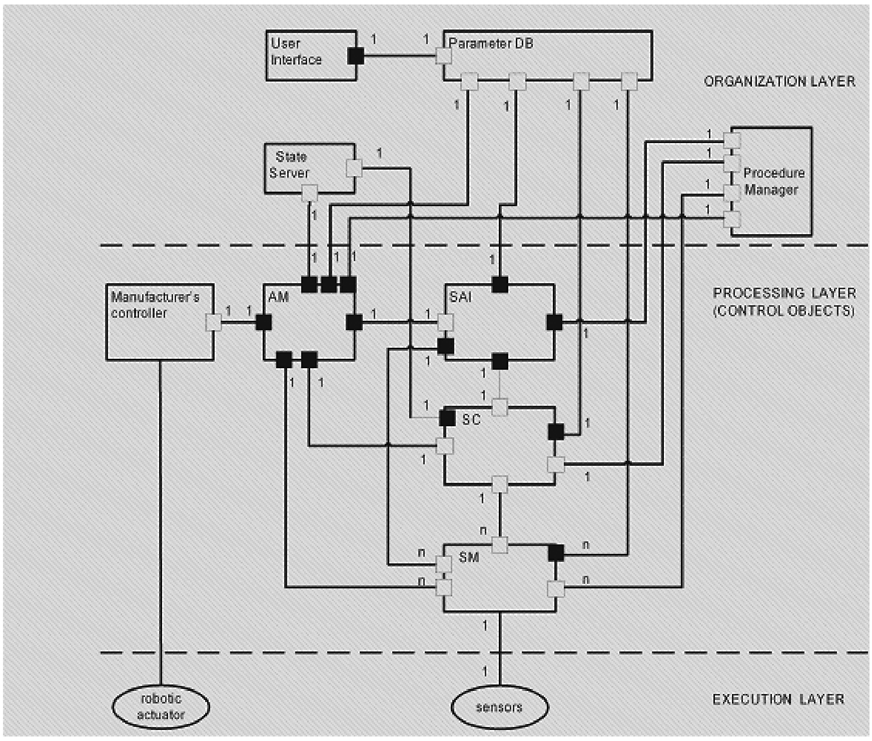


Fig. 4. The decomposition of the processing layer system to control objects (capsules) and their interconnection with modules of the organization and the execution layer (black and white boxes indicate client and server ports correspondingly). It is composed by the *AM* (which communicates with the manufacturer's controller), the *SAI*, the *SC* and the *SM* capsules. The capsules' activation and states are controlled by the *ProcedureManager*. The system parameters are read from the *ParameterDB*, and the predefined states are provided by the *StateServer*. The interface with the user is executed by the user interface, which communicates with the processing layer through the *ParameterDB*.

The framework has to be independent of sensor - specific and industrial robot controllers – specific implementations in order to provide integration of every sensor to any industrial controller type. This is achieved by common communication rules (protocols) for all sensors and controllers.

The framework has to be independent of specific control schemes to ensure wide applicability. It is achieved by defining separate components handling the control scheme – specific aspects of the design (*AM*, *SC* capsules as will be mentioned in next section.
The framework has to be independent of specific vision algorithms and recognition patterns (features) as well. It should allow the organizational layer to change the used algorithms for a task at run time without disturbing the execution at the control object layer. The use of the "strategy" pattern (see e.g., (Douglass 2002)) allows for substitution of algorithms at run-time from higher architectural layers without altering the control flow of the lower layers; it applies to it also facilitates the easy integration of existing algorithms. It also allows easy incorporation of existing code or frameworks. In this manner the fulfillment of the requirements for reliability, efficiency, integrity, usability, maintainability, flexibility, testability, and reusability can be achieved.
The data structures used for handling sensory information are variable mainly due to the type of the acquired data. Therefore the related structures should be flexible enough to incorporate data generated by a wide variety of sensors. The use of class templates that are available in C++ provides a solution to the problem.
A balance has to be achieved between flexibility and usability. The framework should make as less assumptions as possible but on the other hand the less assumptions made the more effort is required to build a system. A good trade-off between flexibility and usability is provided if we build a library of components, which is enriched as long as more applications are implemented.
The implementation of the design decisions at the control object layer is demonstrated in the following subsections. At the class level the design followed does not differ significantly from popular approaches and thus will not be analyzed here.

6. Control objects

This work focuses on the control objects layer, which offers its services independently of the organization, execution and its underlying layers with regard to the open controller architecture. The control objects are usually associated with a particular sensor or actuator type or they can just provide services based on input from other control objects. The sequence of the provided services is defined by the organization level (procedure manager, user interface) and is influenced by the environment.
For the modeling of the system structure we use the UML notation for real-time systems that is proposed in (Selic & Rumbaugh 1998), where the notions of capsules, ports and protocols are defined. Capsules are complex, physical, possibly distributed components that interact with their surroundings through one or more signal-based boundary objects called ports. A port is a physical part of the implementation of a capsule that mediates the interaction of the capsule with the outside world— it is an object that implements a specific interface. Each port of a capsule plays a particular role in a collaboration that the capsule has with other objects. To capture the complex semantics of these interactions, ports are associated with a protocol that defines the valid flow of information (signals) between connected ports of capsules. In the presented design the capsules are the high level system components (control objects), the ports are the interface classes and the protocols are the communication schemes between the components. These are presented in Fig. 4 and explained in the following.

```
/* Initialization */
{COMMUNICATION WITH DB AND INITIALIZATION OF LOCAL PARAMETERS}

/*Main loop*/
while {CONDITION}
{
   {RECEIVING INPUT}
   {PROCESSING INPUT}
   {SENDING OUTPUT}
}

/* Exit */
{STORING TO DB AND MEMORY DEALLOCATION}
```

Table 1. Execution scheme of a system process with the initialization phase, the main loop and the exit phase.

Generally the role of the *Controller* that was presented in Fig. 3 is to receive the sensory data from the sensors, to calculate the desired state and communicate the next state to the actuator. It may be divided to the *Manufacturer's Controller* (*MC*), which includes the software and hardware provided by the robot vendor for motion planning and motion control, and the sensor integration modules. These modules are displayed as capsules in Fig. 4 and are the *Actuator Manager* (*AM*), the *Sensor Manager* (*SM*), the *State Calculator* (*SC*) and the *Sensor-Actuator Interface* (*SAI*); their communication with the organization and execution layer modules is also presented.

Before we proceed to the description of the processing-layer capsules it is necessary to describe briefly the organizational level. The role of the *Procedure Manager* is to issue high level commands using events coming from the processing layer. It activates and defines the states of all the processing layer capsules. The *StateServer* is the module of the organization level that maintains the pre-defined robot states, which may include pose, velocity and acceleration. The states may be retrieved from a database or may come as input after online calculation. The *ParameterDB* stores all the parameters and attributes that are useful for the execution of the capsules in the processing layer. The parameters' update is performed online and offline. *ParameterDB* may also act as a link between the *User Interface* and the processing layer.

We begin the description of the processing layer capsules with the *AM*. Its role is to calculate the next robot state and to forward it to the *MC* for execution. The *AM* receives at each robot cycle the current end-effector state from the *MC* and the desired state from the sensory modules and then it calculates the next state (e.g., using a control law - state regulation which helps to avoid the unwanted end-effector oscillations); the next state is then sent to the *MC*. Different control laws can be applied to each DOF. The robot interpolation cycle and the sensor cycle may differ significantly (multi-rate system). Therefore for closed loop control sensor-guided tasks the robot needs to be coupled with the sensors by using the intermediate capsule *SAI*; *SAI* receives asynchronously the desired state from the sensory modules whenever it is made available by *SC*; *SAI* also forwards the current state - that is calculated using sensory input - synchronously to *AM* at each robot interpolation cycle.

The role of the *SM* is to operate at the image level and to extract the image features from the sensory data. On the contrary, the role of the *SC* is to operate at the workspace level and to use the image-level measurements of *SM* to calculate the environmental state (e.g., physical pose) of the end-effector or the target objects. The *SM* sends data requests to the sensors and receives from them the sensory data. After image processing it outputs to the SC the feature measurements. The *SC* receives the feature measurements from the *SM* along with the current actuator state from the *AM* through *SAI*. It outputs to *AM* through the *SAI* the desired state of the end-effector.

Each of the aforementioned capsules operates within a loop in a system process or thread. The scheme of each of those system processes is displayed in Table 1. During initialization the initial local parameters are loaded to memory from the *ParameterDB* and the appropriate memory blocks are initialized (2). The main loop (5-10) performs the processing work. At the beginning of the loop the data from other processes and the parameter database (when appropriate) are read (7). Then the data become processed (8) and the results are sent to the other system capsules or to the organization level (9). The integration platform may implement synchronous or asynchronous communication from either or both sides. The main loop commands may vary according to the capsule state. During the exit procedure, which is executed when the task is finished, the memory is released and the data may be stored into the *ParameterDB*.

The implementation of the design framework uses the services of a communication layer (Fig. 3), which has to be deterministic to cover the needs of a real-time industrial system. The respective communication is performed through the ports presented in Fig. 4.

The communication that is performed within the system follows the producer - consumer concept. A method for communication of modules that reside in different processes is the "proxy" pattern. In this pattern a pair of proxy objects, each of them residing on a different thread/process are used. The local calls to the proxy objects are mapped to inter-process messages (message destinations may be located either at compile time or at run-time through broker objects).

The synchronization protocols are distinguished in synchronous and asynchronous. The "message queuing" pattern can be used for implementing the asynchronous port communication protocol, e.g., through queued messages of unit length. A synchronous protocol can be implemented via the "rendezvous" pattern. The basic behavioral model is that as each capsule becomes ready to rendezvous, it registers with a Rendezvous class and then blocks until the Rendezvous class releases it to run. A detailed description of the use of the proxy, message queuing and rendezvous patterns for inter-process communication and synchronization may be found in (Douglass 2002). In the applications that we created using the proposed framework the ports were implemented as proxy objects.

7. Use cases implementation

In the following we show how to implement the use cases by employing the control objects defined in section 6. At the beginning we briefly describe the tasks that do not require sensory input and namely the *manual movement*, the *state reading* and the *teaching*. Then we describe the general *pre- and post-conditions*. Finally we discuss the interactions for the tasks that require sensory input, which are the *parameterization*, the *training* and the *sensor-guided movement*.

7.1 Operations without sensory feedback

During *manual movement* the user defines through a user interface module the desired robot state, which results in robot movement. Normally the robot manufacturers provide user interface units through which the user is able to move the robot. The user input is then passed to the *StateServer* and the robot reaches the new state after several interpolation cycles calculated in *MC*; for such tasks no third-party software modules are necessary.
During *state reading* the desired trajectory is read from the organizational level and executed without sensory input. The desired states come from the *StateServer*. The state reading task is usually executed before and after tasks that need sensory data; the robot moves without external sensing until a state is reached (e.g., a nominal pose) at which a task that requires sensory input has to be executed e.g., "blind" movement from a "home" robot position to a nominal position where a grasping task is to be executed. The trajectory has been previously recorded during the teaching task described next.
The *teaching* task is performed in order to record a trajectory (interpolation nodes) that will be used later for robot guidance without sensory input (state reading). Its execution has as follows: while the user moves the robot along the desired trajectory each intermediate state is received synchronously from the robot controller and sent to the *StateServer* for storing.

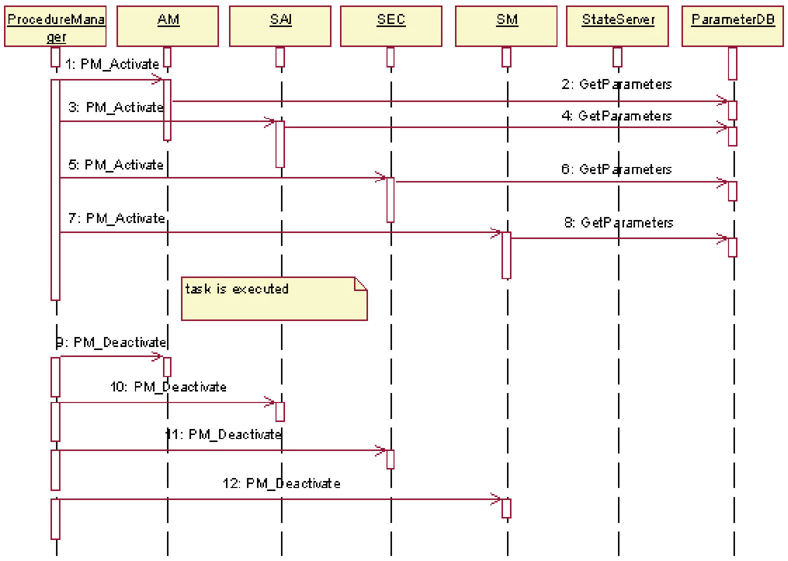


Fig. 5. The general pattern for task sensor-guided movement.

7.2 Pre- and post conditions

We define a common pattern for the conditions that must be satisfied before the invocation of a use case (task) as well as what happens at the end of its execution; this is displayed in Fig. 5.
Before the task invocation the *ProcedureManager* activates the capsules that participate in this specific task and sets their states (1, 3, 5, 7). Through this activation message(s) the ProcedureManager also sends to the capsules the unique id number of the task that is to be executed. This id identifies the type of the task as well the environmental or (and) system status parameters as they were (will be) defined during parameterization or training. Using this id each capsule is able to load the task-specific parameters from the parameter database e.g., threshold values for image processing, control variables for regulation etc. (messages 2,

4, 6, 8). This communication was previously mentioned in line 2 of Table 1. After the task execution the participating capsules are deactivated or eventually set to idle state (messages 9, 10, 11, 12). Before deactivation the new parameters or results are stored into the *ParameterDB* (line 13 of Table 1).
In the tasks that are presented in the following these pre-and post - conditions apply but will omit their presentation for simplicity.

7.3 Parameterization

During the offline parameterization task the user changes manually the parameters used by one or more capsules and is able to evaluate the impact of this change. The parameters are changed to the *ParameterDB* through the user interface and then they are read asynchronously by the active capsules and are applied. If appropriate the parameters may be saved by the user to be used in future tasks. The interactions for the parameterization of an *SM* capsule are given in the following:
The *ProcedureManager* instructs the *SM* to measure (1). The measurement is executed (2) and the results are sent to the *ParameterDB* (3), from where they are accessed (asynchronously) by the User Interface (4). The steps (2-5) are repeated until the user changes an *SM* parameter e.g., a region of interest (6). The new parameters are read by the *SM* (7) and new measurements are performed (8) and written to the *ParameterDB* (9). The new measurements are read asynchronously by the User Interface (10). The steps 7-10 last for a sensor cycle and are performed until the user decides to store permanently the content of the *ParameterDB* (11).

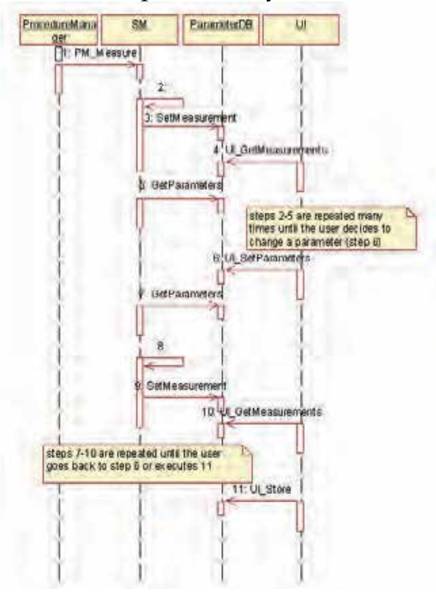


Fig. 6. A parameterization example for the *SM* capsule. The user changes the *SM* parameters and the *SM* reads them through the *ParameterDB* and applies them in processing. Finally the parameters are saved by the user.

7.4 Training

A sub use case of parameterization with particular interest is the training, which is an automated calculation of some sensor-related system parameters. An example of training is the calculation of a feature Jacobian, which provides the system state from the image features; another example is the calculation of the camera pose relative to the robot end-effector (calibration). In these training tasks the end-effector moves stepwise along a predefined training trajectory for each degree of freedom of the task space. During each training step the sensor subsystem measures the features on the images that are acquired synchronously by the cameras. When the training movement is completed the required parameters are calculated based on the feature measurements; they may be task-specific (e.g., feature Jacobian matrix) or system-specific (e.g., camera pose with respect to robotic hand). Finally the calculated parameters are stored into the database.

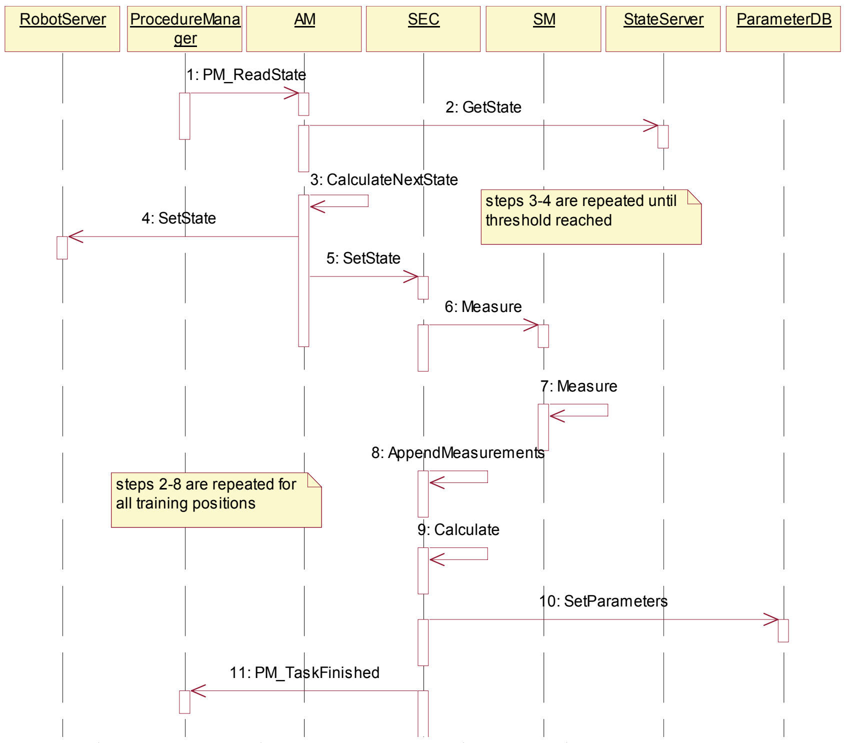


Fig. 7. The capsule interaction during training. When a task requiring sensory input is to be trained the end-effector moves stepwise along a predefined training trajectory for each degree of freedom of the task space and the sensor subsystem measures the image features. When the training movement is completed the parameters are calculated and then stored into the database.

The capsules that participate in training are the *AM*, *SC*, *SM* that communicate with the *StateServer*, *ParameterDB*. The *StateServer* holds the states defining the training movement. The training interaction using a single *SM* instance is presented in detail in Fig. 7 and has as follows:

1. The *ProcedureManager* commands *AM* to read the desired training state from the StateServer.
2. *AM* requests synchronously from *SC* the state vector that defines the next training End Effector State (EES).
3. After receiving the requested state the next EES is calculated using a control law.
4. *AM* sets the next EES to the *RobotServer* to move the robot. The current EES is returned.

Steps 3-4 may be repeated until the distance between the current and the desired training state becomes smaller than a predefined threshold, because high positioning accuracy is required; another scenario is that a timeout occurs without achieving the desired positioning accuracy and in this case we have an exception that either stops training or warns the user. Assuming the first case we have:

5. *AM* sends to the *SC* the currently achieved state, which acts as a trigger to activate the sensors and measure. *AM* waits until the measurement execution.
6. *SC* requests from *SM* to measure and remains blocked waiting for response.
7. *SM* executes the measurement. After measuring, the *SC* receives the measurements and becomes unblocked.
8. *SC* stores the measurement vector. Then the *AM* becomes unblocked.

Interaction steps 2 to 8 are repeated for each training step for all degrees of freedom of the task space. After loop completion:

9. *SC* calculates the required parameters.
10. The parameters are sent to *ParameterDB* where they are stored.
11. The *ProcedureManager* is informed about the task termination.

After step (11) training is finished. The same procedure has to be repeated for all tasks that need training. The steps (2-4) are executed in the robot interpolation cycle, while the duration of step 7 is decided mainly by the duration of the sensor cycle.

7.5 Sensor-guided movement

During sensor-guided movement the robot moves based on the desired states that are calculated after processing sensor input using the pre-defined parameters. This movement may be performed using closed-loop control at the end-effector level, or open loop schemes such as the "look-then-move" or "scan-then-move" that will be presented in the following.
In this task the actuator manager requests from the sensor – actuator interface the measured EES. Due to the difference between the robot and the camera sampling rate the measurement may not be available; in that case a zero state vector is received; else from the state vector (desired state) a new vector is calculated (using a regulator for the reasons explained in section 4.2) and sent to the robot controller. The procedure is repeated until the difference between the desired state and the current becomes very small.
The task execution is performed in real time using the parameters defined during parameterization and training (Fig. 8). The participating subsystems are the *AM*, *SAI*, *SC*, *SM* and the *StateServer*, *ParameterDB*.
The *RobotServer*, *AM* and *StateServer* operate at robot interpolation rate, while the *SC* and *SM*, operate at sensor sampling rate. The *SAI* communicates asynchronously with *AM* and *SC*. Therefore the relative order of the signals between subsystems operating in different rates is just indicative e.g., step 12 is not certain to precede 13. More specifically the interaction has as follows:

1. The *ProcedureManager* commands the *AM* to execute the task using sensory feedback.
2. *AM* requests an EES vector from *SAI*. The *SAI* has no available state error vector, due to the fact that no sensor data have been acquired yet and thus returns a zero vector.
3. *SAI* requests the error from the *SC* asynchronously, triggering the sensor-related part of the system to measure.
4. *AM* calculates from the error and the current state the desired EES using a regulator.
5. *SC* requests a measurement from *SM* synchronously, so that *SC* will be able to calculate the state error.

6. *AM* sends to the robot the correction vector.
7. The call is executed similarly to (2) and zero error is returned since there is no measurement available yet.
8. Similarly to (4) the next EES is calculated using zero error.
9. The next state is set to the robot.
10. Similar to (7) since no measurement available.
11. Similar to (8).
12. Data acquisition and processing are finished and *SM* returns to *SC* the feature measurement vector.
13. Similar to (9).
14. *SC* calculates the EES based on the new measurements.
15. *SC* finished the calculations and calls *SAI* asynchronously and sets the current error vector.
16. *AM* requests from *SAI* the calculated error vector and *SAI* returns it.
17. *AM* calculates the desired EES from the input error using a regulator.
18. *AM* sends to the robot the state calculated in (17), which will move the robot towards the desired EES.

The steps 2-18 are repeated until a threshold is reached. Then:

19. *AM* informs the *ProcedureManager* about the task termination.

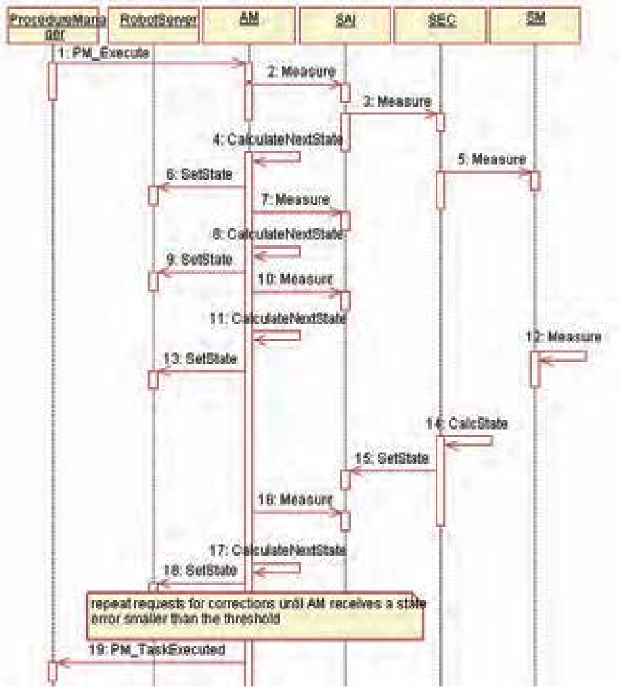
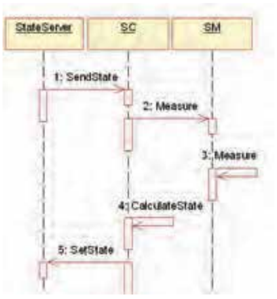


Fig. 8. The capsule interactions during sensor-guided movement for a closed loop system. The actuator manager requests from the *SAI* the measured error of the EES, the *SAI* returns the current state error or zero (due to the different camera and robot interpolation rate) and the regulated pose is sent to the robot controller. The procedure is repeated until the state error becomes very small.

The task of sensor-guided movement in open-loop configurations is much simpler than the respective closed-loop task. We will examine the „look-then-move“ and the „scan- then-move“ cases. In the former the sensors acquire a local view of the target, while in the latter the sensors acquire a global view of the workspace before executing the task.
In the "look-then-move" scheme, when the robot is at a nominal pose (without moving) the sensors acquire the data, these data are processed, the new state is extracted and then the end-effector moves to reach the required pose. Here there is no need for an *AM* capsule due to the fact that there is no need to apply a regulator at the end-effector level.
In Figure. 9. the general interactions at the control object layer is described. The role of the *StateServer* here is to provide the current end-effector state to the *SC* and to communicate the calculated states based on the sensory data to the rest modules of the organizational level and to the robot controller.
The *StateServer* (after receiving a command at the organizational level) sends to the *SC* the current end-effector state (1) and then the *SC* requests synchronously from the *SM* to acquire sensory data and provide image measurements (2). When the *SM* is ready with data processing (3) the *SC* receives the measurements and continues with state calculation (4). The calculated state is then sent to the *StateServer* (5). Then the state is forwarded to the robot controller.
In the "scan-then-move" approach the robot moves initially to a predefined set of poses, where the sensor is activated and thus a part of the workspace is scanned. The acquired images are processed for feature extraction and registered for each pose using the end-effector pose (or perhaps the images are processed after the execution of the scan movement). From the extracted image features the end-effector states are calculated in order to perform the task.
The *StateServer* (after receiving a command from the execution layer) sends the current state to the *SC*. The *SC* requests synchronously from the *SM* to acquire data and measure (2); when this is done (3) the measurements are registered (4) and the *ProcedureManager* becomes informed about the end of the registration for the current end-effector pose. Steps 1-4 are repeated for all predefined poses; then the next end-effector states are calculated (6) and sent to the *StateServer*. The *StateServer* forwards them appropriately to the robot controller.



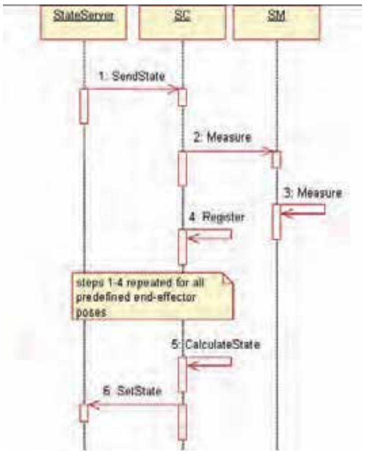


Fig. 9. Sensor-guided movement for a. look-then-move scheme b. the scan-then-move execution scheme.

8. Applications

The applicability of the framework has been verified through the implementation of industrial applications. The UML language has been used for modeling and the C/C++ languages have been used for coding. The first application presented here is the sunroof placement fitting, which aims to fit a sunroof onto a car body on the automobile production line (Kosmopoulos et al. 2002).
We used the 6-DOF manipulator K-125 of KUKA with the KRC-1 controller, which permits the employment of external software for control at the end-effector level. The task of fitting the sunroof on the car body was performed using four CCD cameras, monitoring the four corners of the sunroof opening. The corners were identified as the intersection points of the monitored edges (Fig. 10).
The general scheme introduced in Fig. 4 has been used. For this application four instances of the *SM* capsule have been employed and one of the *AM*, *SAI* and *SC*. As regards the *SM* capsule, the acquired data were of type char due to the monochrome camera images and thus image objects of type unsigned char were defined. In the *SC* capsule we extracted the desired robot state through an image-based, endpoint open loop approach. For the system state calculation we used an extended Kalman filter. The system state was given by the vector:

$$W_k=[x, x', y, y', z, z', a, a', b, b', c, c']^T{}_k$$

which refers to the end-effector pose error and the corresponding velocities with respect to a nominal state, which is achieved when the sunroof is correctly placed on the target. For the calculation of the end-effector pose we have used an over-determined system through measuring the image coordinates x_i, y_i of the four corner points. The measurement vector $\mathbf{f}_k$ is given by the following equation:

$$\mathbf{f}_k = [x_1, y_1, x_2, y_2, x_3, y_3, x_4, y_4]$$

The system Jacobian has been assumed to be locally constant and has been calculated through training. We have also used classes for coordinate transformations from the target coordinate system to the end-effector, the robot base and other coordintate systems.
For the *AM* capsule we have implemented PID control laws for each degree of freedom. The *SAI* was programmed to send the calculated state to the *AM* as soon it was requested. A zero vector was sent until the next calculated state was available from *SC*.
All the use cases described in section 5.2 have been addressed. The closed loop communication sequences described in sections 7 for a single *SM* capsule in this application were executed in parallel for the four *SM* capsules.
A second industrial application implemented using our generic framework, addresses the depalletizing (or robotic bin picking, or pick and place problem) in the context of which a number of boxes of arbitrary dimensions, texture and type must be automatically located, grasped and transferred from a pallet (a rectangular platform), on which they reside, to a specific point defined by the user (Katsoulas & Kosmopoulos 2006).
We used the KR 15/2 KUKA manipulator, a square vacuum-gripper, and a time-of-flight laser range sensor (model SICK LMS200), which provides 3D points detected by a reflected laser beam. The sensor is mounted on the gripper and the latter is seamlessly attached to the robot's hand (Figure 10).

Due to the fact that only a line is acquired in each sensor cycle, the robot moves the sensor over the pile while acquiring data so that the whole pile can be viewed. The "scan then move" process is utilized for registering one dimensional scan lines obtained by the laser sensor during the robot hand movement, into a 2.5D range image. Finally the object states are extracted from the constructed image and the boxes lying on the upper layer, which is visible to the sensor, are grasped one by one and placed to a new position. For the next layer the same procedure is executed until the pallet is empty.
The general scheme presented in Figure 4 has been implemented omitting the *AM* (due to the fact that no closed-loop control functionality is needed), and the *SAI* (because the robot moves only when the data acquisition is finished and no additional synchronisation is needed); their connections are also omitted.
The communication with the manufacturer's robot controller is achieved through the organizational level (*StateServer*).
The processing layer of the application consists of one instance of the *SM* and the *SC* capsules; the *SM* controls the sensor for data acquisition and the *SC* calculates the state of the target objects, in other words their pose in the world coordinate system (target localization).
Sensor-guided movement follows the "scan then move" pattern, described in figure 9b. Firstly, while the robot is at the initial scan position the *StateServer* sends the current robot state to the *SC* thus triggering a message from *SC* to *SM* to acquire data. The *SM* acquires an image (scan line), which is returned to *SC*. The *SC* receives as input the range image corresponding to the current configuration of the object on the pallet; using the current robot state the coordinates of the image elements are calculated in the world coordinate system. These steps are repeated for all predefined end-effector poses and then the scan lines are combined to a single image. This range image, which holds information about the environment, is processed and from it the target object states (poses) are finally extracted and sent to the organizational level through *StateServer*.

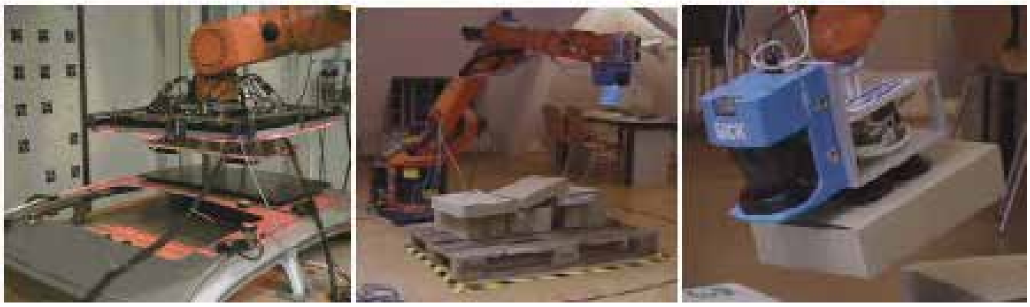


Fig. 10. (a) The sunroof fitting robot using four cameras in a closed loop control scheme and (b), (c) the depalletizing system during scanning and grasping

9. Conclusions and future work

In this chapter we have introduced a design framework for developing software for integrating sensors to robotic applications, provided that open controller architecture is available. The design is general enough and enables the integration of the popular sensors to any open controller using a wide variety of control schemes. By keeping some simple

interfacing rules it is possible to replace modules, e.g., for implementing control laws for the actuator DOFs or for processing the sensor data without much effort. This enables using the freely available or commercial libraries-components maximising the benefit of reusing well – tested modules. It is also possible to integrate multiple instances of the related processes-capsules provided that the interfaces will be the same. The need for a synchronisation module between sensors and actuator in closed loop control schemes has been underlined.
The most common use cases for industrial applications have been presented and the related interaction patterns have been demonstrated. They include some typical open loop and closed loop control schemes. Furthermore, two concrete examples of the development of industrial applications facilitated by the presented framework were demonstrated. Some more applications, not presented here due to confidentiality issues, ranging from robotic measurement to grasping have already employed this approach.
Although the development of design solutions and frameworks for robots is a challenging task we have managed to identify the similarities and differences for the sub-domain of sensor-guided industrial manipulators. The development time for the related applications has been significantly reduced to approximately one third of the initial development time due to reusability of design and components. A good trade-off has been found between over-generalizing and the application-specific design by building concrete examples based on the general principles.
The presented work targets industry-specific systems, where the environment is structured to a significant level. This means that the decisions/use cases taken at the organization level are limited. More complex behaviors e.g., concerning mobile robots moving in unstructured environments are not addressed here.
Anyone who deals with the development of robotic applications can benefit from this work, especially those that seek to couple the mechanical flexibility of industrial robots, with the flexibility to "build" various diverse applications with common characteristics.
In the future the integration in our scheme of a sophisticated configuration tool, such as glue code (code level) and configuration tool in textual or graphical mode (integration and application level), may facilitate the flexibility and the rapid deployment of sensor-specific industrial applications, thus making our framework highly reconfigurable in a dynamic manner. Another objective is to provide functionality through a black box framework from an open integrated environment for the development and testing of the control objects. This environment will support the use of many sensors and actuators and will include their models for simulation purposes employing a variety of vision and control algorithms.

10. References

Anderson, R. J. (1993). SMART: A Modular Architecture for Robots and Teleoperation, *IEEE International Conference on Robotics and Automation*, pp. 416-421, Atlanta, Georgia, May 1993

Corke, P. I. (1993). "Video rate robot visual servoing", In:*Visual Servoing,* K. Hashimoto (Ed.), vol. 7, Robotics and Automated Systems, pp. 257-283, World Scientific, ISBN 9810213646, Singapore

Coste-Manière, E., Turro, N. (1997). The MAESTRO language and its environment: Specification, validation and control of robotic missions. *IEEE International Conference on Intelligent Robots and Systems*, Vol. 2, pp. 836-841, Grenoble, France, September 1997.

Douglass, B.P. (2002). *Real-Time Design Patterns: Robust Scalable Architecture for Real-Time Systems*, Addison Wesley, ISBN 0201699567.

Hager, G., Toyama, K. (1998). XVision: A portable Substrate for Real-Time Vision Applications, *Computer Vision and Image Understanding*, Vol. 65, No 1, January 1998, pp.14-26, ISSN 1077-3142

Hutchinson, S., Hager, G., Corke, P. (1996). A tutorial introduction to visual servo control, *IEEE Transactions on Robotics and Automation*, Vol. 12, No 5, May 1996, pp. 651-670, ISSN 0882-4967.

Intel (2006). Open source computer vision library, http://www.intel.com/research/mrl/research/opencv/

Katsoulas, D.K., Kosmopoulos, D.I, (2006). Box-like Superquadric Recovery in Range Images by Fusing Region and Boundary based Information, *Int. Conf. on Pattern Recognition*, Hong Kong, to appear.

Kosmopoulos, D.I, Varvarigou, T.A, Emiris D.M., Kostas, A., (2002). MD-SIR: A methodology for developing sensor-guided industry robots, *Robotics Computer Integrated Manufacturing*, Vol. 18, No 5-6, Oct-Dec 2002, pp. 403-419, ISSN 0736-5845.

Mazer, E., Boismain, G., Bonnet des Tuves, J.M., Douillard, Y., Geoffroy, S., Doubourdieu, J.M., Tounsi M., Verdot, F., (1998). START: An application builder for industrial robotics, *Proceedings of IEEE International Conference on Robotics and Automation*, Vol. 2, pp. 1154-1159, May 1998, Leuven, Belgium.

MVTec (2006). http://www.mvtec.com/

Selic, B., Rumbaugh, J. (1998). "Using UML for modeling complex real-time systems", White Paper.

Sperling, W., Lutz, P., (1996). Enabling open control systems – an introduction to the OSACA system platform, *Robotics and Manufacturing,* Vol. 6., ISRAM 97, New York, ASME Press, ISBN 0-7918-0047-4.

Orca (2006). Orca: components for robotics http://orca-robotics.sourceforge.net/index.html

Toyama, K., Hager, G., Wang, J. (1996). "Servomatic: a modular system for robust positioning using stereo visual servoing", *In Proceedings International Conference on Robotics and Automation*, Vol.3, pp. 2636-2643, April 1996, Minneapolis, USA.

Valavanis, K.P., Saridis, G.N. (1992). *Intelligent Robotics Systems: Theory, Design and Applications*, ISBN 0792392507, Boston.

ECV-Net –IUE (2000). http://www-sop.inria.fr/robotvis/projects/iue/main.html

VXL (2006). The vxl homepage, http://vxl.sourceforge.net

Vaughan, R. T., Gerkey, B., and Howard, A. (2003). On device abstractions for portable, reusable robot code, *In Proceedings of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pp. 2421-2427,October 2003,Las Vegas, USA.

Montemerlo, M., N. Roy and S. Thurn (2003). Perspectives on standardization in mobile robot programming: The Carnegie Mellon Navigation (CARMEN) toolkit. *In Proceedings of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pp. 2436-2441, October 2003, Las Vegas, USA.

Utz, H., Sablatnog, S., Enderle, S., and Kraetzschmar, G. (2002). MIRO - middleware for mobile robot applications, *IEEE Transactions on Robotics and Automation*, Vol. 18, No 4, August 2002, pp. 493-497, ISSN 0882-4967.

Nesnas, I. A.D., Simmons, R., Gaines, D., Kunz, C., Diaz-Calderon, A., Estlin, T., Madison, R., Guineau, J., McHenry, M, Hsiang Shu, I., Apfelbaum, D., (2006). CLARAty: Challenges and Steps Toward Reusable Robotic Software, *International Journal of Advanced Robotic Systems*, Vol. 3, No. 1, March 2006, pp. 023-030, ISSN 1729-8806.

2

Joint Torque Sensory in Robotics

Farhad Aghili
Canadian Space Agency
Canada

1. Introduction

Joint Torque sensory Feedback (JTF) can substantially improve the performance of a robotic system. JTF makes it possible to achieve dynamic control of a robotic manipulator without the need for modeling its link dynamics. Moreover, it has been proved that JTF can achieve a precise torque tracking in a manipulator joint by compensating the effect of joint friction and actuator nonlinearities. Despite these advantages, accurate joint torque measurement encounters several practical challenges. Since much of the torque/force reaction of the link load on the joints appears in the form of nontorsional components, e.g. overhung force and bending moment, the torque sensing device has to not only bear but remain insensitive to these force/moment components. In addition, it is desirable to design the structure of the sensor such that it generates a large strain for a given load torque and therefore has a high sensitivity. However, this is in conflict with the high-stiffness requirement for minimizing the joint angle error introduced by the sensor.
The main objectives of this chapter are twofold: Firstly, in Sections 2 and 3, we describe the technical challenges and practical solutions to accurate measurement of joint torques in the presence of the non-torsional components in a robot joint. For a torque sensing device, different mechanical structures will be examined and their properties, such as sensitivity and stiffness in different directions and decoupling, will be explored. This allows a systematic design of a sensor structure which is suitable for torque measurement in a robot joint. Finally, the state-of-the-art and design of a torque sensor for robots will be presented in Section 4. The design achieves the conflicting requirements of high stiffness for all six force and torque components, high sensitivity for the one driving torque of interest, yet very low sensitivity for the other five force/torque components. These properties, combined with its donut shape and small size make this sensor an ideal choice for direct drive robotic applications. Experimental data validates the basic design ideas underlying the sensor's geometry, the finite element model utilized in its optimization, and the advertised performance.
The second objective is to describe the application of joint torque sensory feedback (JTF)in robot control. The main advantages claimed for JTF are (i) it can simplify the complexity of the system dynamics by eliminating its link dynamics; and (ii) the control system is inherently robust with respect to both external force disturbance and parameter variation. First, assuming both actuator and torque sensor are ideal, we describe robot control with JTF in Section 5. Then, development of an adaptive JTF is presented in Section 6that requires only the incorporation of uncalibrated joint torque signals, i.e., the gains and offsets of multiple sensors are unknown. Also, all physical parameters of the joints including inertia of the rotors, link twist angles, and friction parameters are assumed unknown to the

controller. Finally, in Section 7, JTF is modified to cope with actuator's finite bandwidth dynamics actuator, i.e., no ideal actuator. An optimal JTF is designed in the frequency domain that minimizes the control system's sensitivity to load torque disturbances and load dynamics. Experimental results demonstrate that the JTF remarkably improves the disturbance attenuation and load decoupling properties of a joint servo controller.

2. Sensing Joint Torque

The benefits of using joint torque sensory feedback to improve the performance of robotic system have been recognized in the robotics community. For example, joint torque feedback can be used to compensate the nonlinearities and modeling uncertainties of manipulator dynamics (Hashimoto, 1989a; Kosuge *et al.*, 1990;Aghili *et al.*, 2001) or simply those of actuators (Asada and Lim, 1985; deSilva *et al.*, 1987; Zhangand Furusho, 1998; Luh *et al.*, 1983). Moreover, in geared systems, the implementation of model based controllers is difficult without measuring the actual output torque, since the efficiency of gears depends greatly on the torque, and to a lesser extend, on the joint velocity, and yet this data is typically not made available by gear manufacturers. Thus there is a need for torque sensors that can be integrated simply between the actuator (and possibly gear) and the load. The sensor research described in this section was motivated by the lack of suitable sensors needed for our joint servo system.
Accurate joint torque measurement encounters several design challenges. In the design of robot manipulators, it is desirable that much of the torque/force reaction of the link load on the joints appears in the form of non-torsional components, because actuation then takes less effort. SCARA robot arm designs, for instance, prevent gravity torques from acting on the joint motors (Newman and Patel, 1991). However, since torque sensors are directly attached to the motor's distal links, they have to bear those potentially large non-torsional components. The first challenge is to measure torque with minimal influence from simultaneous and potentially large non-torsional components. In the following, we shall call the one axis of motor torque of interest the torsion. The other two torque and three force components, we shall call for simplicity the non-torsional components. The second challenge relates to the sensor stiffness. High torsion stiffness is important because any deflection adds positioning errors that cannot be compensated by the joint servo controller. To increase the signal-to-noise ratio and sensitivity of the sensor, it is desirable to design a structure that generates a large strain for a given load torque. However, the sensitivity introduces a torsion compliance that must be minimized. Thus there are two conflicting requirements: High stiffness and high sensitivity for torsion. A new solution to these two challenges will be described here and is distinct from existing designs. Some aspects of this research have been previously reported in (Aghili *et al.*, 2002a).
There is a large literature on the systematic design of six degree-of-freedom (dof) force/torque sensors (Hirose and Yoneda, 1990;Svinin and Uchiyama, 1995;Uchiyama *et al.*, 1991). It is important to note that the design criteria for one and six dof sensors are very different. For instance, isotropy (uniform sensitivity)is a desirable property of a six degree-of-freedom force/torque sensor, hence its elastic structure tends to be fragile and compliant in all directions. In contrast, the elastic sensitivity of a torque sensor has to be maximized only around its torsional axis.
Various techniques have been proposed to instrument geared motors for torque sensing (Hashimoto, 1989b; Luh *et al.*, 1983; Pfeffer *et al.*, 1989; Vischer and Khatib, 1995), while little attention has been paid to find an adequate structure for joint torque sensing (Asada

and Lim, 1985; Jacobsen *et al.*, 1991; deSilva *et al.*, 1987). Hashimoto *et al.* (Hashimoto, 1989b) utilized the elasticity of the flex-spline in a harmonic drive to measure the joint torque. This technique has the advantage of using the existing structural flexibility of the robots. However, eliminating the error caused by rotation of the wave generator is difficult, it requires a nontrivial gear modification, and this approach cannot be used in direct drive systems. Many researchers (Pfeffer *et al.*, 1989; Wu, 1985;Luh *et al.*, 1983;deSilva *et al.*, 1987; Asada and Lim, 1985; Vischer and Khatib, 1995) choose not to place the sensor directly at the joint shaft to avoid the detrimental effects of the support forces and moments. Pfeffer *et al.* (Pfeffer *et al.*, 1989) replaced standard attachment bolts in the PUMA 500 joints with a flexure instrumented with strain gauges. Wu (Wu, 1985) used a shaft with a thin hollow circular section that is supported by two bearings. Strain gauges are mounted on the thin section. Luh *et al.* (Luh *et al.*, 1983) cemented strain gauges on the connecting output shaft which is mounted to the flex-spline of the harmonic drive for each joint of the Stanford manipulator. Visher (Vischer and Khatib, 1995) integrated a torque sensor with the gear transmission, while Asada et al. (Asada and Lim, 1985) integrated strain gauges in the hub supporting the robot of a direct-drive motor. The strain gauges are cemented on three beams connecting the outer ring, mounted to the motor rotor, and the inner ring, which is coupled to the motor shaft. Asada et al. (Asada and Lim, 1985) cemented strain gauges inside the rotor of a direct-drive motor for torque measurement. Since these sensors are not mounted directly on the joints of a manipulator, the entire set of forces and moments are supported by the bearing set rather than the sensor structure. However, these sensors are not ideal because they can not account for the friction in the joint bearings. Moreover, the mechanical joints are complicated and sometimes bulky. In commercial torque sensors non-torsional components are not permitted or are highly restricted. Furthermore, they usually come in bulky packages, are built for shaft mounting and thus are not suitable for integration in a robot joint.

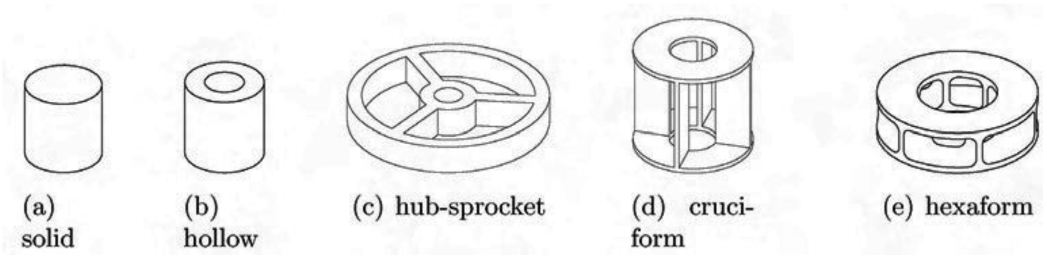


Fig. 1. Different structure for torque sensor.

Fig. 1 illustrates conventional geometries (a, b, c, and d) and the proposed hollow hexaform design (e) for torque sensors. Solid (a) and hollow (b) cylinders have been used extensively for joint torque sensing (Pfeffer *et al.*, 1989; Wu, 1985; Luh *et al.*, 1983; deSilva *et al.*, 1987; Wu and Paul, 1980) but are sensitive to non-torsional components (Wu and Paul, 1980). For this reason, they are usually mounted before the joint bearings so that the bearing can support the non-torsional components. In addition to requiring a bearing support structure, the main drawback of this method is that joint friction can not be observed by the sensor. Hub-sprocket designs Fig.1(c) have been used for geared joints (Hirzinger *et al.*, 2001; Vischer and Khatib, 1995) as well as direct-drive joints (Asada and Lim, 1985; Tani *et al.*, 1983). Although a better rejection to non-torsional components has been reported for this type of sensor (Vischer and Khatib, 1995), the structure is not adequate for a modular robot joint. This is because of the drastic change in the physical size between the input (inner ring) and output (outer ring) of the sensor. Rather this type of sensor should be integrated with the gear or with the rotor of a direct-drive motor, and hence it suffers from the same drawbacks as type (a) and (b) sensors.

The hollow cruciform design Fig. reffig: quad is used in commercially available torque sensors (Lebow, 1997). In this design, strain is induced mainly by bending of the wing elements. In order to achieve good sensitivity, the wing and sensor height is large, and as a result, the stiffness is low and non-torsional torques must be kept small. The proposed hollow hexaform sensor Fig. 1(e) is similar in its basic geometry to the hollow cruciform sensor (d) with only four wings. However, there are fundamental functional differences. Due to the increased number of wing pairs, and the shorter height, strain is induced primarily in torsion, resulting in a much stiffer sensor, and improved sensitivity. In addition, this design can be optimized to support non-torsional torques, making it suitable for direct drive robotic applications.

3. Sensor Design

In this section we describe how the new hollow-hexaform sensor achieves

(i) high sensitivity to torsion,
(ii) low sensitivity to non-torsional components, and
(iii) high stiffness in all axes of forces and moment.

3.1 Design for Decoupling

In general, torque measurements are derived from strain measurements at several locations on an elastic sensor body. Assuming a linearly elastic material, there is a linear relationship between the applied forces and torques and the resultant strains described by

$$\epsilon = C\, f, \tag{1}$$

where $\epsilon \in \mathbb{R}^m$ is the vector of m measured strains, $f \in \mathbb{R}^6$ is the generalized force/moment vector acting at the center of the sensor body where the z-axis and joint axis are identical, and $C \in \mathbb{R}^{m\times 6}$ is the sensitivity matrix whose elements c_{ij} denote the sensitivity of the *i*th strain gauge to the *j*th component of the generalized force/moment. This matrix permits the reconstruction of the torsion moment from the output signal with the gain vector. Unlike in 6-axis force/torque sensors, it is desired to reconstruct only the torsion moment n_z from the measured strains ϵ. However, the sensitivity matrix underlies the mechanical coupling transmitted through the force/moment sensor structure. Therefore, the sensor output should be decoupled from the non-torsional components of forces and moments. We show that one can use the additive properties of the Wheatstone bridge to achieve the decoupling without the need for any subsequent arithmetic. The resulting advantage is a reduction of instrumentation and the number of wires by completing the bridge wiring inside the sensor, and simplification of tedious calibration.

The question arises concerning the condition for which there exists such a mapping. It is necessary to consider each component of force to be a linear function of all strain gauge sensors in order to correct for the coupling. Let v_{out} and v_{ex} represent the output voltage and the excitation voltage of a half-bridge configuration, and GF denote the gauge factor

(Omega, 1995) of the strain gauges. Then, assuming every strain gauge pair constitutes a half-bridge circuit, the overall voltage output is given by $v_{\text{out}} = \kappa t^T \epsilon$ where $\kappa = 0.5GFv_{\text{ex}}$ is the gain of the strain gauges and $t = [-1, 1, -1, 1, \cdots]^T$ represents the gain signs corresponding to the negative and positive branches of a Wheatstone bridge circuit. Substituting ϵ from (1) into the latter equation, we obtain

$$v_{\text{out}} = \kappa t^T C f = \kappa \sum_{i=1}^{6} w_i f_i, \tag{2}$$

where w_i is the inner product of vectors t and the ith column of C. It is evident from (2) that, in the most general case, the sensor output is the superposition of weighted components of the generalized forces and moments transmitted through the sensor unless all weights related to the exogenous forces and moments are zero. That is, the decoupling is achieved if $w_1 = \cdots = w_5 = 0$ and $w_6 \neq 0$. In this case, the sensor output is solely proportional to the torsion torque $\tau_J = f_6$,i.e.,

$$v_{\text{out}} = \alpha \tau_J, \tag{3}$$

where $a = \kappa w_6$ represents the overall sensitivity of the sensor. That vector t is orthogonal to all columns of C matrix except the last one underlines a condition on the structure of a torque sensor by which the senor exhibits the decoupling. As a results, such as sensor is not sensitive to the supporting forces and moments transmitted through the structure of the sensor.

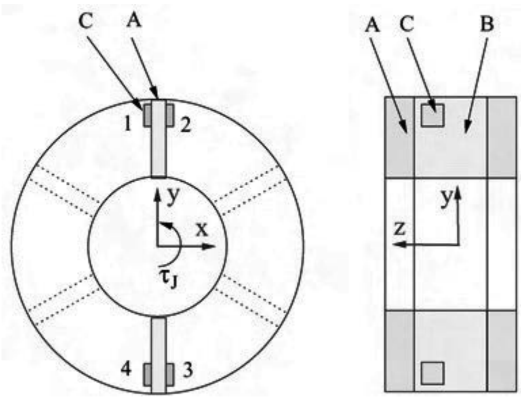


Fig. 2. Basic torque sensor structure. A: solid discs; B: elastic element; C: strain gauge.

For the candidate geometry in Fig. 2andwith four strain gauges located on the numbered locations, the parametric form of the sensitivity matrix can be derived from the symmetry of the structure with respect to the external forces and torques as

$$C = \begin{bmatrix} c_{11} & c_{12} & c_{13} & c_{14} & c_{15} & -c_{16} \\ c_{21} & c_{12} & c_{13} & c_{14} & -c_{15} & c_{16} \\ c_{21} & c_{32} & c_{13} & -c_{14} & -c_{15} & -c_{16} \\ c_{11} & c_{32} & c_{13} & -c_{14} & c_{15} & c_{16} \end{bmatrix}.$$

The jth column of the matrix represents the strain sensitivities in the four site locations with respect to the jth load case, e.g., the third column represents the sensitivity due to f_3.The identical elements in the matrix can be implied from the symmetric configuration of the structure with respect to different load cases. For instance, the symmetric condition of the

strain gauges with respect to the axial load, f_3, implies identical elements of the third column. Now one can readily verify that $w_1 = \cdots = w_5 = 0$ and $w_6 = 4c_{16}$and hence the structure satisfies the condition for mechanical decoupling.
There are two main reasons in practice that violate the above assumption of exact symmetry among the measured strains. First, strain gauges exhibit variations in their gauge factor. Second, the strain gauges will be placed on areas with high strain gradients. This makes the gauge outputs sensitive to placement errors. This can also be modeled as a gauge gain error. As a consequence, exact cancelation of the non-torsional components may not be achieved with the theoretical gain vector. By virtue of the linear mapping (1), the non-torsional components produce no output, if all elements of the sensitivity matrix except that for the last column are sufficiently small. This implies that the strain sensitivity to the non-torsional components has to be held to a minimum by mechanical design. This condition in conjunction with the decoupling property of the sensitivity matrix actually determines the capability of the sensor to reject the effect of non-torsional force/torque to the output and to provide a high fidelity output signal.

3.2 Maximum sensitivity

To increase the signal-to-noise (S/N) ratio and the resolution of the sensor, it is desirable to design the elastic component to provide large output signals, i.e., large mechanical gain α. Therefore one of the design criteria is to increase the torsional sensitivity, subject to not exceeding the allowable strain. In the absence of non-torsional components, the maximum attainable strain sensitivity depends solely on the material properties as the strain due to the maximum load should be close to the maximum allowable material strain or stress. However, non-torsional components produce strains which add to the strain caused by torsion. To ensure that the allowable maximum material strain is not exceeded, we consider the worst-case scenario where the generalized force/torque vector has its maximum force and moment. Then, in order to exploit maximum torsion sensitivity, c_{16}, the other sensitivity components, i.e. $\{c_{11}, \cdots, c_{15}\}$, must be minimized by proper geometry design. This design requirement is consistent with a decoupling property of the sensor. It is interesting to note that cylinders are mainly used in the design of commercial torque sensors. By elementary strength of material analysis, one can show that bending moments produce twice the stress than the same magnitude torsion moment. This is why shear and thrust forces and bending moments must be kept small in these sensors.

3.3 High torsional and bending stiffness

Torsional deflection degrades the position accuracy of the joint angle controller. Moreover, to increase the signal-to-noise ratio requires maximizing the sensor sensitivity. However, highly stiff sensors tend to be less sensitive ones. Therefore, one of the critical design challenges is to maximize the stiffness while maintaining high sensitivity. We propose η, called structure efficiency, which is the product of sensitivity and stiffness as a performance index to capture these contradictory requirements,

$$\text{(torsional sensitivity) (torsional stiffness)}$$
$$= \frac{4\kappa\epsilon_1}{\tau_J} \cdot \frac{\tau_J}{\delta} = 4\kappa\frac{\epsilon_1}{\delta}, \tag{4}$$

where δ is the torsional deflection. As mentioned earlier, the gain of the strain gauge, κ, is

independent of sensor structure. Moreover, ϵ_1/δ is a dimensionless variable which captures the ratio of the local and global strains. These facts suggest that η is a decisive factor in the sensor design and should be maximized. Moreover, since it is dimensionless, the index provides a useful basis for comparison of different size torque sensors. The index is maximized in elastic structures that produce high strain concentration in torsion. In theory, there is no limit on the strain concentration in an elastic body. However, the high strain concentration takes place in a very small area, which might be smaller than the physical size of available strain gauges. Moreover, since strain gauges average the strain field over their area, the detected strain can be significantly lower than the calculated value. Therefore, it is important to generate high strain over a sufficiently large area. This objective seems difficult to formulate analytically, but can be inspected by finite element methods.
Introducing a torque sensor in a robot joint adds flexibility. Although torsional flexibility can, in principle, be compensated via sophisticated controllers, deflection in the other axes is more problematic. Consequently, another design criterion dictates high stiffness in non-torsional directions. Fortunately, the requirements for low deflection and low strain sensitivity for non-torsional components are consistent. The structure shown in Fig. 2 exhibits high bending stiffness around the x-axis. However, its poor stiffness around the y-axis is a drawback. This problem can be simply solved by adding more wing pairs as shown in Fig. 1E. This improves the uniformity of the bending stiffness along different axes as well as the body stiffness. In general, all performance aspects of the sensor improve with the number of wing pairs, but since we will want to machine the sensor from one solid piece of metal, the limit is imposed by manufacturability. For this reason, we consider six wings in our design.

3.4 Practical shape considerations

Addition of a torque sensor to a robot joint must not require the redesign of the joint and should result in a minimal change in the manipulator's kinematics, in particular the link offset. Hence, a shape with a small width is desirable. Minimizing the effects of thermal stresses is a design factor that cannot be ignored. Motors are a source of heat that flows from the motor to the attached link through the sensor body. Therefore, it is desirable to have an axisymmetric design that constrains the heat to flow in the axial direction, where no position constraint usually exists. The common hub-sprocket designs are prone to thermal stresses because of the temperature difference between the hub and the wheel. Since the sensor is specifically designed for a direct-drive motor with hollow shaft, flange mounting is preferred. Finally, the body should be designed for ease of manufacture. It should be a monolithic structure, that is, the body should be machined from a solid piece of metal. This decreases the hysteresis and increase the strength and repeatability of the sensor. The hollow hexaform geometry shown in Fig. 1E satisfies these requirements.

3.5 Material properties and overloading

So far only geometric properties of the elastic body were considered. Nevertheless, the stiffness and sensitivity characteristics of the torque sensor are also determined by the material properties. The maximum allowable strain for foil strain gauges is typically 3 %, which is at least one order of magnitude higher than that of industrial metals ϵ_{allow}, making the materials the limiting factor for sensitivity. Furthermore, the stiffness depends linearly on Young's modulus E of the material. By virtue of Hook's law,

$$\sigma_{\text{allow}} = E\,\epsilon_{\text{allow}}$$

one can conclude that high sensitivity and stiffness are achievable simultaneously only by use of a high-strength material.
Because a linear response is desired from the sensor, the chosen sensor material must have a linear strain-stress relationship. Steel is the best available industrial material that has good linearity properties within a large stress range. Moreover, due to the oscillatory nature of the loading, steel can work with in finite fatigue life as the allowable strains are determined based on the endurance limit. The endurance limit or fatigue limit is the maximum stress under which mechanical failure will not occur, independent of the number of load cycles. Only ferrous metals and alloys have an endurance limit.
The sensor is designed for a nominal torque 300 Nm that is based on the endurance limit of mild steel, which is twice as much as the yield point. Hence the safety factor in torque overloading is two. Remarkably, FEM results demonstrated that the stress induced by bending moment is very low for the proposed structure. As a result the structure can resist bending moments as high as 2000 Nm, which is almost an order of magnitude higher than the nominal torque.

3.6 Thermal deviation

The gauge resistance and gauge factor of all known strain sensitive materials vary with temperature. The change in resistance with temperature for a mounted strain gauge is a function of the difference in the thermal expansion coefficient between the gauge and the sensor body and of the thermal coefficient of resistance of the gauge alloy. Self-temperature compensating gauges can be achieved for specific materials by processing the strain sensitive alloy such that it has thermal characteristics that compensate for the effects of the mismatch in thermal expansion coefficients between the gauge and the body of the sensor (Omega, 1995). The manufacturer of the strain gauge (OMEGA (Omega, 1995)) claims that their products accurately compensate the effect of temperature if the yare chosen according to specific coefficient of thermal expansion of material on which the gauges are mounted.

4. Design and Analysis

4.1 FEM Analysis

Once we had determined the basic hollow hexaform shape of the sensor, we used the FEM capabilities of IDEAS (Structural Dynamics Research Corp.) to optimize the sensor dimensions and to determine the size and placement of the strain gauges. Strain concentration is the design key to simultaneously achieve high torsional sensitivity and high stiffness. For maximum sensitivity, strain gauges should be located where maximum induced strains due to the torsion load occur. Since the strain field is averaged over the area covered by the strain gauges, it is very important first to determine the loci of the peak strain, and second to ensure the creation of a sufficiently large strain field. FEM is ideally suited to solve this problem.
The sensor body is modeled by solid elements as shown in Fig. 3A. Since the body is symmetrical in geometry and applied boundary conditions, it suffices to analyze only one half, provided that adequate position constraints are imposed on the nodes of the cutting plane. To simplify the FEM, small geometric features of the body are suppressed. Several load cases were investigated, corresponding to axial and shear forces as well as bending and torsion moments.
In our application, the maximum forces and moments are 1000 N and 300 Nm,

respectively. A preliminary stress analysis showed that the axial and shear forces have negligible elastic effects because they produce a uniform strain/stress field in the elastic body, resulting in a very weak maximum strain. In fact, the bending moment is the critical non-torsional component, and consequently two different load cases corresponding to the external torsion and bending torques are established for FEM. It is important to note that in robotic applications the maximum angular deflection due to external torques (which is amplified by the robot links)is a more restrictive constraint than linear deflection due to the forces. It has been investigated that the worst-case strain due to the bending load happens when its axis lies perpendicular to one of the wings, and consequently that axis is chosen for the bending.

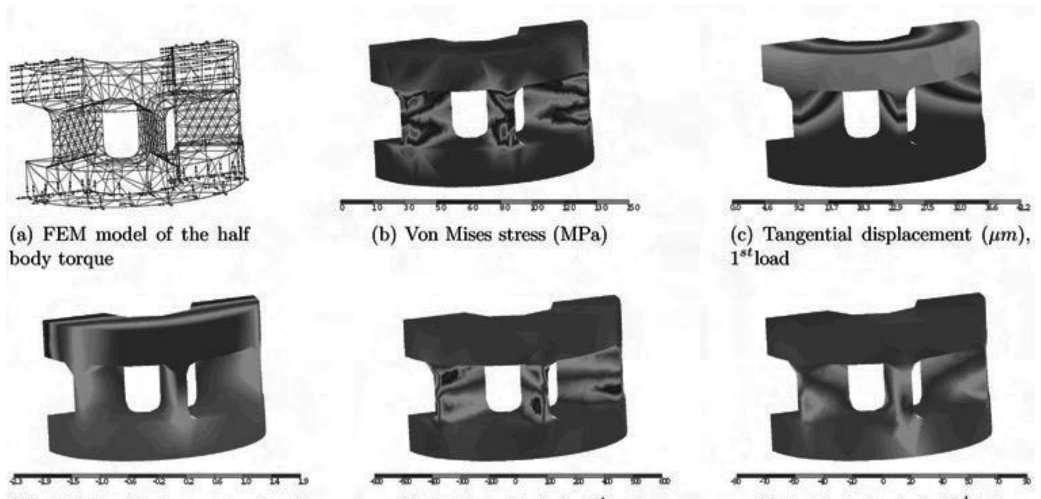

(a) FEM model of the half body torque (b) Von Mises stress (MPa) (c) Tangential displacement (μm), 1^{st}load (d) Axial displacement (μm), (e) Axial strain ($\mu\epsilon$), 1^{st}load (f) Axial strain ($\mu\epsilon$), 2^{nd}load

Fig. 3. FEM analysis.

Moreover, as mentioned earlier, maximum torsional sensitivity requires minimum bending sensitivity. In particular, the radial displacement of the disc's outer diameter due to the torsion, principal strain at the strain gauge seats due to both load cases, and maximum von Mises stresses/strains due to a combination of all load cases are selected as the design benchmarks. The described design criteria can be checked with the FEM results to modify the geometry of the sensor iteratively.

In the shape optimization process we chose as design variables the wing thickness, the distance between two disks, and the inner hole diameter. The 95 mm outer diameter was selected to match our particular motor. The selected dimensions were varied to maximize the structure effciency (4) subject to keeping the maximum von Mises stresses σ within the allowable limits σ_{allow}, considering fatigue. That is

$$\max_{\text{design varibles}} \eta$$
$$\text{subject to} \quad \sigma < \sigma_{\text{allow}}$$

The IDEAS solver uses the Steepest Decent (the gradient method) method with penalty function for finding the local minimum - the penalty function adds simply the weighted constraint equation into the objective function.

The post-processing stage was guided by the following design benchmarks: the tangential and axial displacement of the disc's outer diameter; the principal strain in the axial direction and parallel to the gauge axes, due to both load cases, and the maximum von Mises stress/strain due to a combination of all load cases. Hence, the performance index can be obtained from Figures 3(c) and 3(e), while the the constraint condition is given by Fig. 3(b). These design criteria were checked with the FEM results to modify the geometry of the sensor iteratively. The FEM results of the elastic body's final design are shown on Fig. 3. The worst-case von Mises stress, i.e. the combination of the two load cases, is shown in Fig. 3(b) where its maximum occurs at 150 MPa. This is close to the endurance limit of mild steel with a reasonable factor of safety. Figs 3(c) and 3(d) illustrate the tangential and axial displacement fields by which the torsional and bending stiffnesses are carried out; $k_z = 2.7 \times 10^5$Nm/rad and $k_x = 4.9 \times 10^6$Nm/rad, respectively. The axisymmetric pattern in the figure confirms the correctness of the imposed boundary conditions. Figs. 3(e) and 3(f) show the strain contour in the axial direction in which the strain gauges are oriented, for the 1st and 2nd load cases, respectively. The FEM results demonstrate that the strain sensitivity in torsion is seven times higher than in bending, while the bending stiffness is 18 times higher than the torsional stiffness

4.2 Experimental Characterization

The torque sensor was machined from a solid steel rod (Fig. 4). Foil strain gauges (SG-3/350-LY41 from Omega (Omega, 1995)) were cemented at the locations determined by FEM. The strain gauge bridge is excited by a precisely regulated 8.0 V DC voltage. Instrumentation amplifiers built into the sensor boost the signal level of the Wheatstone bridge output before A/D conversion. We took advantage of the hollow motor shaft, which is common in direct-drive motors, to locate the electronic circuit board beside the sensor. The local signal conditioning provides a stronger output signal and improves the S/Nratio. Moreover, since the electronic circuit is totally enclosed by the motor's hollow shaft, it is well shielded from the powerful magnetic noise created by the motor.

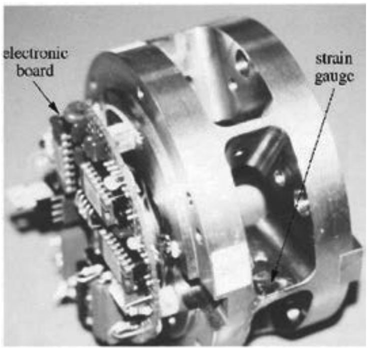


Fig. 4. The torque sensor prototype.

4.2.1 Static Test

In order to characterize the linearity and sensitivity of the sensor, static torsional and bending torques were applied in the experimental apparatus illustrated in Fig. 5. One side of the sensor is affixed to a bracket, while two aluminum bars were attached radially and axially to the other side. The ends of the bar were connected to a mechanical lever via ropes in which load cells (MLP-50 from Transducer Techniques (Techniques, n.d.)) were installed. The lever varied the tension in the cord gradually between zero and maximum. During loading and unloading, the reference load cell output and the torque sensor output (device under test) were recorded.

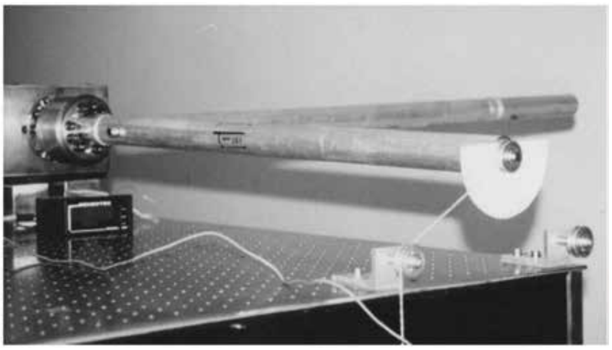
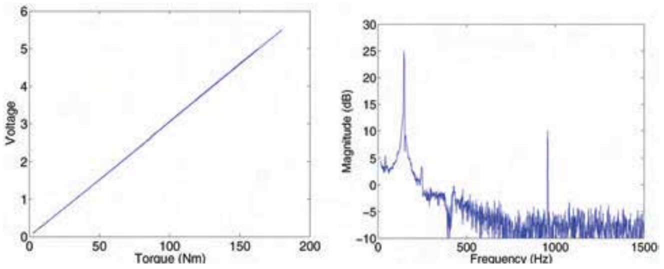


Fig. 5. The static test setup.



(a) Sensor output versus applied torque (b) Dynamic response

Fig. 6. Characteristics of the torque sensor.

The force transducer signal is scaled to torque and then is plotted versus the torque sensor output voltage in Fig. 6(a) for 2, 000 sample points. The slope of the line indicates the sensor calibration coefficient of α =30mV/Nm.The experimental data shows that all collective deviations from linearity are less than 0.2 % full scale.

Low sensitivity to the other axes is one of the key characteristic of a good joint torque sensor. The cross sensitivity measurements were performed by utilizing the static tesbed setup. Forces and moments are applied on different axes by the system of pullies and weights shown in Fig. 5. The bending moment is applied via an axial bar firmly connected to the sensor, while the torsion torque is applied by the radial arm. The direction of the force is set by a pulley, as shown in Fig. 5. Theoretically, the sensor output should not be responsive to the bending moment or the forces at all. However, in practice, due to inaccurate placement of the strain gauges and/or differences in the Gauge Factors of the strain gauges, exact decoupling may not be achieved. In the course of the experiment, it becomes evident that, with the exception of the torsion moment, the bending moment dominates the sensitivity of the sensor. The experimental result indicates that the ratio of the sensor readings with respect to the bending and torsion -the cross sensitivity -is only 0.6%. This confirms that the sensor effectively decouples the effect of the non-torsional components on the measured torque signal.

4.2.2 Dynamic Test

Dynamic testing serves mainly to validate the FEM results on which the stress analysis is based. The experiment is arranged to extract the stiffness of the sensor prototype. Again, the sensor is held rigidly by a bracket while a steel disk is flanged to the other side. The disk is massive, with an inertia of $I_{zz} = 0.24$ kgm^2, and the whole system behaves like a second order system. To detect all the vibration modes corresponding to all compliance directions, the cross sensitivity is deliberately increased by electrically by-passing the strain of all strain gauge pairs except one. Therefore, the torque sensor no longer has the decoupling property, and its output is the summation of all torque/force components weighted by their corresponding gains. The system is excited impulsively by a hammer, and a data acquisition system records the subsequent vibration with a sampling rate of 3.2 kHz. Since the torsion gain is highest, the sensor signal is dominated by torsion vibration which decays due to the structural damping of the sensor material. Nevertheless, the sensor's modes are shown clearly in the frequency domain. To this end, a 0.5 second interval of the signal is taken via a Hamming window and then its spectrum is found by FFT. Fig. 6(b) reveals the modal frequencies associated with the bending and torsion compliances occurred at 150 Hz and 980 Hz. Due to the low damping, the modal frequencies are almost the same as the natural frequencies. The corresponding torsion stiffness is calculated to be $k_z = 2.4 \times 10^5$Nm/rad, which results in a high torsional stiffness. The bending stiffness can be found in the same fashion. However, it should be noted that the relative inertia is half of the inertia for disks, i.e. $I_{xx} = 0.5 I_{zz}$. The bending stiffness is calculated to be twenty times higher than the torsion stiffness, $k_x = 4.8 \times 10^6$ Nm/rad. A comparison with the FEM predictions reveals an acceptable 20% error.

5. Joint Torque Sensory Feedback

5.1 Motivation

Model-based robot control schemes have been proposed in the past for achieving precise motion tracking, e.g., resolved acceleration control (Luh *et al.*, 1980), or the computed torque method (An *et al.*, 1988). These approaches depend on dynamics modeling of the manipulator's load and link dynamics; they potentially perform poorly when the model is not accurate (An *et al.*, 1988). Adaptive control of manipulators was proposed to deal with parametric uncertainties of the manipulator model (Slotine and Li, 1987). However, these controllers cannot deal with robotic systems with a unknown or variable payloads unless a more complicated adaptive versions is used (Yong-Duan *et al.*, 1989). Robust control (Slotine, 1985) and robust adaptive control of manipulators (Reed and Ionnou, 1989) have been developed that can maintain the stability with respect to uncertainties, including bounded disturbance, time-varying parameters, as well as unmodeled dynamics. However, the performance may be compromised in these control approaches.

Alternatively, the dynamic control of manipulators can be performed by using joint-torque sensory feedback without the need for modelling link dynamics (Hashimoto, 1989b; Kosuge *et al.*, 1990;Stokic and Vukobratovic, 1993; Aghili *et al.*, 2001; Aghili and Namvar, 2006). Such a controller can achieve robust performance against variation of link dynamics and rejection of external force disturbances at the expense of using additional sensors. Kosuge *et al.* (Kosuge *et al.*, 1990) demonstrated experimentally the effectiveness of using joint-torque measurements to compensate for the nonlinear link dynamics of a

SCARA-type direct-drive robot. Hashimoto (Hashimoto, 1989b) applied this technique to a harmonic-drive actuator where the deformation of flex-splines are used to measure joint-torques.
A survey of JTF can be found in (Stokic and Vukobratovic, 1993). In summary, the main advantages claimed for JTF are:

(i) Joint sensory feedback can simplify the complexity of a manipulator system dynamics by eliminating the link dynamics;
(ii) The control system is inherently robust w.r.t. parameter variation, e.g., a manipulator grasping a payload with mass variation; and
(iii) The control system can completely reject external force disturbances applied on the links.

It should be pointed out that essentially the joint torque measurement is possible if there is a small, but nonzero, joint deformation. A complete model of elastic-joint manipulators, which accounted for the dynamics of both the rotors of the drive system and the link system, were developed in (Murphy *et al.*, 1990). These models were used by many researchers for development and analysis of non-adaptive (Spong, 1987; Tomei, 1991) and adaptive (Lozano and Brogliato, 1992) controllers for flexible joint manipulators. These controllers aimed at achieving high performance by taking finite stiffness of the joints into consideration; however, this complicates the equations of motion by increasing the order of the system. Tomei (Tomei, 1991) showed that even a simple PD controller, similar to that used for rigid manipulators, suffices to globally stabilize the elastic-joint manipulators about a reference point. The dynamics model developed in (Kosuge *et al.*, 1990) for manipulators with JTF is a special case of the flexible joint model where the stiffness approaches infinity. Unlike the other control approaches, the JTF controller (Kosuge *et al.*, 1990) requires only the rotor dynamics, i.e., a half-model. However, since the joint deformation is not compensated for by the controller, a sufficiently high joint stiffness is required so that the overall joint deformation can be assumed negligible.

5.2 Robot Dynamics with JTF

In robotics application, joint torque sensing can be used to eliminate completely the effect of link dynamics if the manipulator's joint are endowed with joint torque sensor device. The system dynamics are determined completely by the gyroscopic motion of the rotors of the joint actuators.

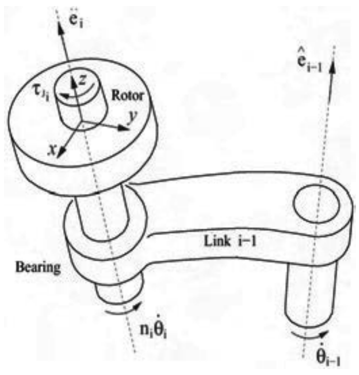


Fig. 7. The *i*th joint.

Fig. 7 depicts the *i*th motor axis and joint axis of a robot with n revolute-joints where each joint is driven by a geared motor with gear ratio n_i and assuming that the motor shaft is cut right at its junction to the link. In the following the subscript *i* denotes quantities related to the *i*th link or joint.
Assume that ω_{zi} be the z-component of the absolute angular velocity of the rotor ω_i; θ_i and $\bar{J}_i$ denote the rotor angle and the rotor inertia along the z-axis. Also, τ_{J_i}, τ_{M_i}, and $b_i(\dot{\theta})$denote the torques of the motor, the joint, and the bearing friction, respectively, all of them seen after the gearbox. Let us assume the followings: (i) The principal axes of all rotor inertias are parallel to their own joint axes; (ii) all rotors are statically balanced, i.e., their center of mass coincident with rotation axis; and (iii) the rotors are axisymmetric, i.e., their inertias about x and y axes are identical. Also as in (Kosuge *et al.*, 1990), we assume that the deformation of the joint torque sensors are negligible. Moreover, let coordinate system $\{x_i, y_i, z_i\}$be attached to *i*th joint according to the Denavit-Hartenberg convention. Since the rotors are axisymmetric, the equation of motion of each rotor can be simply described by

$$J_i\dot{\omega}_{z_i} = \frac{1}{n_i}\left(\tau_{M_i} - b_i(\dot{\theta}_i) - \tau_{J_i}\right). \tag{5}$$

In the above equation, we need to obtain an expression for the absolute angular velocity in terms of joint angle quantities. Let $\hat{e}_j \in \mathbb{R}^3$represent a unit vector in the direction of the *j*th joint axis. Then, since the deformations of the torque sensors are assumed negligible, one can compute the link velocities by a technique similar to Luh *et al.* (Luh *et al.*, 1980)

$$\omega_{i-1} = \sum_{j=1}^{i-1} \hat{e}_j\dot{\theta}_j, \tag{6}$$

$$\omega_{z_i} = \hat{e}_i^T\omega_{i-1} + n_i\dot{\theta}_i. \tag{7}$$

Equations (6) and (7) can be combined in the vectorial form as

$$\omega_z = (D(\theta) + N)\dot{\theta}, \tag{8}$$

where N =diag$\{n_i\}$, and $D(\theta) \in \mathbb{R}^{n\times n}$ is a lower triangular matrix whose elements are

$$d_{ij} = \begin{cases} \hat{e}_i^T\hat{e}_j & \text{if} \quad i > j \\ 0 & \text{otherwise} \end{cases}$$

It is worth pointing out that the unit vectors $\hat{e}_j$'s constitute the columns of the manipulator's rotation Jacobian (Sciavicoo and Siciliano, 2000), i.e. $A_\omega = \partial\omega_n/\partial q = [\hat{e}_1 \quad \hat{e}_2 \quad \cdots \quad \hat{e}_n]$.Hence, the *D* matric can be constructed from the Jacobian by

$$D = \text{tri}(A_\omega^T A_\omega),$$

where matrix function tri(.) returns only the lower-triangular elements of the input matrix, while the rest of elements of the returned matrix are zero. Defining $J \triangleq \text{diag}\{n_i^2\bar{J}_i\}$and $T \triangleq I + N^{-1}D$, and using (8) in (5), we get

$$JT\ddot{\theta} + J\dot{T}(\theta,\dot{\theta})\dot{\theta} + b(\dot{\theta}) = \tau_{\text{net}}, \tag{9}$$

where $\tau_{\text{net}} = \tau_M - \tau_J$ is the net torque acting on the rotors.
Example. 1. *The coupling matrix D of a general 3-DOF robot is*

$$D(\theta) = \begin{bmatrix} 0 & 0 & 0 \\ ca_1 & 0 & 0 \\ \begin{matrix} ca_1ca_2 \\ -sa_1sa_2c\theta_2 \end{matrix} & ca_2 & 0 \end{bmatrix},$$

where a_1 and a_2 are the twist angles, and sa_i and ca_i represent $\sin(a_i)$ and $\cos(a_i)$.

Now, one can readily show that the following inverse-dynamics control scheme

$$\tau_{M_d} = \tau_J + u, \qquad (10)$$
$$u = JT(\theta)\left(\ddot{\theta}_d + G_D(\dot{\theta}_d - \dot{\theta}) + G_P(\theta_d - \theta)\right) + J\dot{T}\dot{\theta} + b,$$

where $G_D > 0$ and $G_P > 0$ are the feedback gains, can achieve dynamic tracking of the desired trajectory θ_d.

6. Adaptive Joint Torque Feedback with Uncalibrated Torque Sensor

In this section, we present development of an adaptive JTF controller that takes uncalibrated joint-torque signals, and then identifies the sensor calibration parameters as well as all joint parameters including rotor inertia, the link twist angles, and joint-friction parameters (Aghili and Namvar, 2006). We show that asymptotic stability of the proposed adaptive controller can be ensured if an upper bound on the estimated sensor-gain is respected. Subsequently, the parameter adaptation law is modified according to the projection parameter adaptation to meet the stability condition.

6.1 Adaptive Control

Note that dynamics formulation (9) is not adequate for an adaptive control because $J\,T$ is not a symmetric matrix. This problem can be fixed by changing the torque coordinate. Pre-multiplying (9) by $T(\theta)^T$ leads to

$$M_R\ddot{\theta} + h_R(\theta, \dot{\theta}) + b(\dot{\theta}) = T^T(\tau_M - \tau_J). \qquad (11)$$

where $M_R = T^T JT$ and $h_R = C_R\dot{\theta} = T^T J\dot{T}\dot{\theta}$.

Remark. 1. *Since J is positive-definite and T is a nonsingular matrix, the symmetric inertia matrix M_R is positive-definite. Also, it can be readily verified that matrix $\frac{d}{dt}M_R - 2C_R$ is skew-symmetric.*

We assume that all joints are equipped with torque sensing devices and that vector $\tau_S \in \mathbb{R}^n$ represents the sensor outputs. Recall from (3) that the output signals are linearly related to the actual joint-torques. Hence, we can say

$$\tau_J = \alpha\tau_S + \beta, \qquad (12)$$

where $\alpha = \mathrm{diag}\{a_i\}$ and $\beta^T = [\beta_1, \, , \beta n]$ are sensor gains and offsets, respectively. Substituting (12) in (11)gives

$$M_R\ddot{\theta} + h_R(\theta, \dot{\theta}) + T^T(b(\dot{\theta}) + \alpha\tau_S + \beta) = T^T\tau_M, \qquad (13)$$

where the friction $b(\dot{\theta})$ can be captured by a friction model reported in the literature (de Wit et al., 1995).

The control objective here is to find a control law τ_M such that the rotor positions θ tracks the desired trajectory $\theta_d(t)$, while parameters J, D, a, β and friction coefficients are considered unknown. Moreover, note that the manipulators' twist angles are the only relevant parameters to matrix D. Hence, $T(\theta, \rho t)$ can be parameterized in terms of the relevant kinematics parameters ρ_t –note that the kinematics parameters appear also independently in ρ_s in combination with dynamics and calibration parameters. Assume that the dynamics equation (13) is rewritten in a linearly parameterized form as

$$M_R\ddot{\theta} + C_R\dot{\theta} + T^T(b(\dot{\theta}) + \alpha\tau_S + \beta) \triangleq Y_s(\ddot{\theta}, \dot{\theta}, \theta, \tau_S)\rho_s, \qquad (14)$$

where vector ρ_s contains combined dynamics, kinematics and calibration parameters. Now, denoting $\hat{T}$, $\hat{\rho}_t$, $\hat{J}$, $\hat{C}$, $\hat{\alpha}$, $\hat{\beta}$, and $\hat{b}$ as the estimated variables, we propose the following control law

$$\tau_M = \hat{T}(\hat{\rho}_t)^{-T}(Y_s(\dot{v}, v, \theta, \tau_s)\hat{\rho}_s - Ls) \tag{15}$$

where

$$\begin{aligned} v &\triangleq \dot{\theta}_d - \Lambda(\theta - \theta_d), \\ s &\triangleq \dot{\theta} - v, \end{aligned} \tag{16}$$

and L > 0 and Λ > 0 are the gains. The matrix inversion in (15) should not be a problem, because det $(\hat{T}) = 1$meaning that $\hat{T}^{-T}$is always nonsingular regardless of the value of $\hat{\rho}_t$.
Consider the parameter update law

$$\dot{\hat{\rho}} = -\Gamma Y^T s, \tag{17}$$

where $Y(\dot{v}, v, \theta, \tau_S, \tau_M) = [\ Y_s \quad -Y_t\ ]$, $\rho^T = [\ \rho_s^T \quad \rho_t^T\ ]$,and Γ > 0 is the gain matrix. In the following, the stability of the control error $\tilde{\theta} = \theta_d - \theta$ will be investigated.

Therorem. 1. The control law (15) together with the parameter update law (17) ensure the following properties: $\tilde{\theta}, s \in \mathcal{L}_2 \cap \mathcal{L}_\infty$, and $\tilde{\rho} \in \mathcal{L}_\infty$where $\mathcal{L}_\infty$ and $\mathcal{L}_2$ stand for the space of bounded and square integrable signals, respectively.

Proof: Substituting the control signal (15) in the system dynamics (13) yields the error dynamics described by

$$\begin{aligned} M_R\dot{s} + C_R s + Ls &= Y_s(\dot{v}, v, \theta, \tau_S)\tilde{\rho}_s - \tilde{T}^T\tau_M \\ &= Y_s(\dot{v}, v, \theta, \tau_S)\tilde{\rho}_s - Y_t(\theta, \tau_M)\tilde{\rho}_t \\ &= Y(\dot{v}, v, \theta, \tau_S, \tau_M)\tilde{\rho}, \end{aligned} \tag{18}$$

Here we have used the property that for any dimensionally compatible vector *x*,

$$T(\theta, \rho_t)x = Y_t(\theta, x)\rho_t,$$

Introducing the positive storage function

$$V = \frac{1}{2}s^T M_R s + \frac{1}{2}\tilde{\rho}^T\Gamma^{-1}\tilde{\rho}$$

and differentiating it along with trajectories of the error dynamics and using Remark 1 yields

$$\dot{V} = -L\|s\|^2 \leq 0. \tag{19}$$

From this, a standard argument (Ioannou and Sun, 1996) proves the theorem.

6.2 Asymptotic Stability

Asymptotic stability of control error $\tilde{\theta}$,or s, can be ensured provided that s is uniformly continuous, i.e., $\dot{s}$ is bounded. The latter can be inferred only if control input τ_M is bounded. The condition for boundedness of τ_M is obtained in the following. For simplicity in presentation we assume that kinematics parameter vector ρ_t is known and then we can derive a simple condition for boundedness of τ_M. Extending results to the case where ρ_t is unknown is straight forward.

Following the Lagrangian approach in (Luca and Lucibello, 1998), the joint torque can be computed from

$$M_L'\ddot{\theta} + h_L(\theta, \dot{\theta}) + d(t) = \tau_J, \tag{20}$$

'where $M_L' = M_L + D^T N^{-1} J$, $M_L \in \mathbb{R}^{n\times n}$ is link inertia matrix, $h_L \in \mathbb{R}^n$ contains all the nonlinear terms, andvector $d(t) \in \mathbb{R}^n$captures the effect of allexternal force disturbances

Eliminating the joint accelerations from equations (13) and (20) and solving the resultant equation in terms of τ_S yields

$$\tau_S = \alpha^{-1} R \tau_M + g_1 - \alpha^{-1}\beta, \tag{21}$$

where $R \triangleq M'_L(M'_L + JT)^{-1}$, and $g_1 = (I - R)(h_L + d) - R(h_R + b)$. Substituting τ_S from (12)into (15) gives an expression for the control input independent of τ_S. That is

$$\tau_M = (I - \breve{\alpha}R)^{-1} g_2(\dot{v}, v, \dot{\theta}, \theta, \hat{\rho}), \tag{22}$$

where $g_2 = T^{-T}(\hat{M}_R\dot{v} + \hat{C}_R v - Ls) + \hat{\tau}_F + \hat{\alpha}g_1 + \hat{\beta} - \breve{\alpha}\beta$, and

$$\breve{\alpha} \triangleq \operatorname{diag}\{\hat{\alpha}_i/\alpha_i\}$$

'is the normalized estimation of torque sensor gain. Since all variables of function $g_2(\dot{v}, v, \dot{\theta}, \theta, \hat{\rho})$ are bounded, then $\|g_2\|$is bounded too. Hence, we can say τ_M is bounded if and only if $I - \breve{\alpha}R$ is invertible, i.e.

$$\begin{aligned} \tau_M \in \mathcal{L}_\infty \;&\Leftrightarrow\; \lambda_i(I - \breve{\alpha}R) \neq 0 \quad \forall i \\ &\Leftrightarrow\; \lambda_i(\breve{\alpha}R) \neq 1 \quad\quad \forall i \\ &\Leftrightarrow\; \{\theta, \breve{\alpha}\} \notin \mathcal{S}(\theta, \breve{\alpha}), \end{aligned}$$

where

$$\mathcal{S} = \bigcup_{i=1}^{n} \{\theta, \breve{\alpha} | \lambda_i(\breve{\alpha}R(\theta)) = 1\}. \tag{23}$$

Therefore, the control input τ_M remains bounded and the control error $\tilde{\theta}$asymptotically converges to zero if this part of the system states $\{\theta, \breve{\alpha}\}$α} does not enter the region $\mathcal{S}$. In this case, $\dot{s}$ is bounded and hence s converges asymptotically to zero, which, in turn, implies convergence of $\tilde{\theta}$ to zero.

In the following, we derive a sufficient condition on the gain sensor estimate to achieve a bounded control input.

Remark 2 *Using norm properties, we can obtain a conservative condition satisfying (23) as follows*

$$\begin{aligned} \tau_M \in \mathcal{L}_\infty \;&\Leftarrow\; \bar{\sigma}(\breve{\alpha}R) < 1 \\ &\Leftrightarrow\; \max_i |\breve{\alpha}_i| < 1/\bar{\sigma}(\mathcal{Q}). \end{aligned} \tag{24}$$

Therefore, joint torques remain bounded providedthat an over-estimation ofthe sensor-gains does not exceed $1/\bar{\sigma}(R)$. Let us choose constant $\bar{\alpha}_i > 0$ so that $\bar{\alpha}_i \geq |\alpha_i|/\bar{\sigma}(R)$ always holds. Then, by virtue of Remark 2, one can show (24) is satisfied if

$$|\hat{\alpha}_i| \leq \bar{\alpha}_i. \tag{25}$$

Based on the projection adaptation law (Ioannou and Sun, 1996), it is possible to modify the parameter update law (17) to ensure that inequality (25) holds. Assume that α_i and ψ_i be the ith elements of $\hat{\rho}$ and $-\Gamma Y^T$ s, respectively. Then, consider the following projection parameter adaptation law for ˆ

$$\dot{\hat{\alpha}}_i = \begin{cases} \psi_i & \text{if} \quad |\hat{\alpha}_i| < \bar{\alpha}_i \text{ or} \\ & \quad\quad |\hat{\alpha}_i| = \bar{\alpha}_i \text{ and } \operatorname{sgn}(\bar{\alpha}_i\psi_i) < 0 \\ 0 & \text{otherwise} \end{cases}$$

A standard argument (Ioannou and Sun, 1996) shows that the modified parameter update law guarantees (25), while keeping (19) negative, i.e. $\dot{V} \leq -L\|s\|$

7. Joint Torque Feedback in the Presence of Actuator Dynamics

Ideally, positive joint torque feedback decouples the load dynamics exactly and provides infinite stiffness for external torque disturbance. This can be possible if an ideal actuator reproduces the same torque as measured by the torque sensor and hence the load torque can be trivially compensated. However, in practice, due to actuator's finite bandwidth dynamics, the feedback system may not respond fast enough to the measured load-torque. As a result, a complete compensation of disturbance torque cannot be achieved in the presence of actuator dynamics and/or delay. This section presents an optimal filter for positive joint torque feedback control which takes the actuator's dynamics into account and minimizes the servo-controller sensitivity to load torque disturbance (Aghili *et al.*, 2001). We show that finding optimal torque feedback is equivalent to the model-matching problem that has an exact solution (?). In the case of dynamic load, optimal torque feedback minimizes the effect of load perturbation on the nominal rotor dynamics. A single variable case is considered herein, yet the analytic solution can be readily extended for a multi-variable case.

7.1 Optimal JTF Design

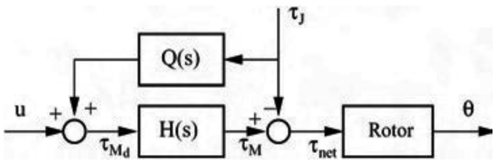


Fig. 8. Joint torque feedback through filter Q(s).

Fig.8 illustrates the general block diagram of a JTF system, where transfer function H(s) represents the actuator dynamics and Q(s) is a filter. The external disturbance τ_J is measured via a torque sensor and the torque signal is fed back for compensation through the filter Q(s). Let u(s)be the compensated control input under positive joint torque feedback in the Laplace domain. Similar to (10), we can define the compensated control input u as

$$\tau_{M_d}(s) = Q(s)\tau_J(s) + u(s), \tag{26}$$

where s is the Laplace variable. The disturbance sensitivity function

$$\chi(s) \triangleq \frac{\tau_{\text{net}}(s)}{\tau_J(s)} = -1 + H(s)Q(s). \tag{27}$$

shows how the disturbance τJ(s)is transmitted to the net torque τnet(s) acting on the rotor. Note that for ideal actuator, where *H*(s) = 1, we can achieve complete decoupling of the disturbance, i.e., χ(s)=0, by trivially choosing *Q*(s) =1.
Also, let ***G***(s) represent the compliance of a position feedback controller, i.e.

$$G(s) = \left.\frac{\partial\theta(s)}{\partial\tau_J(s)}\right|_{Q=0}.$$

Then, one can show that addition of the JTF loop changes the overall compliance of the motion servo to

$$G_{\text{JTF}}(s) = \chi(s)G(s),$$

which is equivalent to the weighted disturbance function (Aghili *et al.,* 2001), if the weighting function is chosen as $W_1(s) = |G(s)|/\|G\|_\infty$.Now, the problem is to finda stable and realizable filter $Q(s) \in \mathcal{RH}$ (the class of H_∞ functions which are rational) such that the maximum weighted sensitivity of the system is minimized, that is

$$\inf_{Q(s)\in\mathcal{RH}} \left\| \begin{bmatrix} W_1(1-HQ) \\ W_2Q \end{bmatrix} \right\|_\infty . \tag{28}$$

Note that the first weighting function, $W_1(s)$, shapes the disturbance gain over frequency band of interest, while the second weighting function, $W_2(s)$, shapes the magnitude of the optimal filter Q(s). Note that $W_2(s)$ causes the magnitude of the torque filter Q(s) is rolled off at high frequency where the magnitude of $W_1(s)$ is sufficiently low. Problem (28) is a standard H_∞ problem and the optimal solution is available.

7.2 Experimental Results

This section evaluates the performance of the proposed JTF experimentally in terms of torque disturbance attenuation and position tracking accuracy under varying payload. In order to measure the torque disturbance sensitivity, torque disturbances are directly injected into the a joint-servo system by using a hydraulic dynamometer. Also, an arm with adjustable payload is mounted on the motor's shaft to investigate the load decoupling and robust stability properties.

7.3 Setup

Fig. 9 illustrates the experimental setup which consists of the McGill/MIT Direct-Drive electric motor (Aghili *et al.*, 2002b) mounted on the dynamometer, instrumented with our custom torque sensor described in Section 4, and coupled to a hydraulic rack and pinion rotary motor (Parker 113A129BME). The role of the hydraulic motor is to generate a random disturbance torques in order to measure the disturbance sensitivity of the servo controller.

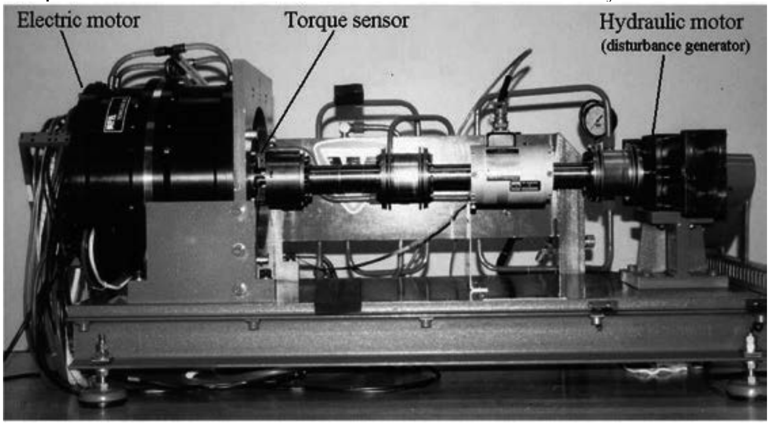


Fig. 9. Direct-drive motor on the hydraulic dynamometer testbed.

7.4 Identification

The performance of joint torque feedback critically relies on the knowledge of actuator dynamics, that, in turn, can be obtained form disturbance sensitivity tests. Let us consider again the control system shown in Fig. 8.The challenge in identification system *H*(s) is that signal τ_M is not me asurable. How-ever, the actuator transfer function can be extracted from measurements of the disturbance sensitivity functions with respect to two different torque feedbacks. Denoting G_{JTF} as the sensitivity function corresponding to $Q \equiv 1$, one can obtain $G'_{\rm JTF}(s) = (H(s)-1)G(s)$ from (27). Let us assume that $\Phi_{\tau_J\tau_J}$ and $\Phi'_{\theta\tau_J}$ denote the corresponding spectral τ_J densities when $Q = 1.$. Then, one can obtain the empirical (non-parametric) transfer function $\hat{H}(\omega)$ from the latest equation as

$$\hat{H}(j\omega) = 1 + \frac{\Phi_{\theta\tau_J}(j\omega)\Phi'_{\tau_J\tau_J}(j\omega)}{\Phi'_{\theta\tau_I}(j\omega)\Phi_{\tau_J\tau_J}(j\omega)}. \tag{29}$$

The next step of the identification involves a numerical procedure to represent the complex function (29) by a rational transfer function as close as possible .Several parametric models were examined, and it turned out that a second order systems is sufficient to match the input-output behavior adequately. The parametric models approximating *G* (jω) and *H* (jω), which are used for the feedback design, are

$$\hat{G}(j\omega) \approx \frac{-0.002s + 0.226}{s - 9.523}, \quad \text{and} \quad \hat{H}(j\omega) \approx \frac{-5.014s + 578}{s^2 - 35.59s + 604}.$$

Note that $\hat{H}(j0) = 0.96$ represents the DC gain of the actuator.

7.5 Disturbance Rejection Tests

We compare the disturbance attenuation of our servo controller with and without using JTF. To this end, we command a ramp reference signal through the PID position controller while the hydraulic motor injects random torque disturbances. Fig. 10 illustrates the position tracking error trajectories due to the random torque disturbances without and with the torque feedback is applied. The control system exhibits relatively high disturbance sensitivity when there is no torque feedback. The figure clearly shows that the tracking error is substantially reduced when the torque feedback is applied. The disturbance attenuation is better explained in Fig. 11 in the frequency domain. As expected, at a sufficiently high frequency, the disturbance sensitivity drops due to the attenuation effect of position feedback. The torque feedback lowers this system sensitivity remarkably over the whole frequency range.

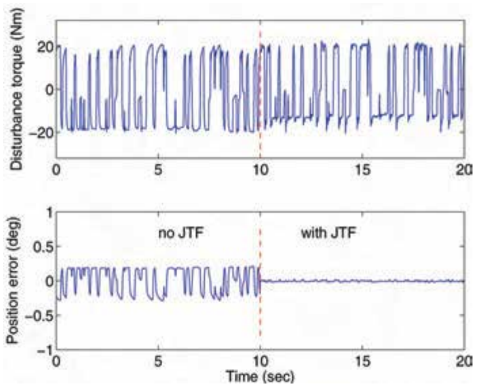


Fig. 10. Tracking error with and without JTF is applied.

due to the random torque disturbances without and with the torque feedback is applied. The control system exhibits relatively high disturbance sensitivity when there is no torque feedback. The figure clearly shows that the tracking error is substantially reduced when the torque feedback is applied. The disturbance attenuation is better explained in Fig. 11 in the frequency domain. As expected, at a suciently high frequency, the disturbance sensitivity drops due to the attenuation effect of position feedback. The torque feedback lowers this system sensitivity remarkably over the whole frequency range.

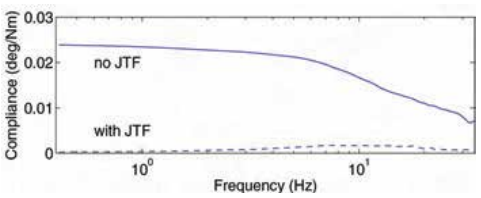


Fig. 11. Disturbance attenuation with and without JTF is applied.

The next objective is to demonstrate experimentally the performance of the servo controller with JTF under a dynamical load. To this end, a link with a 7.2 kg mass is mounted on the motor's torque sensor that plays the role of an uncertain payload. The counterbalance weight produces a nonlinear gravity torque to be compensated with positive joint torque feedback. To investigate the tracking performance of the control system, we commanded a sinusoidal reference position trajectory θ_d (t) = π/4cos (πt/3) rad to the motion controller. First, no JTF is applied. Since the nonlinear link dynamics (due to the gravitational term) are not compensated by a JTF controller, the tracking error resulting from the PID controller alone is large, as shown in Fig. 12. Yet, when the joint torque feedback is applied the tracking error is reduced significantly.

Sensor Type	Sensitivity mV/Nm	Torsional Stiffness 10^4 Nm/rad	Bending Stiffness 10^4 Nm/rad	η 10^3 V/rad
B	45.7	3.4	N/A	1.55
C	96.5	1.5	N/A	1.45
E	30	24	480	7.2

Table 1. A comparison with various type of torque sensors.

8. Conclusion

Motivated by the need for accurate joint torque sensing in robots, we designed a new torque sensor, based on the hollow hexaform geometry. Its key features were its extremely high stiffness and its insensitivity to the set of support forces and moments which persist in a robot joint. These features permit to mount the sensor directly in the joints of a robot manipulator leading to accurate joint torque sensing and to a compact and modular design. The structure of the sensor also exhibits strain concentration to torsion loads which maximizes the sensitivity to torsion without sacrificing torsional stiffness. Other design issues such as practical shape consideration, material properties and overloading also considered. The sensor geometry was analyzed and optimized using the finite element method. The sensor was tested extensively to confirm its design goals, and is well suited as a torque-sensing device in robots or other industrial high performance motion control applications. A quantitative comparison with different types of sensors is shown in table 1. The table indicates that our sensor's performance characteristics compare very favorably.

The applications of adaptive control in conjunction with joint-torque sensory feedback was used for dynamic motion control of manipulators. The control system had the advantages of requiring neither the computation of link dynamics nor the precise measurement of joint torques, i.e., the torque sensor's gains and offsets are unknown to the controller. The adaptive controller could also tune all the joint parameters including the rotor inertia, twist

angles of joint axes, and joint friction parameters. The stability of the control system was investigated analytically. It was shown that the control error asymptotically converges to zero if an upper bound on the estimated sensor gain is respected. Subsequently, the parameter update law was modified based on the projection algorithm to satisfy the boundedness condition.
Next, we formulated the problem of optimal positive JTF in the presence of actuator's finite bandwidth dynamics. The theory of JTF was further developed to suppress the effect of load torque disturbance on a motion control systems in the presence of actuator dynamics. An experimental setup comprised of a direct-drive electric motor, torque-sensor, and hydraulic motor was constructed to measure disturbance sensitivity of a motion servo mechanism. The results demonstrated that when the servo controller was cascaded with the optimal JTF, a significant reduction in sensitivity was achieved. In our second experiment, a single link with adjustable inertia was attached to the motor. In the absence of any torque feedback, the tracking error increased due to load nonlinearity, and increases with payload, while the optimal feedback maintains the tracking accuracy.

9. References

Aghili, F. and M. Namvar (2006). Adaptive control of manipulator using uncalibrated joint-torque sensing. IEEE Trans. Robotics 22(4), 854-860.

Aghili, F., M. Buehler and J. M. Hollerbach (2002a). Design of a hollow hexaform torque sensor for robot joints. *Int. Journal of Robotics Research* 20(12), 967 –976.

Aghili, F., M. Buehler and J. M. Hollerbach (2002b). Development of a high performance joint. *Int. Journal of Advanced Robotics* 16(3), 233–250.

Aghili, Farhad, Martin Buehler and John M. Hollerbach (2001). Motion control systems with h-infinity positive joint torque feedback. *IEEE Trans. Control Systems Technology* 9(5), 685–695.

An, C. H., C. G. Atkeson and J. M. Hollerbach (1988). *Model-Based Control of a Robot Manipulator*. MIT Press. Cambridge, MA.

Asada, H. and S.-K. Lim (1985). Design of joint torque sensors and torque feedback control for direct-drive arms. In: *ASME Winter Annual Meeting: Robotics and Manufacturing Automation*, PED-Vol. 15. Miami Beach. pp. 277–284.

de Wit, C. Canudas, H. Olsson, K. J. Aström and P. Lischinsky (1995). A new model for control of systems with friction. *IEEE Transactions on Automatic Control* 40(3), 419–425.

deSilva, C. W., T. E. Price and T. Kanade (1987). Torque sensor for direct-drive manipulator. *ASME Journal of Engineering for Industry* 109, 122–127.

Hashimoto, M. (1989a). Robot motion control based on joint torque sensing. In: Proc. *IEEE Int. Conf. Robotics and Automation*. pp. 256–1261.

Hashimoto, M. (1989b). Robot motion control based on joint torque sensing. In: *IEEE Int. Conf. Robotics and Automation*. pp. 256–261.

Hirose, S. and K. Yoneda (1990).Development of optical 6-axial force sensor and its signal calibration considering non-linear interference. In: Proc. *IEEE Int. Conf. Robotics and Automation*. Cincinnati. pp. 46–53.

Hirzinger, G., A Albu-Sc¨ahnle, I. Schaefer and N. Sporer (2001). On a new generation of torque controlled light-weight robots. In: *IEEE Int. Conf. On Robotics & Automation*. Seoul, Korea. pp. 3356–3363.

Ioannou, P. A. and J. Sun (1996). *Robust Adaptive Control*. Prentice Hall. New Jersey.

Jacobsen, S.C., F.M. Smith, D.K. Backman and E.K. Iversen (1991). High performance, high dexter ity, force reflective teleoperator ii. In: *ANS Topical Meeting on Robotics and Remote Systems*. Albuquerque, NM. pp. 24–27.

Kosuge, K., H. Takeuchi and K. Furuta (1990). Motion control of a robot arm using joint torque sensors. *IEEE Trans. Robotics and Automation* 6(2), 258–263.

Lebow (1997).Lebow load cell torque sensor handbook. Technical Manual 710.Eaton Corporation, Lebow Products. Troy, Michigan 48099, USA.

Lozano, R. and B. Brogliato (1992). Adaptive control of robot manipulators with flexible joints. *IEEE Trans. on Automatic Control* 37(2), 174–181.

Luca, A. De and P. Lucibello (1998). A general algorithm for dynamic feedback linearization of robots with elastic joints. In: *IEEE Int. Conf. On Robotics & Automation*. Leuven, Belgium. pp. 504– 510.

Luh, J. Y. S., M. W. Walker and R. P. Paul (1980). Resolved acceleration control of mechanical manipulator. *IEEE Trans. Automatic Control* 25(3), 468–474.

Luh, J.Y.S, W.D. Fisher and R.P.C. Paul (1983). Joint torque control bya direct feedback for industrial robot. *IEEE Trans. Automatic Control* 28, 153–161.

Murphy, S. H., J. T. Wen and G. N. Saridis (1990). Simulation and analysis of flexibly jointed manipulators. In: 29 *Th IEEE Conf. On Decision and Control*. Honolulu, HI. pp. 545–550.

Newman, W. S. and J. J. Patel (1991). Experiments in torque control of the adept-one robot. In: Proc. *IEEE Int. Conf. Robotics and Automation*. pp. 1867–1872.

Omega (1995).The pressure strain and force handbook. Technical Manual 29.OMEGA. Stamford, CT 06907-0047, USA.

Pfeffer, L. E., O. Khatib and J. Hake (1989). Joint torque sensory feedback in the control of a puma manipulator. *IEEE Trans. Robotics and Automation* 5(2), 418–425.

Reed, J. S. and P. A. Ionnou (1989). Instability analysis and robust adaptive control of robotic manipulator. *IEEE Trans. on Robotics and Automation* 5(3), 381–386.

Sciavicoo, L. and B. Siciliano (2000). *Modeling and Control of Robot Manipulator*. 2 ed.. Springer.

Slotine, J. J. E. (1985). The robust control of manipulators. *Int. J. of Robotics Research* 4(2), 49–64.

Slotine, J. J. E. and W. Li (1987). On the adaptive control of robot manipulators. *Int. J. of Robotics Research* 6(3), 49–59.

Spong, M. W. (1987). Modelingandcontrol of elastic joint robots. Transactions of the *ASME Journal of Dynamic Systems, Measurement, and Control* 109, 310–319.

Stokic, Dragan and Miomir Vukobratovic (1993). Historical perspectives and states of the art in joint force sensory feedback control of manipulation robots. *Robotica* 11, 149–157.

Svinin, M.M. and M. Uchiyama (1995). Optimal geometric structures of force/torque sensors. *The International Journal of Robotics Research* 14(6), 560–573.

Tani, Y., Y. Hatamura and T. Nagao (1983).Development of small three component dynamometer for cutting force measurement. *Bulletein of the JSME.*

Techniques, Transducer (n.d.). Load cell transducer. Technical manual. Transducer Techniques. 43178T Business Park Dr., CA 92590 (NR).

Tomei, P. (1991).A simple PD controller for robots with elastic joints. *IEEE Trans. on Automatic Control* 36, 1208–1213.

Uchiyama, M., E. Bayo and E. Palma-Villalon (1991). A systematic design procedure to minimize a performance index for robot force sensors. *Journal of Dynamic Systems, Measurement, and Control* 113, 388–394.

Vischer, D. and O. Khatib (1995). Design and development of high-performance torque-controlled joints. *IEEE Trans. Robotics and Automation* 11, 537–544.

Wu, C-H. (1985). Compliance control of a robot manipulator based on joint torque servo. *The International Journal of Robotics Research* 4(3), 55–71.

Wu, C-H. and R.P. Paul (1980). Manipulator compliance based on joint torque control. In: Proc. *IEEE Int. Conf. Decision and Control*. pp. 88–94.

Yong-Duan, S., G. Wei-Ningand C. Mian (1989). Studyon path tracking control of robot manipulators with unknown payload. In: *IEEE Int. Conf. On System Eng*. Fairborn, OH. pp. 321–324.

Zhang, G. and J. Furusho (1998). Control of robot arms using joint torque sensors. *IEEE Control systems* 18(1), 48–54.

Design and Validation of a Universal 6D Seam Tracking System in Robotic Welding Based on Laser Scanning

Mikael Fridenfalk, Gunnar Bolmsjö
Lund University
Sweden

1. Introduction

Robot automation increases productivity, the product quality and frees man from involuntary, unhealthy and dangerous work. While computational power has increased exponentially during the last decades, the limitations in flexibility constitute today a bottleneck in the further evolution of robotic systems.

One application for intelligent systems is seam tracking in robotic welding. Seam tracking is among others used for the manufacturing of large ships such as cruisers and oil tankers, where relatively high tolerances in the sheet material are allowed to minimize the manufacturing costs (Bolmsjö, 1997; Fridenfalk & Bolmsjö, 2002a). Seam tracking is today typically only performed in 2D, by a constant motion in x and compensation of the errors in y and z directions, see Fig. 1. There exist solutions for higher autonomy that include seam tracking in 3D where compensation is also performed in *o* direction or around an orientation axis. These limitations make however seam tracking only possible for work pieces with relatively simple geometrical shapes.

The next step in the evolution of sensor systems in robotic welding is the introduction of a full 6D sensor guided control system for seam tracking which is able to correct the TCP in x, y, z and around roll, pitch and yaw. Such an ultimate system is per definition able to follow any continuous 3D seam with moderate curvature. This system has many similarities with an airplane control system, where a change in either position or orientation affects all other degrees of freedom.

Though seam tracking has been performed in 2D for at least 40 years, the hypothetical design of 6D seam tracking systems was probably first addressed in the beginning of the last decade by suggestions of models based on force sensors (Bruyninckx et al.,1991a; Bruyninckx et al., 1991b) and laser scanners (Nayak & Ray, 1990a; Nayak & Ray, 1990b). It is however difficult to evaluate these systems, since no experimental results are presented, neither any explicit control systems for creating balance between the subsystems.

The contribution of this paper is the invention of a universal 6D seam tracking system for robotic welding, validated by simulation experiments based on physical experiments, which proved that 6D seam tracking is possible and even very robust for laser scanning (Fridenfalk & Bolmsjö, 2002b). On a more detailed level, the contributions of this paper are considered to be the introduction of: (1) differential control in laser scanning, using the same techniques as used in arc sensing (Cook et al., 1987), (2) trajectory tangent vector control using differential vector interpolation, and (3) pitch control by vector interpolation. These components constitute the foundation of the 6D seam tracking system.

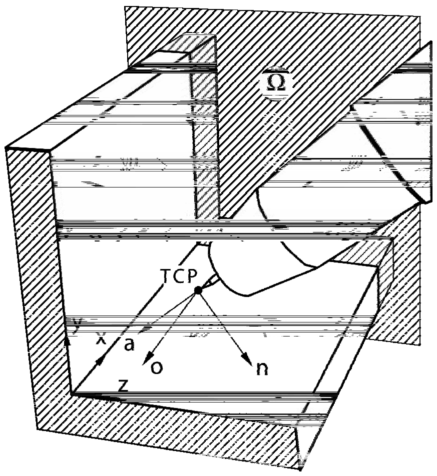


Fig. 1. Definition of Tool Center Point (TCP) and the orthonormal coordinate system {*n, o, a*}. Here, *o* is opposite to the direction of welding and perpendicular to the plane Ω, and *a* is the direction of approach. {x, y, z} is a local coordinate system, defined for the work-piece.

The authors have found only a few references from the literature which describe similar work, which is the introduction of a trajectory tangent vector by curve fitting (Nayak & Ray, 1990a; Nayak & Ray, 1990b). There exist however some essential differences between how such trajectory vector was used and implemented. The differences consist of (1) curve fitting by 2nd instead of 3rd degree polynomials, for faster computation and still high control stability, (2) using an analytic solver for curve fitting of 2nd degree polynomials developed and implemented for this system, increasing the calculation speed further and (3) using differential vector interpolation instead of direct use of the trajectory tangent vector, which showed to be essential for maintaining control stability.
The accuracy needed to perform robotic arc welding depends on the welding process. In our case we were working with GMAW (Gas Metal Arc Welding). The accuracy needed to follow the seam and position the weld torch relative the weld joint is in the order of ± half diameter of the welding wire, or ±0.6 mm in our case. As indicated above and shown in our simulations, the curvature or radius is an important measure of the performance of the system. As shown from simulation experiments, seam tracking curves with a radius down to 20 mm produces no added positional error during continuous path motion in the sensor feed-back loop which controls the robot. However, the torch angle changes from a 4 degrees deviation at 50 mm radius to 12 degrees at 20 mm radius. Thus, the accuracy needed for this process can be obtained by the system provided the sensor and robot have matching specifications. As indicated by the results, the system can be used for welding processes which requires higher accuracy providing the robot and sensor can meet this as well.
A 6D seam tracking system increases the capability and flexibility in manufacturing systems based on robotic welding using laser scanning. This reduces the need for accurate robot trajectory programming and geometrical databases. The simulation results (based on physical experiments) showed that the initial objective to design a 6D seam tracking system

was reached that could manage a radius of curvature down to 200 mm. Furthermore, if low-speed seam tracking is performed without welding, to find the geometry of the work-piece, there is basically no limit how small radius of curvature this system is able to manage.

2. Materials and methods

2.1 Sensors guided robot control

Many systems have been developed during the last decades for seam tracking at arc welding. The patents and scientific publications within this application area show that laser scanners and arc sensors today replace older systems. Laser scanners have in general high accuracy but are expensive and have to be mounted on the torch, thereby decreasing the workspace of the robot. This problem is partly solved by introduction of an additional motion that rotates the scanner around the torch to keep its relative orientation constant with respect to the seam profile. In practical applications presently, solutions based on pure image processing by a CCD camera are further successfully used to find the initial point where the seam tracking should start (Fridenfalk & Bolmsjö, 2002a), in addition to the laser scanner sensor.

2.1.1 Laser scanning

In laser scanning, the seam profile is extracted by periodically scanning the laser beam across a plane perpendicular to the direction of the seam. The laser scanner is mounted on the torch in front of the direction of welding. As a general rule, the control system is based on full compensation of the position errors, which is equal to a constant value of $K_1 = 1$, as defined in Fig. 2.

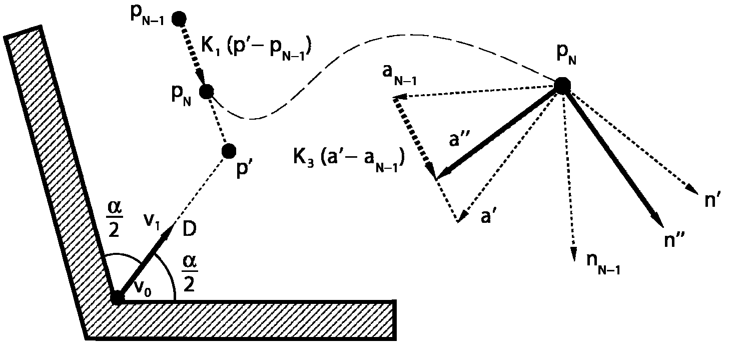


Fig. 2. Cross section, plane Ω. Vector interpolation used for position and pitch control at laser scanning. The input from the laser scanner system is the intersection point of the seam walls v_0 and the unit vector v_1 on Ω between the seam walls. D denotes the distance between v_0 and p′.

2.1.2 Arc sensing

Another method used for seam tracking is "through-arc" sensing or simply arc sensing (Cook et al., 1987). Here, the arc current is sampled while a weaving motion is added to the TCP in n direction, see Fig. 1. Since the distance from TCP to the nearest seam wall may be approximately calculated as a function of the current, it is possible to compensate for deviations in the seam by analysis of the current measurements. Arc sensors are inexpensive, do not decrease the workspace of the robot, but are compared with other sensors for seam tracking relatively inaccurate due to the stochastic nature of arc welding. Since welding of large structures in general requires a relatively low accuracy however, arc sensors are a competitive alternative to laser scanners. Arc sensing may also be used as a complementary sensor to laser scanners in some cases where the laser scanner is unable to access the seam track.

2.1.3 Virtual sensors

The development of sensor guided control systems may be accelerated using virtual sensors in the simulation control loop. Since a system's state is often easily obtained in simulation, assuming that the simulation model is sufficiently accurate compared to the real system, this provides for a better foundation for process analysis than in general is possible by physical experiments. Furthermore, insertion of a sensor in a control loop may cause damage to expensive equipment, unless the behaviour of the entire sensor guided control system is precisely known, which is rarely the case at the development stage. Virtual sensors are used in many application areas, such as robotics, aerospace and marine technologies (Bolmsjö, 1999; Dogget & Vazquez, 2000; Hailu & Zygmont, 2001; Oosterom & Babuska, 2000; Ridao et al., 2001). Development of new robot systems such as for seam tracking may be accelerated by the use of simulation (Fridenfalk et al., 1999a; Fridenfalk et al., 1999b). In the development of virtual sensors, if the model is not analytically definable, artificial neural networks or statistical methods are often used (Arroyon et al., 1999; Pasika et al., 1999).

2.2 Robot system

The design of the 6D seam tracking system was based on experiments and results from the European ROWER-2 project (European Commission, 2002). The objective of this project was to automate the welding process in shipbuilding by the design of a robotic system, specifically for joining double-hull sections in super-cruisers and oil tankers. Figure 3 shows the robot system developed in the ROWER-2 project and the corresponding simulation model in FUSE (described below). The implementation of the 3D seam tracking system was performed in C/C++, running on a QNX-based embedded controller (QNX is a real-time operating system for embedded controllers). The robot was manufactured partly in aluminium alloy to decrease the weight for easy transportation and mounted on a mobile platform with 1 degree of freedom for increased workspace. The robot system was integrated and successfully validated on-site at a shipyard.

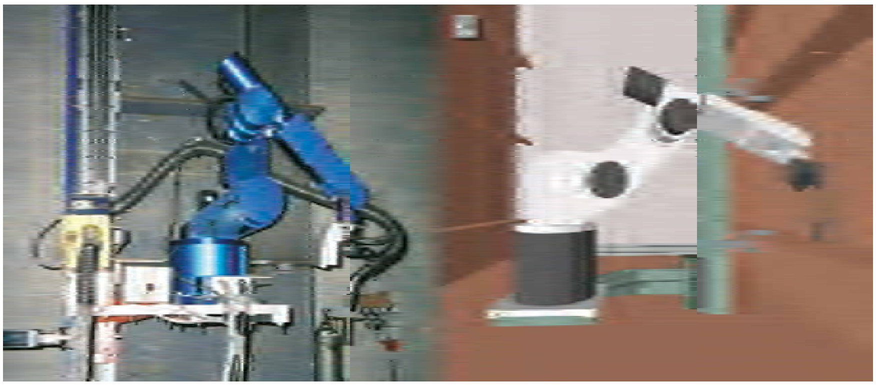


Fig. 3. The design of the robot system for 3D seam tracking based on arc sensing in the ROWER-2 project (left) was based on simulations in FUSE (right). The 6D seam tracking system presented here was developed based on experience and results gained from the ROWER-2 project.

2.3 Simulation system

The development and implementation of the 6D seam tracking system was initially performed in the Flexible Unified Simulation Environment (FUSE) (Fridenfalk & Bolmsjö, 2001). FUSE is an integrated software system based on Envision (robot simulator, Delmia Inc.) and Matlab (MathWorks Inc.), including Matlab Robot Toolbox (Corke, 1996). The integration was performed by compiling these software applications and their libraries into one single software application, running under IRIX, SGI. The methodology developed for the design of the 6D seam tracking system was based on the integration of mechanical virtual prototyping and software prototyping. The 6D seam tracking simulations in FUSE were performed by using the robot developed in the ROWER-2 project and an ABB S4 IRB2400.16 robot, see Figs. 3 and 4. After the design and successful validation of a 6D seam tracking system for arc sensing, the virtual prototyping and software models, partly written in C-code and partly in Matlab code, were translated to pure Matlab code. The Matlab models were modified to work also for laser scanners by the addition of a virtual model of the Servo-Robot M-SPOT laser scanner system. The translation was necessary, since it saved development time for the implementation of the evaluation system, for the accurate calculation of position and orientation errors during simulation.

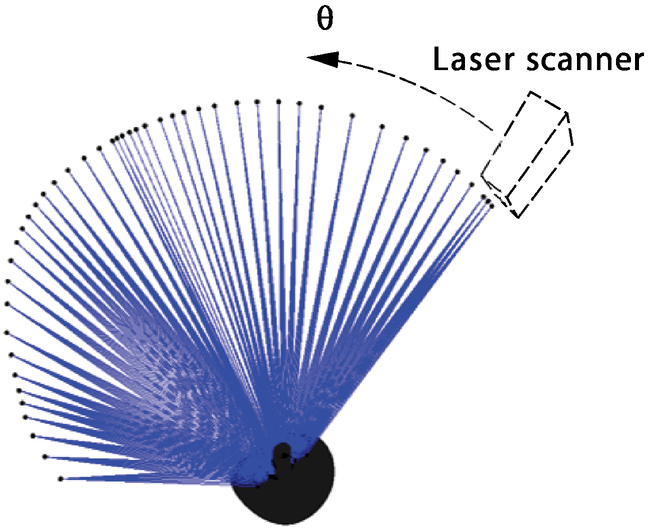


Fig. 4. Example of a 6D seam tracking experiment performed in Matlab. Each laser scan is plotted at each control cycle while seam tracking is performed 180° around a pipe intersection. The work-piece object corresponds to the object in the upper experiment in Fig. 7.

2.4 Calculation of position and orientation errors

Error is defined as the difference between desired and current pose **P**. The position errors were calculated by projection of ε_a and ε_n on a plane Ω_t perpendicular to the desired trajectory while intersecting current TCP, $\mathbf{P}_t$. Thereby, per definition $\varepsilon_o = 0$. The orientation error (the orientation part of $\mathbf{P}_\delta$) is calculated by:

$$\mathbf{P}_\delta = \mathbf{P}_t^{-1} \cdot \mathbf{P}_{\Omega t} \tag{1}$$

where $\mathbf{P}_t$ is the current TCP and $\mathbf{P}_{\Omega t}$ is the desired TCP for any sample point t. The errors around n, o and a were calculated by the subsequent conversion of the resulting transformation matrix to roll, pitch and yaw, here defined as rotation around, in turn, a, o and n. Rotation in the positive direction is defined as counter clockwise rotation from the perspective of a viewer when the axis points towards the viewer.

3. Modelling

In this section the design of the 6D seam tracking system is described. The principal control scheme of this system is presented in Fig. 5. The control system is based on three main components, position, trajectory tangent vector and pitch control. Pitch denotes rotation around o. The main input of this control system is the 4×4 transformation matrix $\mathbf{P}_{N-1}$, denoting current TCP position and orientation. The output is next TCP, $\mathbf{P}_N$. N is the number of samples used for orientation control. The input parameters are N, the proportional control constants K_1, K_2 and K_3, and the generic sensor input values $\xi = [v_0\ v_1]$ and $\zeta = v_1$, where v_0 and v_1 are defined in Fig. 2. The proportional constants K_1, K_2 and K_3 denote the control response to errors in position, trajectory tangent vector and pitch control. N is also used in trajectory tangent vector control and is a measure of the memory size of the algorithm. The seam profile presented in Fig. 1 is a fillet joint. Since v_0 and v_1 may be extracted for any joint with a predefined signature, no other joints were required in the experiments.

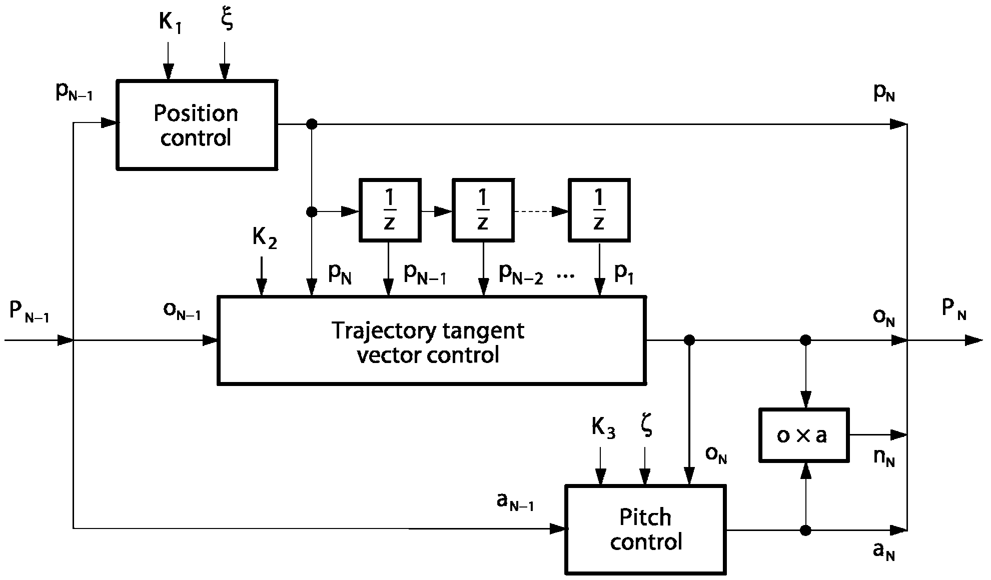


Fig. 5. The principal scheme of the 6D seam tracking control system. $\mathbf{P}_{N-1}$ and $\mathbf{P}_N$ denote current and next TCP. $\mathbf{P}_N$ is calculated for each control sample designated for the seam tracking motion.

3.1 Position control

The 6D seam tracking algorithm starts by the calculation of next position for the TCP. Next position p_N is calculated by vector interpolation based on K_1 in Fig. 2, between current position p_{N-1} and the position p′ extracted from the laser scanner.

3.2 Trajectory tangent vector control

The positions $p_1 \ldots p_N$ are used for trajectory estimation by curve fitting, see Fig. 6. To decrease calculation time, an analytical estimator was devised for 1st and 2nd degree polynomials that showed to work much faster than traditional matrix multiplication followed by inversion techniques using Gauss elimination. The estimation of the trajectory at N gives o'. Since an attempt to abruptly change the direction of o from o_{N-1} to o' would most probably create instability in the system, the motion between o_{N-1} and o' is balanced by vector interpolation using K_2. The scalar κ_2 is not explicitly used in the calculations, but used for visualization of the calculation of o_N in Fig. 6.

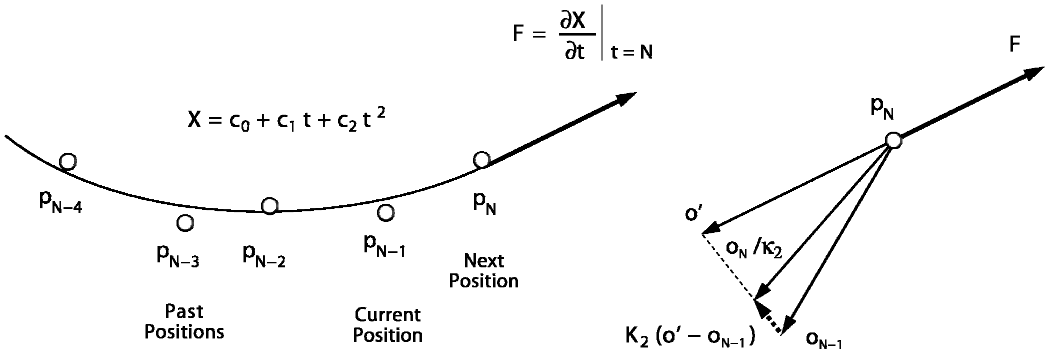


Fig. 6. Trajectory tangent vector control is performed by the least square curve fitting of the last N trajectory points (left) to a 2nd degree polynomial curve for x, y and z, followed by vector interpolation (right). Here N = 5. X, c_0, c_1 and c_2 denote vector entities. The trajectory tangent vector F is for a 2nd degree polynomial equal to $c_1 + 2Nc_2$.

3.3 Pitch control

The first step in pitch control is performed by vector interpolation between a_{N-1} and a', using K_3, see Fig. 2. In a second step, a_N is calculated by orthogonally projecting a'' on a plane perpendicular to o_N and normalization of the resulting vector. Finally, n_N is calculated by the cross product $o_N \times a_N$.

3.4 Code and pseudo code samples

The actual implementation of the 6D seam tracking system showed to be straightforward. Most of the development effort was spent on the design of the surrounding environment of the seam tracking system, including a semi-automatic testing and evaluation system. In simulation, the plane Ω is assumed to intersect the TCP point. In reality however, the laser scanner is mounted on the torch, often several centimetres ahead of the direction of the motion. The transformation to such system at real-time control is performed by storing the trajectory data for use with a time delay. A pseudo code representation of the 6D seam tracking system in Fig. 5, follows below:

1. $p_N = p_{N-1} - r_w \cdot o_{N-1} + K_1 \cdot (v_0 + D \cdot v_1 - p_{N-1})$, where v_0, v_1 and D are defined in Fig. 2 and r_w is the nominal displacement in the direction of welding during a control sample.
2. Calculation of x_N by shifting x_{N-1} one step to the left with p_N: $x_{N-1} = [p_0\ p_1 \ldots p_{N-2}\ p_{N-1}] \rightarrow x_N = [p_1\ p_2 \ldots p_{N-1}\ p_N]$.
3. Estimation of the trajectory tangent vector F by least square curve fitting of the parameterized 3D curve $X = C \cdot T$ with a parameter t to 3 independent second-degree

polynomials (for x, y and z). $F = \partial X/\partial t|_{t=N}$ with $X = C \cdot T$, where $C = [c_0\ c_1\ c_2] = [c_x\ c_y\ c_z]^T$ and $T = [1\ t\ t^2]^T$ (Note that A^T is the transpose function for a vector or matrix A and has no correlation with the vector T). This gives $F = C \cdot \partial T/\partial t|_{t=N} = C \cdot [0\ 1\ 2t]^T|_{t=N} = C \cdot [0\ 1\ 2N]^T$. C is calculated by the estimation of the parameters c_i for i = {x, y, z}, where $c_i = (A^TA)^{-1}(A^Tp_i)$ and p_i are the row vectors of x_N, which may also be written as $x_N = [p_x\ p_y\ p_z]^T$. The matrix A consists of N rows of $[1\ t\ t^2]$ with $t = 1...N$, where t is the row number. It should be mentioned that C may be directly calculated by the expression $C = ((A^TA)^{-1}(A^T\ x_N^T))^T$, but we have chosen to use the notation above for pedagogical reasons.

4. $o_N = \text{normalize}(o_{N-1} - K_2 \cdot (\text{normalize}(F) + o_{N-1}))$. Normalization of a vector v denotes v divided by its length.
5. $a'' = a_{N-1} - K_3 \cdot (a_{N-1} + v_1)$.
6. $a_N = \text{normalize}(a'' - \text{dot}(o_N, a'') \cdot o_N)$, where dot (o_N, a'') is the dot product $o_N^T \cdot a''$.
7. $\mathbf{P}_N = \begin{bmatrix} o_N \times a_N & o_N & a_N & p_N \\ 0 & 0 & 0 & 1 \end{bmatrix}$

The initial value of x is determined by linear extrapolation with respect to the estimated direction at the beginning of the seam. The elements of F(i) with i = {1, 2, 3} for {x, y, z} were here calculated by the Matlab function pfit((1:N)′,x(i,:)′) (the input parameters x and y in this function are $[1\ 2\ ...\ N]^T$ and b_i as defined above). The pfit function was developed in this work using Maple (Waterloo Maple Inc.) and gives an analytical solution to 2nd degree polynomial curve fitting for this application, that showed to work 8 times faster for N = 10 and 4 times faster for N = 100 than the standard Matlab function polyfit. The analytical function pfit was implemented according to below:

```
function k = pfit(x,y)
    n = length(x); sx = sum(x); sy = sum(y); sxx = x'*x; sxy = x'*y;
    sx3 = sum(x.^3); sx4 = sum(x.^4); sxxy = sum(x.^2.*y);
    t2 = sx*sx; t7 = sx3*sx3; t9 = sx3*sxx; t12 = sxx*sxx;
    den = 1/(sx4*sxx*n-sx4*t2-t7*n+2.0*t9*sx-t12*sxx);
    t21 = (sx3*n-sxx*sx)*t15;
    k = 2.0*n*((sxx*n-t2)*t15*sxxy-t21*sxy+(sx3*sx-t12)*t15*sy)- ...
    t21*sxxy+(sx4*n-t12)*t15*sxy+(t9-sx4*sx)*t15*sy;
```

The calculation speed was optimization by the application of Cramer's rule by an analytical inversion of A^TA in Maple and the elimination of c_0. In the function above, x and y are N × 1 column vectors, x′ equals x^T, sum(x) is the sum of the vector elements of x, and a point before an operator such as the exponential operator ^, denotes element-wise operation. This implementation may be further optimized by reusing terms that only depend on x in pfit (which is equal to t outside pfit). Since such optimization is however outside the scope of this paper, we have chosen to keep this presentation as simple as possible.

3.5 Computational costs

The computational costs for the 6D seam tracking system based on laser scanning (without consideration to memory operations or index incrementation) was preliminary estimated to less than 250 + 50N floating point operations for the implementation presented in this paper, which for N = 10 gives 750 operations per control cycle. At a rate of maximum 30 scans per second, this gives less than 22.500 floating point operations per second.

3.6 Kinematics singularities

In sensor guided control, the manipulator may involuntarily move into areas in which no inverse kinematics solutions exist, generally called kinematics singularities. It is however possible to minimize inner singularity problems caused by robotic wrists. A novel method is suggested in (Fridenfalk, 2002) which was designed for the 6D seam tracking system, but works basically for any industrial robot with 6 degrees of freedom. In this method, the stronger motors, often at the base of the robot, assist the weaker wrist motors to compensate for any position error. This method allows also for smooth transitions between different configurations. In addition to this, in one-off production or for generation of a reference program, pre-scanning and trial run may be performed before welding. In the first step, low-speed (or high-speed) seam tracking is performed to calculate the trajectory of the TCP, following the seam. In the second step, a trial run is performed in full speed to check the performance of the robot, and finally the actual welding is performed.

3.7 Initial and final conditions

In the experiments, seam tracking was initiated from a proper position already from start. The start position for seam tracking is in general found either by image analysis using a camera or by seam tracking itself. Since previous positions are unknown at start, estimation is made of previous $N - 1$ positions, by extrapolation of the trajectory in the direction of o. This is the reason why in some experiments, higher orientation errors occurred at the beginning of seam tracking. The condition to stop seam tracking may be based on the position of the TCP. The stop signal may be triggered by definition of volumes outside which no seam tracking is allowed.

4. Experimental results

4.1 Work-piece samples

The position and orientation errors at seam tracking were calculated based on simulation experiments of basically 8 different work-pieces, see Figs. 7-11. It should be mentioned that the experiments in the bottom of Fig. 8 and Fig. 9 are not actually possible in practice due to collision problems, but are performed for the analysis of the control system. The work-pieces in Matlab were approximated by lines orthogonal to the direction of the weld and the resolution was defined as the nominal distance between these lines. The resolution was here set to 0.02 mm. The reason why no noise was added to the profile obtained from the laser scanner (except for perhaps decreasing the number of measurement points compared to the M-SPOT, which may use up to 512 points per scan), was because large number of measurements allow for high-precision estimations of v_0 and v_1, with a constant offset error less than the accuracy of the laser scanner. Since the 6D seam tracking system corrects the position and the orientation of the TCP relative to the extracted profile, such error will not influence the behaviour of the control system more than highly marginally, by basically adding a small offset error to the experimental results in Figs. 7-11. The work-pieces are categorized into two groups, continuous and discrete. In the experiments presented in this paper, the range was $0-180°$ for continuous and 12 mm for discrete work-pieces. In seam tracking of a continuous work-piece, sudden changes is interpreted as a curve with a small radius of curvature, see Figs. 7-8, while in seam tracking of a discrete work-piece a sudden change is regarded as a disturbance, see Figs. 10-11. The settings of the control parameters decide alone the mode of the control system: continuous or discrete. For seam tracking of objects with a large radius of curvature, the discrete mode works well for both continuous and discrete work-pieces, see Fig. 9.

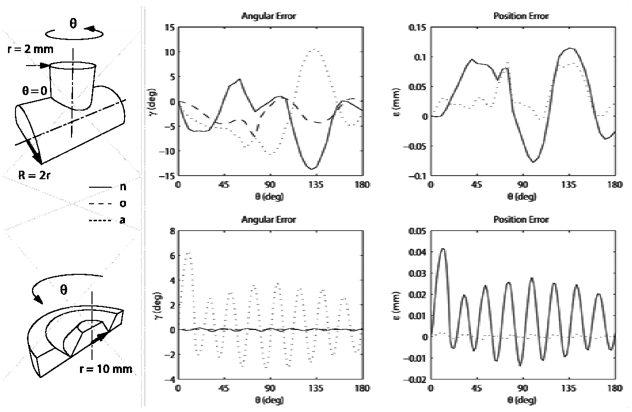


Fig. 7. Seam tracking, laser scanning. Pipe intersection (top) and work-piece for isolating control around a circular seam (bottom). K_1 = 1.0, K_2 = K_3 = 0.8, N = 10. Error denotes the difference between desired and current pose.

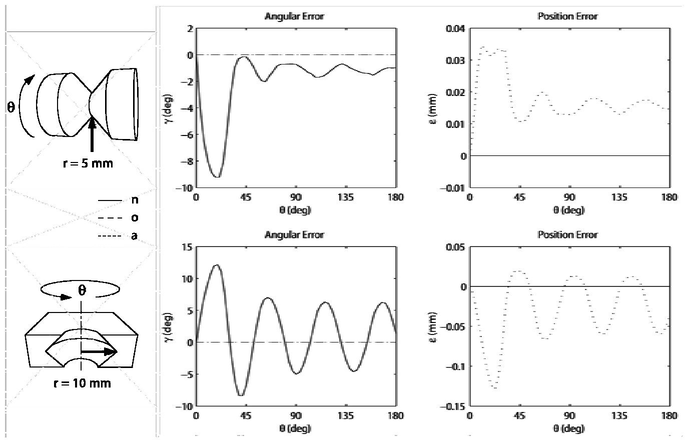


Fig. 8. Seam tracking, laser scanning. Isolating rotation around *n*, following inner and outer curves. Note that in this figure, the case in the bottom is not very realistic, but only included for the completeness of the experiments. K_1 = 1.0, K_2 = K_3 = 0.8, N = 10.

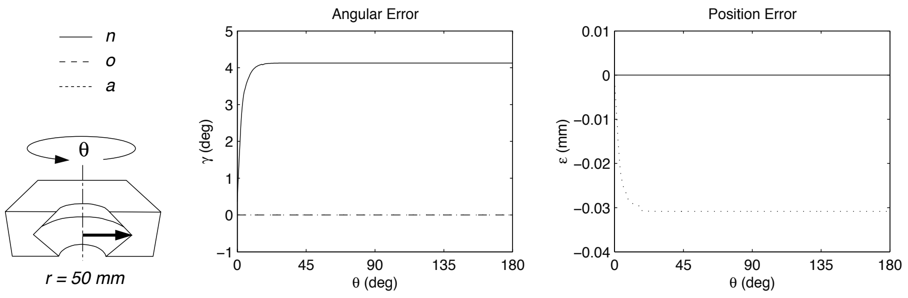


Fig. 9. Seam tracking, laser scanning. The same as in previous figure, except for K_2 = 0.1 and r = 50 mm. This shows that K_2 = 0.1 works fine as well for continuous seams with moderate radius of curvature. Further, the same experiment was also performed with r = 20 mm,

which showed to be basically identical to this figure, but with saturation at 12 degrees instead of 4. Note that while other laser scanning experiments were performed on the verge of instability to find the limits of the seam tracking system, using the M-SPOT laser scanner, the system is highly stable at moderate curvatures using low values for K_2.

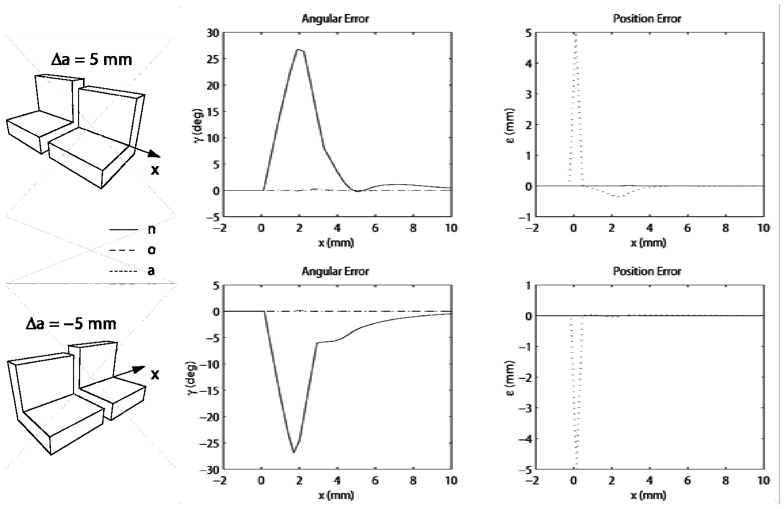


Fig. 10. Seam tracking, laser scanning. The step is introduced at x = 0 mm. K_1 = 1.0, K_2 = 0.1, K_3 = 0.8, N = 10.

The control parameters K_1, K_2, K_3 and N used in the presented experiments showed to work well for each category, continuous and discrete. These settings were found by more than 50 laser scanning simulation experiments in Matlab, and large deviation from these parameters showed to create instability in the system, making the control components working against each other instead of in synergy, resulting in divergence from the seam and ultimately failure at seam tracking. The seam tracking system is theoretically able to handle any angle α (in Fig. 2) between 0 and 180°. In practice however, a small angle may cause collision between torch and work-piece. For simplicity, α was set to 90° for all experiments except for pipe intersections.

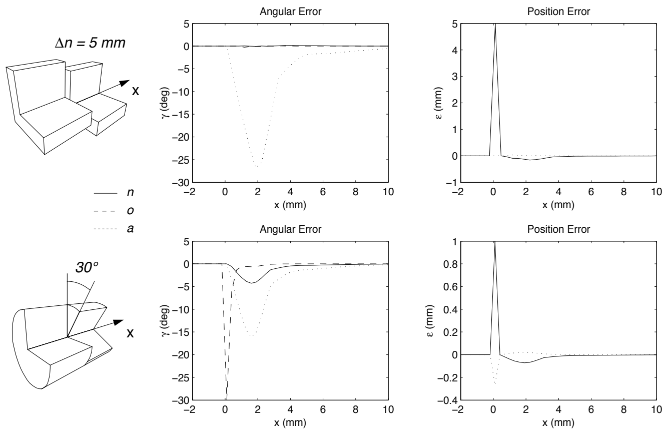


Fig. 11. Seam tracking, laser scanning. In this figure, the case in the bottom denotes a step of −30° performed around *o*. K_1 = 1.0, K_2 = 0.1, K_3 = 0.8, N = 10.

4.2 Laser scanning

The reference laser scanner, M-SPOT, used in the experiments has a sweeping angle of 28^o, an accuracy better than 0.02 mm at the distance of 100 mm and 1.5 mm at a distance of 1 m. It is able to scan at a frequency of up to 40 Hz with a resolution of maximum 512 points per scan. The virtual model of the M-SPOT that was designed, were programmed to have an accuracy of 0.02 mm and was mounted on the torch at a distance of approximately 80 mm from the work-piece. The resolution was set to 11 points per scan for a sweeping frequency of 5^o (the only exception was steps in the n direction, see Fig. 11, where 31 points per scan with a sweeping angle of 10^o was used instead) and the scanning frequency to 30 Hz, which showed to be fully sufficient for successful seam tracking.
In fact, the simulation experiments proved that the accuracy of M-SPOT was higher than needed for seam tracking. The accuracy is so high in close range (100 mm) that only robots with very high performance are able to use the accuracy of this sensor to its full extent, when seam tracking is performed in real-time. The experiments presented in Figs. 7-8 and 10-11 show the limits of performance for the 6D seam tracking system, when it is at the verge of instability. For larger radius of curvature and a lower value for K_2, i.e. for normal operation, Fig. 9 gives perhaps a better indication of the efficiency of the 6D seam tracking system. Since the performance of a robot is dependent on current joint values, it is further not possible to translate these values directly to joint positions, velocities and accelerations. What these experiments proved however, was that it was the performance of the robot that constituted the limitations in seam tracking based on laser scanning.
The laser scanning simulations presented in this paper were performed with a constant welding speed of 10 mm/s (which equals 3 scans/mm at a scanning frequency of 30 Hz) and a virtual TCP, 2 mm away from the intersection point of the seam walls. Empirical experiments showed that by moving the TCP from previously D = 15 mm in Fig. 2 to only 2 mm, the control ability of the 6D seam tracking was highly improved, especially in seam tracking of objects with a small radius of curvature. Though the parameter settings $K_1 = K_2 = K_3 = 1$ showed to be unstable in laser scanning, $K_1 = 1$, $K_2 = K_3 = 0.8$ showed to be relatively stable for continuous welds. For discrete work-pieces, K_2 was set as low as 0.1 to maximize control stability.

5. Results and discussion

A 6D seam tracking system was designed and validated by simulation that is able to follow any continuous 3D seam with moderate curvature, using a laser scanner, and is able to continuously correct both position and orientation of the tool-tip of the welding torch along the seam. Thereby the need for robot trajectory programming or geometrical databases is practically eliminated, except for start and end positions. Simulations based on the M-SPOT laser scanner showed that 6D seam tracking was theoretically possible of objects less than a radius of curvatures of 2 mm. Seam tracking in real-time requires however a robot with very high performance. It is however possible to perform seam tracking without welding to find the geometry of the work-piece, and perhaps even checking that the trajectory is within the workspace of the robot, before actual welding is performed. Thereby it will be possible to perform welding of objects with small radius of curvature, theoretically also including sharp edges. This is much better than the initial objective to design a 6D seam tracking system that could manage a radius of curvature down to 200 mm, which is still considered as relatively small for robotic welding in for instance shipbuilding applications.

As indicated in section 2 an adaptation of the algorithm presented here can be used for arc sensing. Depending on the actual welding application, the two sensors can be used. In general, laser scanning can be used for search, pre-scanning and during welding with a very high accuracy while arc sensing only can be used during welding and during weaving. In general, arc sensing has a lower accuracy but enough for GMAW while laser scanning can be applied for most welding processes, provided there is space for the laser scanning in front of the welding torch. As laser scanning provides richer information about the weld joint, i.e. joint geometry and distance to the weld joint in addition to the specific information used in seam tracking, it can be used for optimized control of the welding process based on information in a WPS (Welding Procedure Specification). Seen from a general perspective, our results indicate a foundation for sensor guidance using a laser scanner for robotic arc welding. A future development from this point will be to integrate the WPS to achieve the specified productivity and quality which will be important when automation of low volume down to one-off production is possible.

6. Future work

Though the 6D seam tracking system showed to be very robust, optimization of the presented algorithm is still possible. The control system worked well using proportional constants alone, but if needed, for instance PID or adaptive control may be included to enhance the performance. As an example, the control parameter K_2 could be defined as a function of the "radius of curvature" (or more specifically as a function of for example c_2 in Fig. 6) of the 2nd degree polynomial used for calculation of the trajectory tangent vector, thereby enable the system to automatically change between normal operation, used for maximum control stability, and extraordinary operation with fast response, used for management of small radius of curvatures.

7. References

Arroyo Leon, M.A.A., Ruiz Castro, A. and Leal Ascencio, R.R. (1999). An artificial neural network on a field programmable gate array as a virtual sensor. In *Proceedings of the Third International Workshop on Design of Mixed-Mode Integrated Circuits and Applications*, pages 114–117, Puerto Vallarta, Mexico.

Bolmsjö, G. (1997). *Sensor systems in arc welding.* Technical report, Robotics Group, Dep. of Production and Materials Engineering, Lund University, Sweden, 1997.

Bolmsjö, G. (1999). Programming robot welding systems using advanced simulation tools. In *Proceedings of the International Conference on the Joining of Materials*, pages 284–291, Helsingør, Denmark.

Bruyninckx, H., De Schutter, J. and Allotta, B. (1991a). Model-based constrained motion: A. modeling, specification and control. In *Advanced Robotics*, 1991. 'Robots in Unstructured Environments', 91 ICAR, pages 976–981.

Bruyninckx, H., De Schutter, J. and Allotta, B. (1991b). Model-based constrained motion: B. introducing on-line identification and observation of the motion constraints. In *Advanced Robotics*, 1991. 'Robots in Unstructured Environments', 91 ICAR, pages 982–987.

Cook, G. E., Andersen, K., Fernandez, K. R., Shepard, M. E. and A. M. Wells. (1987). Electric arc sensing for robot positioning control. *Robot Welding*, pages 181–216. Editor: J. D. Lane. IFS Publications Ltd, UK, Springer-Verlag.

Corke, P. (1996). A robotics toolbox for MATLAB. *IEEE Robotics and Automation Magazine*, 3(1):24–32, Mar. 1996.

Doggett, W. and Vazquez, S. (2000). An architecture for real-time interpretation and visualization of structural sensor data in a laboratory environment. In *Digital Avionics Systems Conferences, 2000. Proceedings. DASC.* Volume 2, pages 6D2/1–6D2/8.

European Commission (2002). *ROWER-2: On-Board Automatic Welding for Ship Erection – Phase 2.* Record Control Number 39785. www.cordis.lu, update 2002-11-05.

Fridenfalk, M. (2002). *Eliminating position errors caused by kinematics singularities in robotic wrists for use in intelligent control and telerobotics applications.* Patent application SE0203180-5.

Fridenfalk, M. and Bolmsjö, G. (2001). The Unified Simulation Environment —Envision telerobotics and Matlab merged in one application. In *Proceedings of the Nordic Matlab Conference*, pages 177–181, Oslo, Norway.

Fridenfalk, M. and Bolmsjö, G. (2002a). Design and validation of a sensor guided robot control system for welding in shipbuilding. *International Journal for the Joining of Materials*, 14(3/4):44–55.

Fridenfalk, M. and Bolmsjö. G. (2002b). *Intelligent system eliminating trajectory programming and geometrical databases in robotic welding.* Patent application SE0203239-9.

Fridenfalk, M., Lorentzon, U. and Bolmsjö, G. (1999). Virtual prototyping and experience of modular robotics design based on user involvement. In *Proceedings of the 5th European Conference for the Advancement of Assistive Technology*, pages 308–315, Düsseldorf, Germany.

Fridenfalk, M., Olsson, M. and Bolmsjö, G. (1999). Simulation based design of a robotic welding system. In *Proceedings of the Scandinavian Symposium on Robotics 99*, pages 271–280, Oulu, Finland.

Hailu, G. and Zygmont, A. (2001). Improving tracking behavior with virtual sensors in the loop. In *Proceedings of the IEEE Aerospace Conference*, pages 1729–1737.

Nayak, N. and Ray, A. (1990a). An integrated system for intelligent seam tracking in robotic welding. Part I. Conceptual and analytical development. In *Proceedings of the IEEE Conf. on Robotics and Automation*, pages 1892–1897.

Nayak, N. and Ray, A. (1990b). An integrated system for intelligent seam tracking in robotic welding. Part II. Design and implementation. In *Proceedings of the IEEE Conf. on Robotics and Automation*, pages 1898–1903.

Oosterom, M. and Babuska, R. (2000). Virtual sensor for fault detection and isolation in flight control systems — fuzzy modeling approach. In *Proceedings of the IEEE Conf. on Decision and Control*, pages 2645–2650.

Pasika, H., Haykin, S., Clothiaux, E. and Stewart, R. (1999). Neural networks for sensor fusion in remote sensing. In *Proceedings of the Int. Joint Conf. on Neural Networks, IJCNN '99.* pages 2772– 2776.

Ridao, P., Battle, J., Amat, J. and Carreras, M.(2001). A distributed environment for virtual and/or real experiments for underwater robots. In *Proceedings of IEEE Int. Conf. on Robotics and Automation*, pages 3250–3255.

4

Designing Distributed, Component-Based Systems for Industrial Robotic Applications

Michele Amoretti[1], Stefano Caselli[1], Monica Reggiani[2]
[1]*University of Parma*
Italy
[2]*University of Verona*
Italy

1. Introduction

The design of industrial robotic applications is dramatically changing as the result of rapid developments in underlying technologies and novel user requirements. Communication and computing technologies provide new standards and higher levels of performance. Meanwhile, novel and more capable robotic platforms and sensors have become available. In industrial automation, this technology push has opened up new application areas, while raising the expectations casted upon robotic systems in terms of overall performance and functionality.
In next generation robotic applications, robots will turn from traditional preprogrammed, stationary systems into machines capable of modifying their behaviour based on the interaction with the environment and other robots. Changes are also induced by user requirements demanding higher efficiency, functionality and performance in the control of industrial plants. In a modern manufacturing scenario, interaction with customers and suppliers must increasingly occur remotely and on-line, *e.g.* for just-in-time manufacturing, remote maintenance, and quality monitoring. For a number of reasons, transparent and widespread access to run-time production data and internal control variables of robotic systems must be provided. This requirement highlights the distinct transition beyond the conventional idea of isolated robotic workcell. As an added element of complexity, integration between different technologies in the shop floor and in the design and production management departments is required to achieve the full potential of computer-integrated manufacturing.
The growth in complexity in industrial robotic applications demands higher levels of abstraction. A first step toward the improvement of software quality and cost reduction has been the increasing adoption of object-oriented reuse techniques, based on components and frameworks. The next step is to gain a comparable acceptance of reusability and transparency concepts as far as the distributed nature of industrial automation systems is concerned. Indeed, industrial automation applications fall naturally into distributed architectures, since they require the ability to connect multiple robotic and sensory devices, which must be controlled in a coordinated manner. As discussed in this chapter, communication middlewares are the key technology to achieve these goals.

The exploitation of standard components and distributed systems technology as key interconnection infrastructure can help to alleviate many of the complexity issues associated with the requirements of flexibility, fast implementation, maintainability, and portability. However, requirements arising in industrial automation provide additional challenges to application developers, beyond those faced when middleware technology is deployed in standard Internet applications. Control of manufacturing lines and data acquisition systems requires reliable and high-performance architectures operating in real-time. Resorting to custom protocols to satisfy such requirements in the context of complex systems is no longer viable. The need to reconfigure systems quickly and frequently for competitiveness leads to the development of heterogeneous systems built by integrating a mix of new and legacy equipment, with a variety of hardware, operating systems, and programming language to be taken into account. Another challenge confronting developers is the very size and complexity of industrial systems that often prevent a full understanding of the overall system and libraries, making hard for developers to predict the side-effects of future changes.
The purpose of this chapter is to describe how mainstream and advanced features of an object-oriented middleware can be exploited to meet the requirements of novel industrial robotic applications. We identify CORBA as the middleware best suited for such applications. More specifically, we review a number of CORBA services and show how Real-Time CORBA extensions and asynchronous method invocation meet performance and functional requirements. CORBA services for concurrency control and large-scale data distribution provide an effective infrastructure to meet common requirements arising in the control of distributed devices. Limitations in currently available implementations of the CORBA standards (*e.g.* for fault-tolerance and security) are also discussed.
The remaining of the chapter is organized as follows. Section 2 describes the fundamental requirements to be faced in the development of distributed industrial applications. Section 3 provides a brief overview of the two most prominent actors of the distributed middleware scenario, Web Services and CORBA. Section 4 illustrates and evaluates the most advanced CORBA features and other CORBA services relevant to distributed industrial robotic applications. Section 5 presents a sample CORBA-based application including a robotic manipulator platform and several sensor devices. Finally, Section 6 concludes the chapter, discussing its main results.

2. Requirements of Distributed Industrial Robotic Applications

Developing a suitable system architecture meeting requirements for distributed industrial robotic applications is a difficult and time-consuming task. In many cases, the expensive development of a new application can be avoided relying on previous experience or, even better, on a common framework from which specific architectures are instantiated. Stemming from the features characterizing novel distributed robotic applications, we identify the following set of requirements for the system architecture.
The system must ensure **interoperability** and **location transparency**. In modern industrial plants many embedded processors and servers are networked together requiring their software to handle communications and interoperability issues while ensuring reliability and performance. To achieve portability, the active software components of the architecture (or *peers*) should be light-weight and runnable on several platforms (from desktop workstations to embedded systems), without increasing applications complexity. Support

for different robot programming methods (on-line, off-line, hybrid) should be provided, along with transparency (*i.e.*, all accessible applications should appear as if they were local).
Moreover, modern industrial systems need to be reconfigurable, able to produce different parts, depending on which machines are used and in which order. Control software and physical resources, like sensor and robot controllers, need to be easily connected/disconnected to the systems. Hence, **support for multiple clients** is required to system devices, along with suitable **authentication and concurrency protocols** to ensure safe access according to prescribed policies and priorities. Control actions, in fact, may require or be entrusted with different access levels: from simple sensor monitoring, to robot control, to system supervision.
Servers should ensure **real-time operation** to allow implementation of the appropriate control laws with guaranteed operation. Moreover, they should accept and manage requests preserving their ordering, and exhibit differentiated reactions depending on their urgency. An industrial robotic system should also provide the guarantee of correct execution priorities of application tasks at the server. Consistent, end-to-end propagation of priority semantics is especially difficult when heterogeneous peers, with different operating systems and programming languages, must be accommodated.
Control of several devices and sensors in a networked environment requires **concurrency in server operations** and a **multithreaded server structure**. This capability enables execution of parallel independent or coordinated actions in the target workcell, improves the response time, and simplifies CPU sharing among computational and communication services. Portable thread synchronization mechanisms should therefore be available to achieve reliable and efficient concurrency in servers.
The system should allow **asynchronous (non-blocking) client requests** to servers controlling networked devices. This feature is not always required for simple applications consisting of a limited number of distributed requests, possibly operated in stop-and-go mode. However, it becomes necessary for advanced scenarios where multiple concurrent activities can be invoked at the server. Examples of such tasks are coordinated operation of multiple arms or concurrent sensing and manipulation.
In industrial applications, the timely availability of adequate sensory data is required for plant control, production monitoring, remote maintenance, data logging, and post-production analysis. **Efficient and scalable data distribution** techniques must be available to return sensory information to a dynamic and possibly large set of data consumers.
Finally, while early approaches to distributed robotic systems were based on development of in-house solutions, new industrial applications take advantage of the availability of standard infrastructures and technologies. In this way, the total cost of ownership and management of the architecture controlling the robotic system can be reduced thanks to the adoption of **component-off-the-shelf (COTS)** and **component-based design**. Approaches promoting **software reuse**, such as developing a common framework and exploiting COTS, have the potential of drastically reducing maintenance and integration costs.

3. Distributed Middlewares

A distributed middleware is a connectivity software designed for the interaction of software components in complex, distributed applications. During the 90's, this technology has greatly evolved to provide interoperability and application portability in client/server architectures. A distributed middleware, however, has emerged as an enabling technology also for distributed industrial robotic applications, requiring the ability to interconnect multiple and possible heterogeneous platforms while ensuring location transparency and extensibility. Among the most prominent actors in the distributed middleware arena are Web Services and CORBA technologies.

3.1 Web Services

Service-oriented computing [Bichier et al., 2006] defines an architectural style whose goal is to achieve loose coupling among interacting software entities. In this context, Web Services [W3C, 2004][Motahari Nezad et al., 2006] provide a standard, simple and lightweight mechanisms for exchanging structured and typed information between services in a decentralized and distributed environment. The main goal of Web Services is to allow machine-to-machine interaction, whereas traditional Web applications are human-to-human oriented. Relevant specifications and standards for Web Services include *eXtensible Markup Language* (XML), *Simple Object Access Protocol* (SOAP), the communication protocol, *Web Service Description Language* (WSDL), a machine-processable format to describe service interfaces, and *Universal Description, Discovery and Integration* (UDDI), centralized or distributed repositories listing service interfaces. We refer to the cited literature for the full development of these concepts.

A Web Service contract describes provided functionalities in term of messages. By focusing solely on messages, the Web Service model is completely language, platform, and object model-agnostic. As long as the contract that explains service capabilities and message sequences and protocols it expects is honoured, the implementations of Web Services and service consumers can vary independently without affecting the application at the other end of the conversation.

The use of Web Services is rapidly expanding driven by the need for application-to-application communication and interoperability. The question thus is whether Web Services are the right middleware for robotics as well. In our view, while the generality and reuse capabilities of Web Services are appealing for robotic applications as well, they fail to provide the performance guarantees for end-to-end system operation required in robotic applications.

3.2 CORBA and Other DOC Middlewares

Several Distributed Object Computing (DOC) middleware standards are available, notably the Common Object Request Broker Architecture (CORBA) [OMG A, 2004] promoted by the Object Management Group (OMG), and Microsoft's Distributed Component Object Model (DCOM). The distributed objects concept is an important paradigm, because it is relatively easy to hide distribution aspects behind an object's interface. Furthermore, since an object can be virtually anything, it is also a powerful paradigm for building systems.

DCOM's major design goal appears to be providing improved functionality while staying compatible with previous versions incorporated in the early Windows systems. Unfortunately, DCOM has proven exceedingly complex, especially considering its proprietary nature. Today, DCOM has been largely superseded by its successor .NET, which however remains a proprietary solution mainly restricted to Microsoft operating systems.

When language, vendor, and operating system independence is a goal, CORBA is a mature solution that provides similar mechanisms for transparently accessing remote distributed objects while overcoming the interoperability problems of .NET. CORBA is the result of an effort to provide a standard middleware platform to allow applications from many different software manufacturers to interoperate. Implementing a distributed architecture with

CORBA allows smooth integration of heterogeneous software components. To ensure portability, reusability, and interoperability, CORBA architecture is based on the Object Request Broker (ORB), a fundamental component that behaves as a system bus, connecting objects operating in an arbitrary configuration (Figure 1).
To achieve language independence, CORBA requires developers to express how clients will make a request to a service using a standard and neutral language: the OMG Interface Definition Language (IDL). After the interface is defined, an IDL compiler automatically generates client stubs and server skeletons according to the chosen language and operating system. Client stub implementation produces a thin layer of software that isolates the client from the Object Request Broker, allowing distributed applications to be developed transparently from object locations. The Object Request Broker is in charge of translating client requests into language-independent requests using the Generic Inter-ORB Protocol (GIOP), of communicating with the Server through the Internet Inter-ORB Protocol (IIOP), and of translating again the request in the language chosen at the server side.

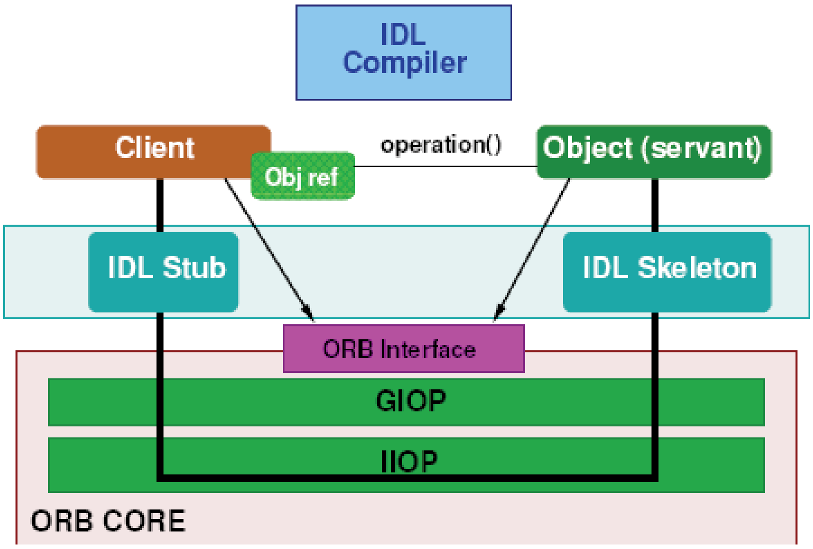


Fig. 1. CORBA ORB architecture.

Together with the Object Request Broker, the architecture proposed by OMG introduces several CORBA Services, providing capabilities that are needed by other objects in a distributed system. We refer to the cited literature [OMG A, 2004] for additional information on general features of the CORBA Standard. The most advanced CORBA implementation, in terms of compliancy, is The ACE ORB (TAO) [Schmidt et al., 1998]. TAO implements all the QoS-oriented parts of the specification. For example, the Real-time CORBA 1.0 [OMG C, 2002] standard supports statically scheduled Distributed Real-time and Embedded (DRE) systems (such as avionics mission computers and industrial process controllers) in which task eligibility can be mapped to a fixed set of priorities. Starting from version 1.5, TAO supports the Real-time CORBA 1.2 specification [OMG D, 2003] which addresses limitations with the fixed-priority mechanisms by introducing two new concepts in Real-time CORBA: *distributable threads*, that are used to map end-to-end QoS requirements to distributed computations across the endsystems they traverse, and a *scheduling service architecture*, that allows applications to enforce task eligibility.

3.3 Choosing CORBA in Industrial Robotic Applications

The decoupling and interoperability features offered by the message-oriented approach characterizing Web Services are of course interesting for the industrial robotics scenario, where mobile robots and server applications often run on different platforms. However, XML processing is too "heavy" to meet the real-time constraints arising in industrial robotic applications. Several researchers have experimented with performance improvements to Web Services, such as faster parsers [Bayardo et al., 2004], data compression [Liefke, 2000], protocol optimizations [Chiu et al, 2002], or binary encodings [Tian et al., 2004], but the outcome generally depends on the structure of the data used and the application itself.

Today, CORBA is in wide use as a well-proven architecture for building and deploying significant enterprise-scale heterogeneous systems, as well as DRE systems. In these sectors CORBA is growing because it can harness the increasing number of operating systems, networks, and protocols supporting real-time scheduling into an integrated solution that provides end-to-end QoS guarantees to distributed object applications. Research over the past several years has identified key architectural patterns and performance optimizations required to build high-performance, real-time ORBs. This effort resulted in the development of prototype real-time ORB endsystems that can deliver end-to-end QoS guarantees to applications [Gill et al., 2004], [Schantz et al., 2006], [Zhang et al., 2005], [DiPippo et al., 2003].

CORBA's most advanced and recent features (Real-Time CORBA, AMI) provide functionalities almost essential in application domains sharing critical concerns such as industrial automation, avionics, and telecommunications. Based on the features discussed in the next section, CORBA seems the logical choice for building complex distributed robotic applications, thereby satisfying the interoperability, transparency, COTS availability, component-based design, and software reusability requirements previously highlighted.

4. CORBA Features for Distributed Robotic Applications

This section discusses CORBA features introduced with version 2.4 of the Standard, along with other CORBA Services relevant to distributed robotic systems. Real-Time CORBA 1.0, Messaging and several consolidated services are evaluated by means of simple benchmarks designed to assess their suitability for distributed robotic applications. Real-Time CORBA 1.2 is not considered here because of its draft status.

4.1 Concurrency Among Clients

A requirement of advanced distributed robotic systems is to manage input from all controlling devices while generating a single and coherent control sequence for each robot server, allowing collaborative and coordinated operation [Chong et al., 2000].

As a basic functionality, server applications must ensure atomicity of calls to library functions devoted to the interaction with the robot controller, regardless of the server thread-safety. Potential conflicts arising from multiple clients can be avoided forcing an exclusive access to library functions through the RTCORBA_Mutex construct, implementing the mutual exclusion lock. The server side is solely responsible for the implementation of this functionality, since Mutexes are introduced in the servant code.

In addition to ensuring single command consistency and atomicity, concurrent access control at session level must be implemented to guarantee full robot control without undesired interferences from other operators. Implementation of a coordination scheme allowing multiple clients to control a single robot through a coherent and logically safe pattern of interaction can be obtained exploiting the CORBA **Concurrency Control Service** [OMG, 2000]. This service allows several clients to coordinate their concurrent accesses to a shared resource so that the resource consistent state is not compromised. Of course, it is up to the developer to define resources and identify situations where concurrent accesses to resources conflict.

For these purposes, the Concurrency Service provides the *lock* as the basic coordination mechanism. Each shared resource should be associated with a lock, and a client must get the appropriate lock to access a shared resource. Several lock modes are defined, allowing different conflict resolution policies among concurrent clients. We show how the CORBA Concurrency Service can be exploited by means of an example. This pattern can be adapted to several situations involving conflicting reading and writing access rights to one or more devices across the distributed system.

In a scenario where several clients compete to control a single robot and/or access data from multiple sensors, exclusive control of the robot must be granted only to one client in a given interval of time, while simultaneous read of sensor data should be allowed to other consumers as well. For each robot, a Robot and a RobotStatus objects are created. The RobotStatus class maintains information about a robotic device, whereas the Robot class controls movements and sets parameters. Then, for each Robot object, a lock object is created and registered in the Naming Service.

As shown in Figure 2 (scenario 1), the client invoking commands on the Robot object holds an exclusive lock ensuring exclusive control. Indeed, as the exclusive lock conflicts with any other lock, a client requesting a lock on the same resource will be suspended waiting its release. Clients invoking methods on the RobotStatus object, instead, are not required to hold locks as the class has only ''accessor'' methods.

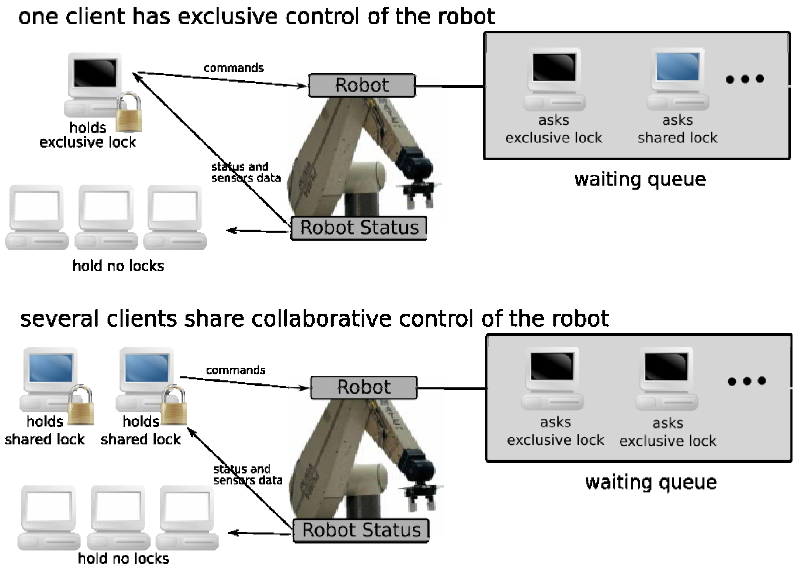


Fig. 2. Two scenarios of concurrent access to a single robot device from several clients.

To preserve generality and cover a wider domain of applications [Chong et al, 2000], an alternative scenario can be outlined, where a group of clients wants to control a single robot

in a collaborative way (*e.g.*, in a telerobotic application, a ''primary'' operator with some ''backup'' operators), while preventing further clients from obtaining exclusive control of the robot. In this scenario (Figure 2, scenario 2), a collaborating client holds a shared lock. Since the shared lock conflicts only with exclusive locks, it allows sharing of robot control with other clients holding shared locks, whereas clients requesting exclusive control through an exclusive lock are suspended in the exclusive access queue.
An experimental evaluation of the access control mechanisms outlined above for multiple concurrent clients is reported in Section 5.

4.2 Server-side Concurrency and Real-Time Guarantees

Control of several robots and sensors teleoperated from multiple remote clients requires one or more multithreaded servers allowing concurrency among actions. Moreover, servers should be able to discriminate among services, granting privilege to critical tasks (emergency stop, reading of safety sensors), and should avoid priority inversion (low-priority tasks blocking high-priority ones).
With the Real-Time CORBA specification, OMG released a set of standard CORBA APIs for multithreading, thereby avoiding the use of proprietary ORB features to program multithreaded real-time systems. The core mechanism of RT CORBA is the **Thread Pool**, enabling pre-allocation of Server resources. With the Thread Pool mechanism, a group of threads is statically created by CORBA in the server at start-up time. These threads are always ready to be bound to requested methods. To achieve predictability, a maximum number of dynamic threads is available. These threads are created only once static threads are exhausted. The Thread Pool avoids the overhead of thread creation/destruction at run-time and helps in guaranteeing performance by constraining the maximum number of threads in each host.
Under the extreme condition where its whole set of threads has been bound to low-level requests, a server could miss the deadlines of high-priority actions, a situation clearly unacceptable in an industrial robotic system. To avoid depletion of threads by low-priority requests, a Thread Pool can be further partitioned in **Lanes** of different priority. This partitioning sets the maximum concurrency degree of the server and the amount of work that can be done at a certain priority. Partitioning in Lanes and related parameters cannot be modified at run-time; the only freedom is reserved to higher priority methods which can ''borrow'' threads from lower level Lanes once their Lane is exhausted.
Servers in distributed robotic applications should also provide clients with the guarantee of correct execution priorities of application tasks. However, heterogeneity of nodes, in distributed applications, precludes the use of a common priority scale. In early CORBA versions and other middleware approaches, this problem forced users to concern about low-level details of threads on different OSes. Real-Time CORBA has solved the problem by providing a **priority mapping** mechanism converting CORBA priority levels, assigned to CORBA operations, to OS native priority levels (and vice versa).
Critical distributed robotic applications also require task execution at the right priority on the Server side. RT CORBA defines two invocation models: SERVER_DECLARED, in which objects are created with assigned execution priority, and CLIENT_PROPAGATED, in which the Client establishes the priority of the methods it invokes, and this priority is honoured by the Server. Thanks to the Thread Pool and Priority features, a server based on RT CORBA can thus guarantee real-time execution of critical computations and achieve coherent, end-to-end priority semantics.

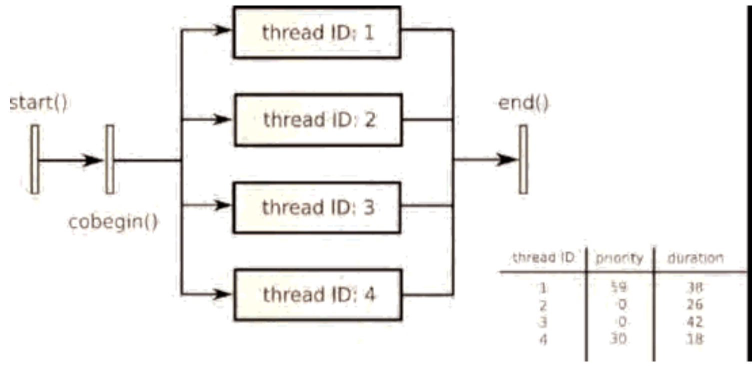
Another client mechanism enabling to take advantage of available concurrency in the server has been introduced in the CORBA 2.3 Messaging Specification. Standard service requests in CORBA systems rely on the Synchronous Method Invocation (SMI) model, that blocks the client until the server notifies the end of the requested activity. Non-blocking invocations with earlier CORBA versions either relied on methods not guaranteeing the delivery of the request or on techniques requiring significant programming efforts and known to be error prone [Arulanthu et al., 2000].

The **Asynchronous Method Invocation (AMI)** model [OMG A, 2004] provides a more efficient way to perform non-blocking invocations with either a Polling or a Callback approach. The AMI interface allows a CORBA-based system to efficiently activate multiple concurrent actions at a remote teleoperated site. Moreover, as AMI and SMI share the same object interface, clients can choose between synchronous or asynchronous calls, whereas server implementation is not affected.

The AMI interface, however, cannot guarantee that a set of parallel actions will be actually executed at the "same" time. When a server receives non blocking requests from a client, it dispatches them to the Thread Pool according to their priorities and they are started as soon as possible. Due to the communication delays of the distributed system, requests will reach the server at different and unpredictable times. Synchronized parallel action execution is thus unlikely. This behaviour is not satisfactory for many robotic applications, where a set of parallel actions must often begin at the "same" time and coordination of their execution is required to ensure logical correctness or safety. This problem can be solved by implementing a *waiting rendezvous* strategy [Douglass, 1999] based on the available CORBA mechanisms. We have developed an implementation where an instruction termed cobegin(n) prefixes the invocation of parallel actions in a server, acting as a barrier for the next *n* method invocations, whose execution, therefore, does not start until all calls have reached the server. Without cobegin(n), the server schedules actions invoked by AMI requests as soon as they arrive.

4.2.1 Experimental Evaluation

In the experiment reported in this section, a server application controls a manipulator and the sensors which are directly related to the manipulator. The Thread Pool is set up to include three Lanes (low, medium, and high priority). Low and medium priority Lanes supply threads for the execution of requested actions. The high-priority Lane supplies threads for emergency actions, so as to guarantee their immediate dispatch. The scheduling algorithm is a Priority Level Round Robin (SCHED_RR), which is available in any POSIX-compliant operating system.



thread ID	priority	duration
1	59	38
2	0	26
3	0	42
4	30	18

Fig. 3. Precedence graph of a concurrent task, consisting of an initial action start(), followed by four concurrent actions with different priority.

Experiments involving simulated workload have been carried out to evaluate the correctness and robustness of the server, which has been tested with a variety of sets of concurrent actions, with different priority levels and synchronization requirements. We have also experimented the effectiveness of cobegin(n) in avoiding priority inversion in the execution of parallel actions. One of the experiments is described in Figure 3, showing precedence relations, duration and priority of each method call. The correct outcome of this experiment requires that the four concurrent methods be executed according to their priority. Figure 4 compares two experimental executions of the task. Without cobegin(n) (top diagram), the medium priority action (ID 4), whose request is the first reaching the server, is executed before the high priority action (ID 1). With cobegin(n) (bottom diagram), the priority of threads is always guaranteed and no priority inversion occurs.

4.3 Data Distribution

A data distribution subsystem for networked robot applications should be able to efficiently exchange significant amount of data from the sensors located at the remote site (*suppliers*) to the operators controlling/monitoring the remote environment (*consumers*). Moreover, it should be able to cope with the heterogeneity of consumers, its performance should scale with the number of connections, and it should minimize the load on both sides avoiding polling operations.
These requirements cannot be fulfilled by the classical Remote Procedure Call (RPC) mechanism that is, instead, suitable for transmission of control commands. Indeed, polling operations, required by RPCs, introduce well known saturation effects on both the network, due to the useless flow of requests and responses, and the supplier, unable to answer to a large number of simultaneous consumer requests.

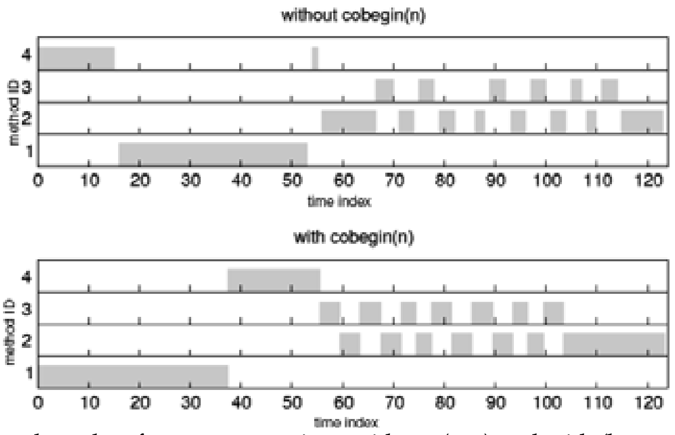


Fig.. 4. Experimental results of concurrent actions without (top) and with (bottom) cobegin(n).

Solutions currently available in CORBA for transmission of time-critical streaming data are still quite limited. A standard for Audio/Video Streaming [OMG B, 2000] is included among CORBA specifications, but it only defines common interfaces for negotiation of the connection among distributed applications, lacking details on its use and control. Due to the weakness of this standard, most existing CORBA implementation do not take advantage of the CORBA Audio/Video Streaming specification to handle multimedia streams. It should be mentioned that OMG has recently defined a new specification to facilitate the exchange of continuous data in the CORBA Component Model [OMG, 2005].

A data distribution technique alternative to Audio/Video Streaming, less efficient but more portable, adopts a Publisher/Subscriber communication model [Buschmann et al., 1996]. Whenever the Publisher (sensor) changes state, it sends a notification to all its Subscribers. Subscribers in turn retrieve the changed data at their discretion. OMG introduced two variants of the Publisher/Subscriber communication model in the CORBA standard, namely the *Event* and *Notification Services*, that strongly decouple Publisher and Subscribers by means of an ''Event Channel.'' Exploitation of these services for data distribution is investigated next.

Event Service. The CORBA Event Service [OMG B, 2004] allows suppliers and consumers to exchange data without requiring the peers to know each other explicitly. The general idea of the Event Service is to decouple Suppliers and Consumers using an *Event Channel* that acts as a *proxy consumer* for the real suppliers and as a *proxy supplier* towards the real consumers. Therefore, the supplier can perform a non blocking send of sensory data in the Event Channel, while the interested consumers can connect to that channel to get the ''event'' (Figure 5). This implementation also allows a transparent implementation of the broadcast of sensory data to multiple consumers. The CORBA standard proposes four different models interleaving active and passive roles of suppliers and consumers. We discarded models with active consumers as they can produce blocking communications when new data are not available at sensor proxy. For distributed robotic applications, the most appropriate model seems to be the Canonical Push Model, where an active supplier pushes an event towards passive consumers registered with the Event Channel.

Despite the benefits introduced by an Event Channel, experimenting with this Service highlights several issues. Firstly, to avoid compile-time knowledge of the actual type of the ''event,'' sensor data must be communicated as an OMG IDL "any type" data type. The communication is therefore type-unsafe and consumers are charged with the duty of converting between the "any type" and the data type they need. Secondly, the Event Service specification lacks event filtering features: everything is conveyed through the Event Channel, that in turn sends everything to any connected consumer. Therefore, the load of a missing property is laid on consumers, that are forced to filter the whole data set looking for the ones they really need. Moreover, the flow of unrequested data can again introduce the problem of network saturation. Finally, the Event Service specification does not consider QoS properties related to priority, reliability, and ordering. Attempting to ensure these properties in an application results in proprietary solutions that prevent ORB interoperability.

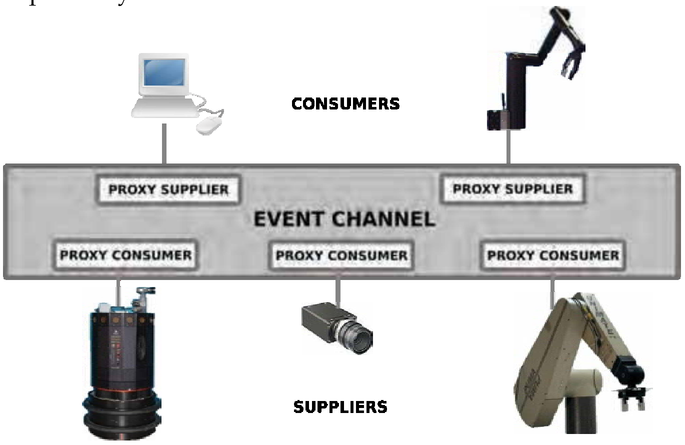


Fig. 5. CORBA Event Service architecture.

Notification Service. A second solution investigated for the data distribution subsystem is based on the CORBA Notification Service [OMG C, 2004], which has been introduced in the CORBA Standard to overcome the limitations of the CORBA Event Service.
The Notification Service is essentially a superset of the Event Service; most components of the Event Service architecture (Figure 5) have been enhanced in the Notification Service. Notable improvements with respect to the Event Service include filtering and QoS management. In the Notification Service each client subscribes to the precise set of events it is interested in receiving through the use of filter objects, encapsulating one or more constraints. Two filter types are defined: a *forwarding filter*, that decides whether the event can continue toward the next component, and a *mapping filter*, that defines event priority and lifetime. Moreover, QoS properties for reliability, priority, ordering, and timeliness can be associated to a channel, to a proxy, or to a single event.

4.3.1 Experimental Evaluation

The communication models described in the previous section have been evaluated in a CORBA-based robotic application requiring to distribute sensory data to a number of clients. The experiments reported in the following assess the relative performance in terms of latency and scalability of the two proposed data distribution mechanisms. All experiments reported in this section follow the Push Model: a supplier generates data and sends them to the Event Channel, when available, or directly to Consumer processes. A single supplier and one or more consumers, all requiring the same sensory data, are considered. Two hosts have been used, one for the supplier, and one for the consumer(s). The experiments have been performed with the Event Channel located either on the machine which hosts the supplier (low workload), or on the other machine (high workload). The main features of the exploited machines are listed in Table I. The hosts are connected via Fast Ethernet switches, with different bitrates. Unless stated otherwise, the network had no significant traffic, nor was any other processing taking place on these hosts.

Host name	Hardware configuration	Operating system	Network connection
Trovatore	Intel P4 2.4 GHz, 512 MB RAM	Gentoo 1.6.14	1Gbps
Malcom	AMD Athlon 64 3500+, 1 GB RAM	Gentoo 1.6.13	1Gbps
Tromba	Intel P4 2.0GHz, 512 MB RAM	Kubuntu 5.10	100 Mbps

Table I. Features of the machines hosting the data distribution tests.

In the first set of experiments, the supplier is hosted by Trovatore and pushes 20000 packets (64 Bytes per packet). Minimum, average, and standard deviation (jitter) values of interarrival times are evaluated. Consumer activity is limited to the update of the latency value so far. We measured the average time interval between two successive 64 Byte packet receptions (interarrival time) increasing the number of Consumers from 1 to 150 (hosted by Malcom). Event and Notification Services results are compared with those of a Distributed Callback implementation based on the Observer patter [Gamma et al., 1995], *i.e.*, a pattern where new sensor data, when ready, are sent by the supplier application to all consumers previously registered as Observers (see Figure 6).

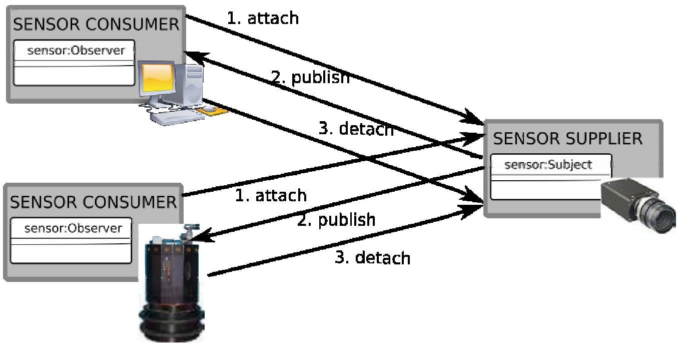


Fig. 6. The Distributed Callback mechanism.

Figure 7 shows the results for five solutions: Distributed Callback, Event Service with channel on Malcom with the consumers (A) , Event Service with channel on Trovatore with the supplier (B), Notification Service with channel on Malcom (A), Notification service with channel on Trovatore (B). With less than 10 consumers, the five solutions are equivalent. With more than 10 clients, ES(A) and NS(A) perform better than Distributed Callback, ES(B), and NS(B). That is, the Event and Notification Services show better performance if they are located on the same machine which hosts the consumers, because this setting reduces network congestion since the only remote connection is the one between the supplier and the event channel, while the connections between the channel and the consumers are local to the machine which hosts them. Focusing on this solution, for more than 100 consumers the Notification Service seems to be more efficient than the Event Service.
The second set of experiments investigates synchronization among consumers on the reception of data packets. Figure 8 shows the average interarrival time for two consumers when they are on the same machine (Malcom, Test 1), and when one of them is on a machine slower than the one hosting the other consumer (Tromba and Malcom, Test 2). In TAO default configuration, the Event Service pairs together the two consumers, forcing the packet reception rate to the one permitted by the slower one (compare Test 1 ES with Test 2 ES). The same result is obtained with the Notification Service in the default single-threaded configuration (compare Test 1 NS with Test 2 NS). Not surprisingly, if the Notification Service is configured to deploy a thread at each proxy supplier, the overall performance is improved, and Test 2 gives better results than Test 1 (label MT stands for MultiThreaded).

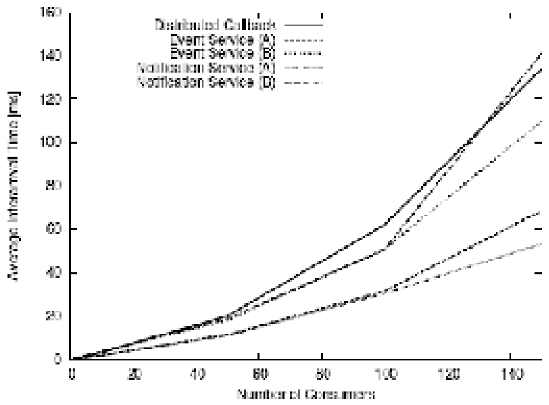


Fig. 7. Interarrival times resulting from different data distribution architectures.

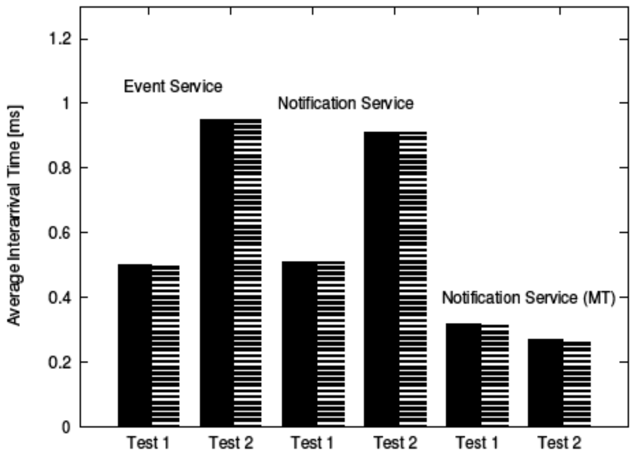


Fig. 8. Interarrival times resulting by three different Publisher/Subscriber models. Test 1 involves two consumers with similar characteristics; in Test 2 one of the consumers (dashed bar) runs on a slower machine.

To summarize, for robot servers performing several complex tasks (*e.g.*, sensor data acquisition and distribution, motion planning, actuator control) and dealing with a large number of attached clients, the Event Channel (offered by the Event and Notification Services) represents an effective tool to maintain the overall workload under control. When consumers have different QoS needs and at the expense of a slight overhead, the Notification Service is the most appropriate solution, thanks to its configurability options not available in Distributed Callback and Event Service implementations.

5. A CORBA-based Telerobotic System

In the previous section, CORBA services relevant to distributed robotic applications have been introduced and evaluated using "microbenchmarks", *i.e.* test suites conceived for the specific service under investigation. The aim of this section is to describe and assess a complete telerobotic application which encompasses many features typical of distributed industrial robotic applications. The telerobotic application has been developed exploiting a framework which integrates various CORBA services.

5.1 The Application

The telerobotic system is illustrated in Figure 9. The remote site includes an Unimation PUMA 560, a six d.o.f. arm whose controller is interfaced to a Pentium-based workstation running the RCCL/RCI robot programming and controlling environment. Mounted on the arm is a parallel gripper, pneumatically actuated with a maximum opening of 80 mm; the gripper is provided with a binary sensor able to detect its closure. The system features additional sensoriality, including a stereo vision system in front of the robot workspace. Clients are connected to the remote robot either with a local (Ethernet LAN) or a geographical network.
To allow the clients to interact with the remote environment, program and monitor the task, each client is provided with a set of applications. Using a graphical user interface (GUI) the

operator can dynamically configure its environment, adding new sensoriality and moving available hardware. The inset graph in Figure 10 shows the GUI with windows displaying sensory information received from the stereo camera.
An advanced client station also supports the control of the robot using an 18-sensor CyberTouch (a virtual reality glove with tactile feedback form Immersion Corp, Inc.) and a six degree of freedom Polhemus tracker (Figure 10).

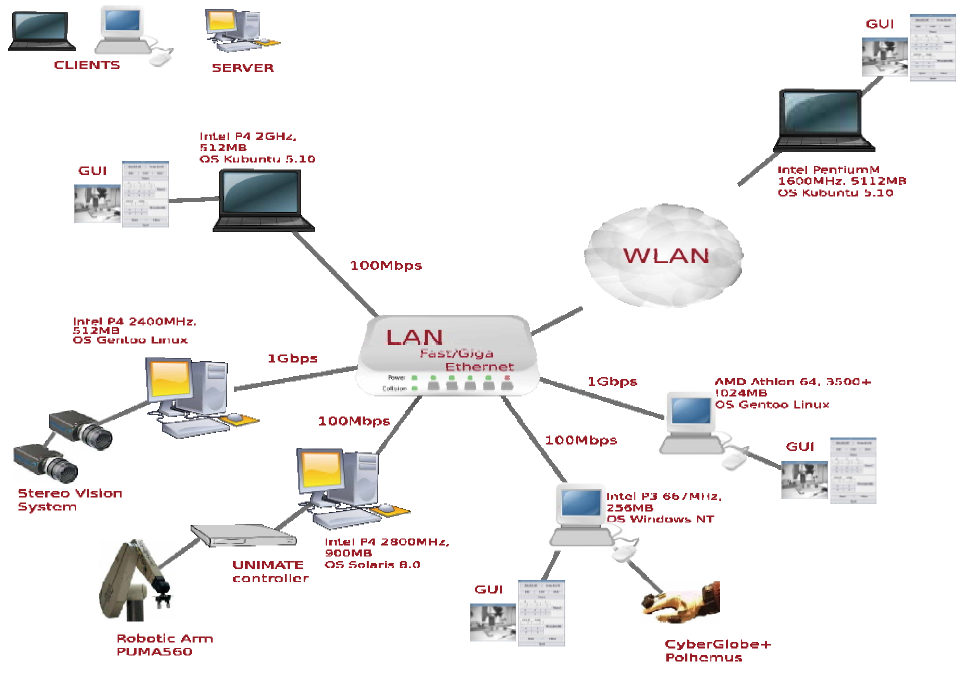

CLIENTS
SERVER
GUI
Intel PentiumM 1600MHz, 5112MB OS Kubuntu 5.10
Intel P4 2GHz, 512MB OS Kubuntu 5.10
GUI
100Mbps
WLAN
Intel P4 2400MHz, 512MB OS Gentoo Linux
1Gbps
LAN Fast/Giga Ethernet
Power
Collision
1Gbps
AMD Athlon 64, 3500+ 1024MB OS Gentoo Linux
100Mbps
100Mbps
GUI
Stereo Vision System
Intel P3 667MHz, 256MB OS Windows NT
UNIMATE controller
Intel P4 2800MHz, 900MB OS Solaris 8.0
GUI
Robotic Arm PUMA560
CyberGlobe+ Polhemus

Fig. 9. The experimental testbed for a CORBA-based telerobotic application.

5.2 Performance Analysis

The experimental results in this section refer to a peg-in-hole manipulation task, and to a concurrent task requiring four clients to perform a stacking manipulation task. While performing the tasks, the system returned to connected users the visual information generated by a camera shooting the workspace.
The first test was performed both over an Ethernet LAN (with average latency of 0.1 ms), and over a WAN (with average latency of 100 ms). The performance observed in the second case was mainly influenced by the time delay and frame rate of the visual information received from the camera using the CORBA-based data distribution mechanisms integrated in the application. To improve the visual feedback, a data channel based on RTP on top of a UDP/IP connection was set up outside the CORBA infrastructure. While this setting requires that client and server agree on the use of non-CORBA communication, it is currently the only solution for efficient streaming of video images over the Internet, since current CORBA support for video streaming is too limited. The drawback of this solution was of course the lower quality of the image due to the lossy compression.

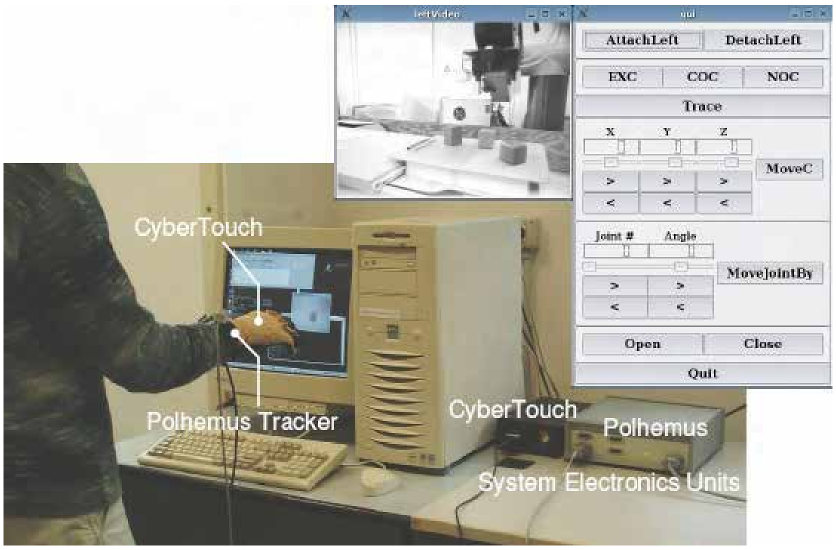


Fig. 10. An example of the client setup, and a snapshot of the GUI with a window displaying sensory information received from the remote site.

Figure 11 shows the Z trajectory of the gripper during two teleoperated peg-in-hole tasks. The solid line shows the gripper height during a task performed in the Ethernet LAN environment. The dashed line shows the same data obtained with the same operator remotely connected to the servers on a high-speed DSL link. In both cases, the GUI was used and proved very useful to ensure effective interaction with the remote system. It can be observed that the task duration was essentially the same for both experiments, even though in the case of teleoperation over LAN the operator was allowed to send more motion commands (65 against 44 of the other case) thanks to the better response time of the system, *i.e.* motion commands (uplink) and sensor data (downlink) were delivered more quickly.

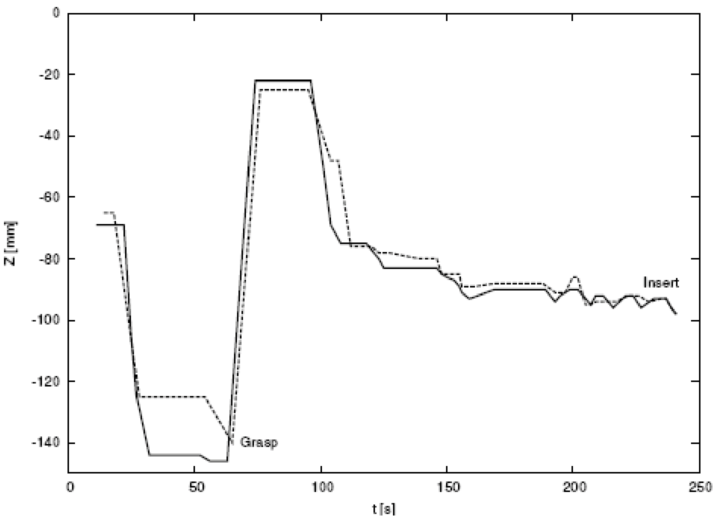


Fig. 11. Z trajectory of the gripper during the peg-in-hole task performed over an Ethernet LAN (continuous line), and over a WAN (dashed line).

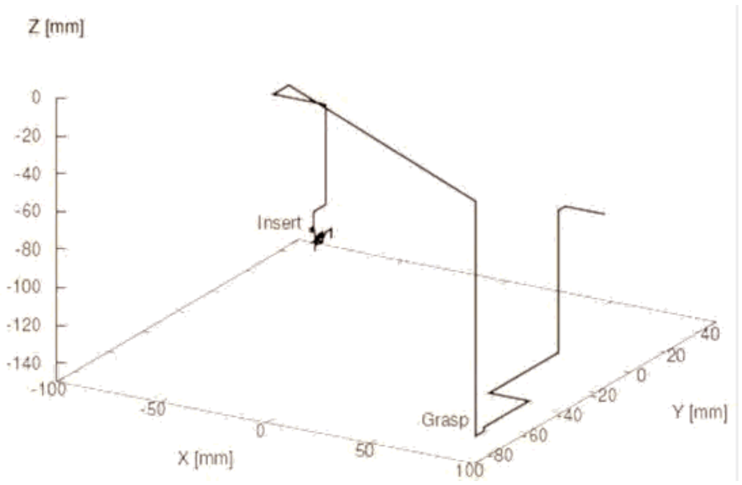


Fig. 12. Trajectory of the gripper in the peg-in-hole experiment (over an Ethernet LAN).

In the second test four clients participated to the execution of a stacking manipulation task (Figure 13). Clients were connected to the robotic server through a Fast Ethernet switch with negligible latency. The main goal of the experiment was to investigate the reliability of the mechanisms for concurrency control and access arbitration among clients described in Section 4.1.

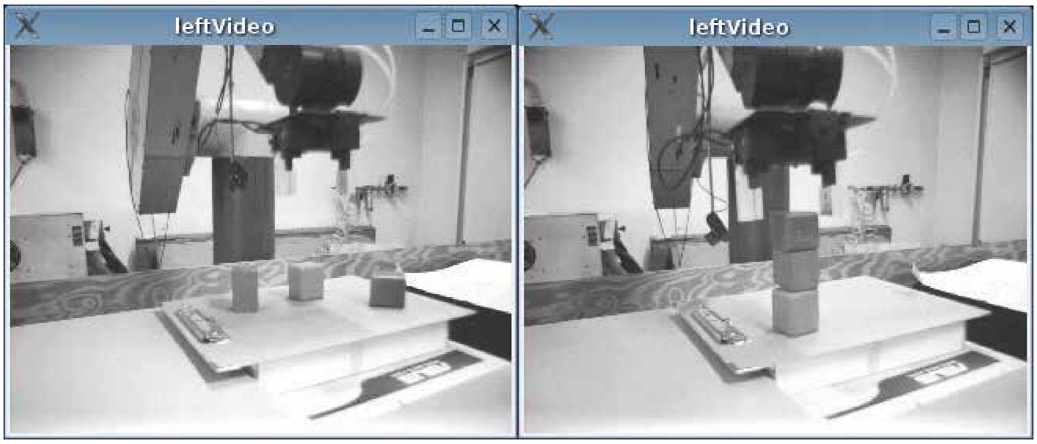


Fig. 13. A stacking experiment with the Puma 560 manipulator. The images are taken from the left eye of the stereo camera.

Figure 14 traces the main environmental values during a task segment. The first row shows the evolution of the distance between the robotic gripper and the table (Z), sampled at 1 Hz, during the execution of the manipulation task. Management of concurrent client access is described by the next four rows of Figure 14.

Each client can be in one of four states: exclusive control, collaborative control, waiting for control, observing. Waiting slots are experienced by clients asking for control when the robot is already controlled in an incompatible mode by other client(s). The bottom row of Figure 14 shows the total number of clients currently connected to the system including slot time when they are connected as ''observer.''

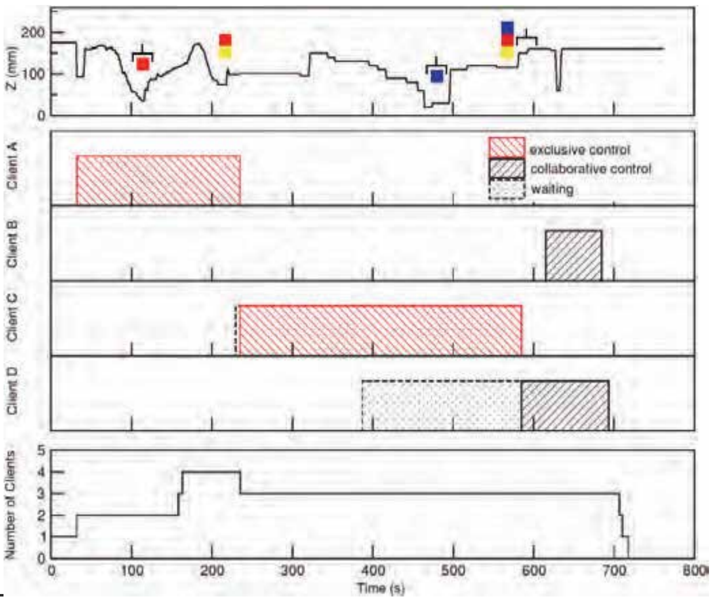


Fig. 14. Tracing of a teleoperation experiment involving four clients that concurrently access the robot arm.

Several task sessions demonstrated the reliability of the CORBA concurrency control mechanisms in providing safe arbitration to critical resources. Tasks similar to the one described were executed, and the presence of multiple clients controlling the robot never prevented their successful completion.

6. Conclusion

The viability and cost effectiveness of new distributed industrial robotic applications can be widely enhanced exploiting COTS-based software components. However, systems implementing those applications must face demanding challenges: they should be dynamically reconfigurable and highly scalable to deal with a potentially large number of peers, they should provide real-time features, guaranteed performance and efficient concurrency mechanisms, both locally and in a distributed environment. The CORBA middleware, especially with its recent extensions, has proven well suited for the needs of many distributed robotic systems. In this paper, we have summarized our experience resulting from the development of distributed robotic applications based on Real-Time CORBA.
Results show that exploitation of CORBA brings a number of remarkable advantages in the distributed robotic domain, enabling portable, highly reconfigurable applications with support for concurrency and real-time features. Furthermore, CORBA standard services for naming resolution, data distribution and concurrency control avoid the need for *ad hoc* solutions, which are often error prone, require significant development effort, and prevent portability.
Of course, providing transparency and high level services in CORBA is not without cost.
Nevertheless, the recent scaling of network and Internet infrastructures to higher levels of performance and QoS guarantees has made less relevant the overhead introduced by CORBA. Drawbacks encountered in our experience stem from the incompleteness in the implementation of the CORBA standard suite. None of the available CORBA ORBs offers a

full implementation of the CORBA standard, *i.e.*, covering aspects such as dynamic scheduling, fault-tolerance, fully compliant CORBA services. Furthermore, some extensions of the standard would be useful, *e.g.*, higher-level synchronization mechanisms (condition variables, barriers) and more effective and useable services for streaming and continuous media management. These deficiencies, however, are widely acknowledged by OMG and in the CORBA standard community, and for some of them solutions are in the making.

7. Acknowledgements

The research described in this chapter has been supported by LARER (Laboratorio di Automazione of Regione Emilia-Romagna, Italy).

8. References

Arulanthu, A.; O'Ryan, C.; Schmidt, D. C.; Kircher, M. & Parsons, J. (2000). The Design and performance of a Scalable ORB Architecture for CORBA Asynchronous Messaging. Proceedings of the ACM/IFIP Middleware 2000 Conference, pp. 208-230, ISBN 3-540-67352-0, New York, NY, USA, April 2000, Springer (Lecture Notes on Computer Science)

Bayardo, R.J.; Gruhl, D.; Josifovski, V. & Myllymaki, J. (2004). An Evaluation of Binary XML Encoding Optimizations for Fast Stream Based XML Processing. *Proceedings of 13th In'l Conf. World Wide Web,* pp. 345-354, ISBN 1581139128, New York, NY, USA, May 2004, ACM Press

Bichier, M. & Lin, K.-J. (2006). Service-Oriented Computing. *IEEE Computer*, Vol. 39, No. 3, pp. 99-101, ISSN 0018-9162

Buschmann, F.; Meunier, R.; Rohnert, H.; Sommerlad, P. & Stal, M. (1996). Pattern-Oriented Software Architecture, Volume 1: A System of Patterns. Wiley and Sons, ISBN 0471958697, New York

Chiu, K.; Govindaraju, M. & Bramley, R. (2002). Investigating the Limits of SOAP Performance for Scientific Computing. *Proceedings of 11th IEEE In'l Symp. High-Performance Distributed Computing,* pp. 246-254, ISBN 0-7695-1686-6, Edimburgh, Scotland, July 2002, IEEE CS Press

Chong, N.; Kotoku, T.; Ohba, K.; Komoriya, K.; Matsuhira, N. & Tanie, K. (2000). Remote Coordinated Controls in Multiple Telerobot Cooperation. *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA 2000)*, pp. 3138-3143, ISBN 0-7803-5889-9, San Francisco, CA, USA, April 2000, IEEE Press.

DiPippo, L.; Fay-Wolfe, V.; Zhang, J.; Murphy, M. & Gupta, P. (2003). Patterns in Distributed Real-Time Scheduling, *Proceedings of the 10th Conference on Pattern Language of Programs 2003*, Monticello, IL, USA, September 2003.

Douglass, B. (1999). *Doing Hard Time: Developing Real-time Systems with UML, Objects, Frameworks, and Patterns*. Addison-Wesley, ISBN 0201498375, Reading, MA

Gamma, E.; Helm, R.; Johnson, R. & Vlissides, J. (1995). *Design Patterns – Elements of Reusable Object-Oriented Software*. Addison-Wesley, ISBN 0201633612, Reading, MA

Gill, C.; Schmidt, D. C.; Krishnamurthy, Y.; Pyarali, I.; Mgeta, L.; Zhang, Y. & Torri, S. (2004). Enhancing Adaptivity Standard Dynamic Scheduling Middleware, *The Journal of the Brazilian Computer Society (JCBS) special issue on Adaptive Software Systems*, Vol. 10, No. 1, pp. 19-30, ISSN 0104-6500

Liefke, H. & Suciu, D. (2000). XMill: An Efficient Compressor for XML Data. *Proceedings of ACM SIGMOD Conference on Management of Data,* pp. 153-164, ISBN 1-58113-218-2, Dallas, TX, USA, June 2000, ACM Press

Motahari Nezad, H. R.; Benatallah, B.; Casati, F. & Toumani, F. (2006). Web Services Interoperability Specifications. *IEEE Computer*, Vol. 39, No. 5, pp. 24-32, ISSN 0018-9162

OMG A (2000). Concurrency Service Specification, Version 1.0, http://www.omg.org/technology/documents/formal/concurrency_service.htm

OMG B (2000). Audio/Video Streams, Version 1.0, http://www.omg.org/technology/documents/formal/audio.htm

OMG (2002). Real-time CORBA 1.0 Specification, http://www.omg.org/docs/formal/02-08-02.pdf

OMG (2003). Review draft of the 1.2 revision to the Real-time CORBA Specification (OMG document realtime/o3-08-01), previously designated Real-time CORBA 2.0, http://www.omg.org/docs/ptc/01-08-34.pdf

OMG A (2004). CORBA/IIOP 3.0.3 Specification, http://www.omg.org/docs/formal/04-03-01.pdf

OMG B (2004). Event Service Specification, Version 1.2, http://www.omg.org/technology/documents/formal/event_service.htm

OMG C (2004). Notification Service Specification, Version 1.1, http://www.omg.org/technology/documents/formal/notification_service.htm

OMG (2005). Streams for CCM Specification, http://www.omg.org/docs/ptc/05-07-01.pdf

Schantz, R.E.; Schmidt, D.C.; Loyall, J. P. & Rodrigues, C. (2006). Controlling Quality-of-Service in Distributed Real-time and Embedded Systems via Adaptive Middleware. To appear in *The Wiley Software Practice and Experience Journal special issue on Experiences with Auto-adaptive and Reconfigurable Systems*, ISSN 0038-0644

Schmidt, D. C.; Levine, D. & Mungee, S. (1998). The Design and Performance of the TAO Real-Time Object Request Broker. *Elsevier Computer Communications Journal, special issue on Building Quality of Service into Distributed Systems*, Vol. 21, No. 4, pp. 294-324, ISSN 0140-3664

Tian, M.; Voigt, T.; Naumowicz, T.; Ritter, H. & Schiller, J. (2004). Performance Considerations for Mobile Web Services. *Elsevier Computer Communications Journal,* Vol. 27, No. 11, July 2004, pp. 1097-1105, ISSN 0140-3664

W3C (2004). Web Services Architecture, *Working Group Note*.

Zhang, J.; DiPippo, L.; Fay-Wolfe, V.; Bryan, K. & Murphy, M. (2005). A Real-Time Distributed Scheduling Service For Middleware Systems. *Proceedings of the 10th International Workshop on Object-Oriented, Real-Time, Dependable Systems*, pp. 59-65, ISBN 0-7695-2347-1, Sedona, AZ, February 2005

Designing and Building Controllers for 3D Visual Servoing Applications under a Modular Scheme

M. Bachiller, C. Cerrada, J. A. Cerrada
Escuela Técnica Superior de Ingeniería Informática, UNED
Spain

1. Introduction

Industrial Robots are designed for tasks such as grasping, welding or painting in where working conditions are well settled. However, if the working conditions change, those robots may not be able to work properly. So, robots require sensor- based control to perform complex operations and to react to changes in the environment. The achievement of such complex applications needs the integration of several research areas in vision and control such as visual matching, visual tracking and visual servoing (Petersson et al. 2002). Information about the actual system and its environment can be obtained from a great variety of sensors. Vision is probably the sensor that provides the greatest and richest sensing information but the processing of such information becomes complicated. Nevertheless, computer vision has been improved a lot in the last years and it is being frequently used in robotics systems, although serious limitations appear in real time applications due to the time necessary for image processing.
The use of computer vision as a feedback transducer strongly affects the closed loop dynamics of the overall system. Latency is the most significant dynamic characteristic of vision transducers and it has many sources, including transport delay of pixels from the camera to vision system, image processing algorithms, control algorithms and communications with the robot. This delay can cause instability in visual closed loop systems. To achieve fast response and high control accuracy the design of a specific visual feedback controller is required.
Visual servoing is the result of merging several techniques in different fields including image processing, kinematics, dynamics, control theory and real time computing. The task in visual servoing is to control a robot to manipulate its environment with the help of vision as a feedback sensor. An excellent overview of the main issues in visual servoing is given in (Nelson & Papanikolopoulos, 1998).
In this work we present a complete form of designing visual servoing controllers for robotic systems built over a modular conception. The designing method is particularly applied to industrial manipulators equipped with a camera mounted on its end effector, known as eye in hand configuration. The modular conception means that the overall system is composed of a set of independent modules, or subsystems, that are put together, configuring a modular system. In this kind of systems any module can be replaced by other one with the same functionality and the visual controller computation procedure will not change. The goal of any visual servoing

system is the same independently of how it is constructed: to control the robot's end effector pose relatively to the target object pose. But the main advantage of our consideration is that if a module has to be changed for any reason it will not be necessary to replace the others or to redesign the full platform, but just to compute a new controller.
This scheme allows us to deal with the problem of controller design from a more generic point of view. We propose a new strategy for designing the control subsystem, which presents advantages as for calculation time, simplicity and estimation errors of object position. One purpose is to demonstrate the possibility of getting competitive results in real time performance with respect to more closed solutions shown in other works and other control configurations, while maintaining the advantage of easy system adaptability.
The remainder of this paper is structured as follows. Firstly, in section 2, we review the relevant fundamentals of visual control arquitectures. In sections 3 and 4, we describe the proposed design methodology and then we analyze the performance of the designed experimental platform to deduce the mathematical models of the involved components. In section 5, we design various control strategies used when the object is static and moving, respectively. After that, we carry out a study of designed controllers in section 6, and we present and discuss some experimental results obtained in the platform in section 7. Finally, conclusions and new tendencies of our work are stated at section 8.

2. Overview of visual control architectures

There are several ways of implementing control schemes when vision is used as a part of the robot position control system. To better understand the different architectures combining control and vision it is previously necessary to analyze how the robotic control system works by itself. A typical robot position control system is shown in figure 1. A system like this is characterized by using a position sensor for each arm joint (Craig, 1989) while the control objective is the Cartesian position of the robot's end effector.
Let $\underline{X}_{ref}$ be the vector denoting the desired Cartesian robot position. For a given reference position $\underline{X}_{ref}$ the corresponding angular reference vector $\underline{\theta}_{ref}$ is obtained by evaluating the inverse kinematics transform of the robot. These joint reference values are the inputs to the joint position controllers of the robot, and there are as many joint controllers as joint drives the robot has. Each controller uses its position sensor to feedback the actual joint angle in an inner control loop. The actual Cartesian position $\underline{X}$ is obtained from the actual joint values $\underline{\theta}$ by evaluating the direct kinematics transform, which depends on the robot geometry.

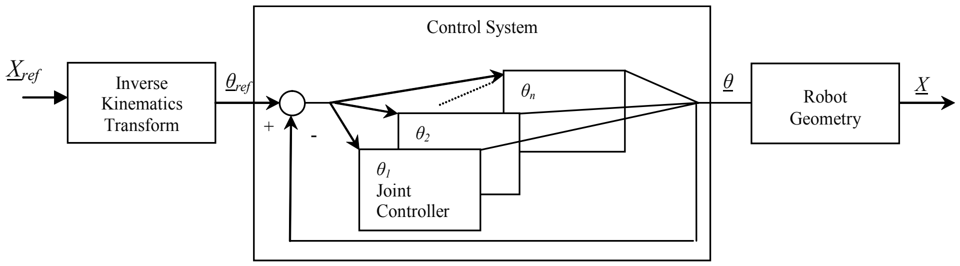


Fig. 1. Block diagram of a robotic control system.

Notice that this scheme represents an open loop control strategy for the Cartesian position because some permanent error can exist between the desired robot position $\underline{X}_{ref}$ and the actual robot position $\underline{X}$. External feedback is necessary to close an outer loop in order to compensate

such error. Visual sensors represent an excellent option to measure and feedback actual robot position in this loop and help to reduce positioning errors. The only drawback of using vision as position sensor is the big amount of computing time required to obtain it.
Earliest applications combining computer vision and robot control where implemented under an open loop scheme, known as **static look and move** architecture. Figure 2 shows a control structure of this type that works by sequentially executing three independent steps (Weiss, 1984). The step by step sequence is repeated until the precision required by the task is achieved. Therefore, the number of necessary iterations depends on the particular task. Let us remark that the nature of this robot positioning process is completely iterative and sequential.

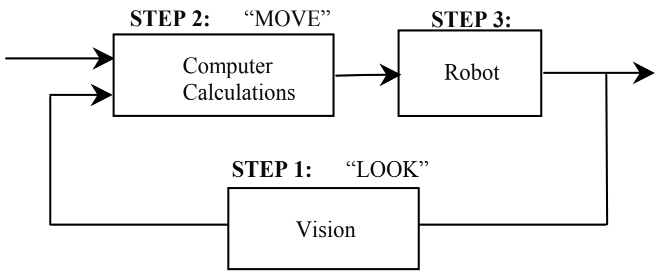


Fig. 2. Static look and move control architecture.

To implement the three steps in parallel in a dynamic control system it is necessary to close the loop by using visual information in the feedback, such as the figure 3 shows. This way of functioning is known as **visual feedback** and the different types of systems that implement it are generically classified as visual servoing architectures.
The main advantage of visual feedback is that it offers much faster dynamic responses than static look and move but, on the other hand, several design problems appear. First, a specific controller is required to compensate the dynamics of the plant, in this case the robot. Secondly, unavoidable delays are introduced in the feedback signal generation, what provokes system instabilities. Nevertheless, the most adequate visual control scheme for robot tasks carried out in unstructured environments is visual servoing because it allows to the robot reacting in real time to unexpected changes in the environment.

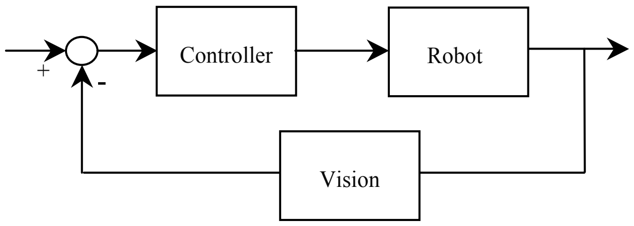


Fig. 3. Generic visual feedback control scheme.

Visual servoing systems typically use one of the two following camera configurations: end effector mounted, often called an eye in hand configuration, or fixed in the workspace. Statically located visual sensors have a very limited working region when taking into account the limited depth of field and spatial resolution that vision sensors typically possess. The working region of a visual sensor can be greatly extended if the camera is allowed to move while tracking the object of interest.
Apart from camera configuration, visual servoing architectures can be differentiated

attending to two main criteria. First, the way of inputting the signal generated by the controller to the robotic system and, secondly, the nature of the visual information supplied by the vision system, i.e. the nature of the controlled variable.
Concerning the first criterion, visual servoing architectures for controlling manipulators can be classified into two fundamental categories: dynamic look and move structures and visual servo structures. When the control architecture is hierarchical and uses the vision system to calculate the set of inputs to the joint level controller, making use of inner joint feedback loops to stabilize the robot, it is referred to as a dynamic look and move structure. A very important advantage of this structure is that it separates the kinematics singularities of the mechanism from the visual controller, allowing the robot to be considered as an ideal Cartesian motion device. In contrast, visual servo structure eliminates the robot controller replacing it with a visual controller in such a way that it is used to compute directly the joints inputs, the loop being the only thing used to stabilize the mechanism.
Concerning the controlled variable, visual servoing systems are classified into two groups: image based control systems and position based control systems. In an image based control system, the error variable is computed in the 2D-image space. This approach eventually reduces the computational delay, eliminates the necessity of image interpretation and eliminates errors due to sensor modeling and camera calibration. However, they require online computation of the image Jacobian. Unfortunately, this quantity inherently depends on the distance from the camera to the target, which in a monocular system is particularly difficult to calculate. Many systems utilize a constant image Jacobian which is computationally efficient, but valid only over a small region of the task space [Hutchinson et al 1996]. On the other hand, in a position based control system, the error variable is computed in the 3D Cartesian space. The main advantage of the last approach is that position of the camera trajectory is controlled directly in the Cartesian space.
Figures 4, 5, 6 and 7 show the schemas of the four possible structures concerning to the combination of the two mentioned criteria, for an eye in hand camera configuration. Notice that visual servo structures require the direct access to the robot operating system in order to govern the joint variables directly, whereas dynamic look and move structures consider the robot as an independent system and can handle it as a black box.
There has been a significant amount of research activity on image based control methods (Weiss et al., 1987), (Feddema & Mitchell, 1989), (Chaumette et al., 1991), (Espiau et al., 1992), (Khosla et al., 1993), (Corke, 1993) whereas there have been only a few researchers working on position based control methods (Koivo & Houshangi, 1991), (Allen et al., 1993), (Wilson et al., 1996), (Vargas et al., 1999). This tendency can be justified because image based systems usually reduce computation delays and eliminate errors due to sensor modeling and camera calibration processes.

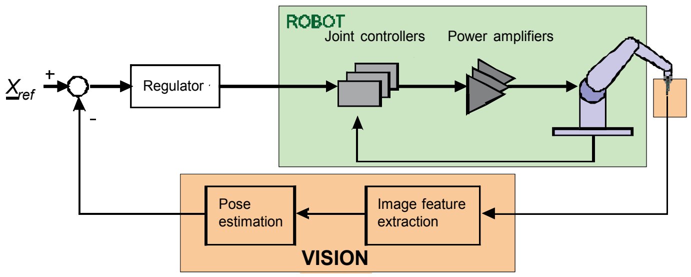


Fig. 4. Dynamic position-based look-and-move structure.

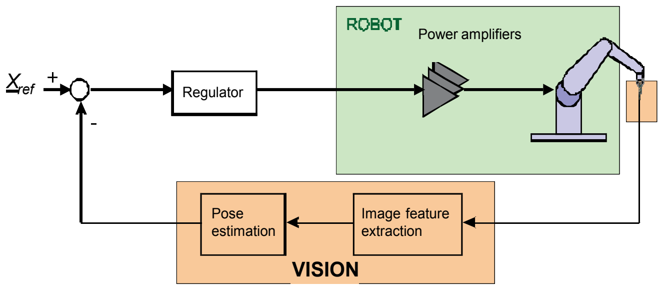


Fig. 5. Position-based visual servo structure.

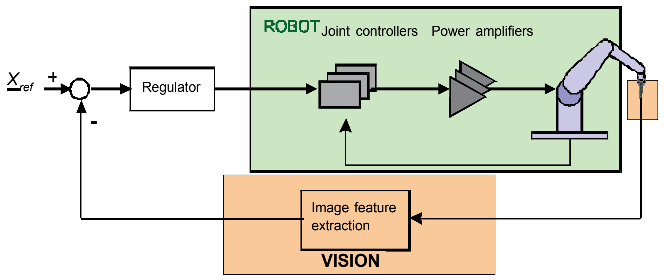


Fig. 6. Dynamic image-based look-and-move structure.

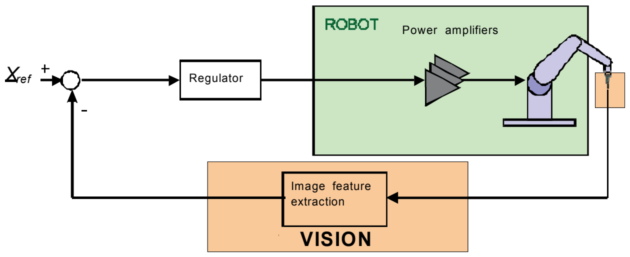


Fig. 7. Image-based visual servo structure.

More recent works (Malis 1999) consider a new visual servo structure that combines the advantages of image based and position based control schemes and avoids some of their disadvantages. This approach, known as 2½D, is characterized by considering the controlled variable composed by two parts, one being represented in the Cartesian space and the other in the image plane. It does not need any 3D model of the object or any precise camera calibration either. However, the method is more sensitive to the existing noise in the image.
Some final considerations can be made concerning the adequacy of the considered visual control architectures to designing and building systems from a modular conception. Having this idea in mind, the most appropriate selection seems to be taking into account position based systems and dynamic look and move structures. Several reasons can be pointed out to think so:

- Position based methods allow a direct and more natural specification of the desired trajectories for the end effector in Cartesian coordinates.
- Many robots have an interface for accepting Cartesian velocity or incremental position commands. In that sense they can be exchanged and all of them can be considered as black boxes.
- The visual controller design of visual servo structures is very complex due to the plant being nonlinear and highly coupled.

3. Modular scheme for visual servoing controllers design

The main idea of the work presented in this paper is to consider the overall visual servoing system formed by a set of independent modules or subsystems connected in a specific way. Taking into account the considerations exposed in the previous section, we have applied this idea to the specific case of designing a position based dynamic look and move system built with an industrial manipulator and an off-the-shelf camera mounted on its end effector. This scheme allows us to deal with the problem of controller design from a more generic point of view. We propose a new strategy for designing the control subsystem, which presents advantages as for calculation time, simplicity and estimation errors of object position. Notice that depending on the type of task we will only have to change the control subsystem specifically designed for this task.

3.1 Preliminary design considerations

First of all, we have used the eye in hand camera configuration in our design proposal keeping in mind its versatility, but this choice does not represent any restriction to the method´s generality. On the other hand, and taking into account that all the studied visual servoing architectures are formed practically by the same functional blocks, we consider that a visual servoing system can be considered as the integration of at least the following set of components (see figure 8):

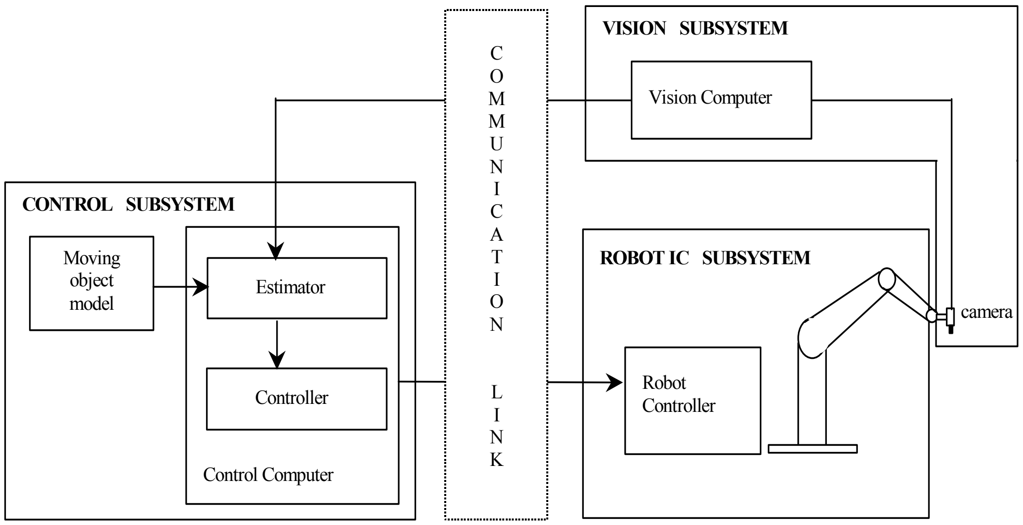


Fig. 8. Subsystems composition of a generic visual servoing system.

1. **Robotic Subsystem**: The robotic subsystem can be any industrial manipulator capable of changing its trajectory in real time by means of specific commands received through a serial line (ALTER line). This command line allows introducing new reference values every T_r milliseconds.
2. **Vision Subsystem:** A camera, the specific hardware and software for image digitalization, and image processing can constitute this subsystem. Both hardware and software can be installed in a vision computer. The function of this subsystem is to measure the actual position of the target object with respect to the robot's end effector in a Cartesian coordinate system every T_v milliseconds.

3. **Control Subsystem**: An estimator (optional) and a controller can form the control subsystem. In principle, this subsystem should be implemented over a specific control computer, but it can also be included in the vision computer, in the robot controller operating system, or both.
4. **Communications**: Communication links between vision and control subsystems and also between control and robotic subsystems form this module.

Notice that the overall system is a multirate system in nature: the sampling interval of the manipulator is T_r while the vision subsystem sampling period is T_v, T_v being larger than T_r. To simplify the study from the control point of view, the vision period can be adjusted to be n times the robot period, n being an integer value, i.e.:

$$T_v = n.T_r \tag{1}$$

For notation concerns, we will use z to represent the z-transform of a system sampled with a period T_v and $\hat{z}$ to the z-transform of a system sampled with a period T_r.

3.2 Fundamentals of the design methodology

The procedure followed to design and evaluate the controllers for the selected visual servoing system can be divided in three well diferenciated phases. The purpose of the first phase is to obtain a mathematical model of the overall system. We call *system model construction* to this phase and it is decomposed into three conceptually different stages: *subsystems modeling*, *system blocks diagram construction* and *identification*. The second phase is properly devoted to design the control subsystem and we call it *control subsystem design*. This design will depend on the type of task to be performed by the system. So, this is the phase where different control techniques can be explored and analyzed. Finally, designed controllers need to be evaluated in order to analyze and compare the performance achieved by each one. All this work is made by means of tests and simulations in the third phase generically called *test results and performance evaluation.*

Apart from the intrinsic value that the comparative study and the performance analysis have by themselves, one of the objectives of the last phase could be to selecting the best controller for each considered task with the purpose of implementing it in the experimental platform. And that is precisely the way we have actuated in this work.

In summary, once the fundamentals and organization of the methodology has been established, the three following sections will explain them more in detail. After that, one additional section will show the real results obtained with the selected controllers built on the experimental platform.

4. First phase: Model System Construction

As it has been mentioned, we are considering a visual servoing system with position based control, dynamic look and move structure and eye in hand configuration. The first phase is to obtain a valid mathematical model for the plant to be governed. In general, to get that model we are using techniques based in conventional control theory, what means that we are going to work with the transfer function of each subsystem and then to build a block diagram of the complete system. In this section we describe the three stages in which we have implemented this phase and the possible alternatives that can be taken into account.

4.1 Subsystems Modeling

We use the architecture shown in Figure 9 which represent to the considered visual servoing system associated to our platform. The four subsystems described in the previous section can be clearly distinguished in this diagram. Nevertheless, several considerations must be taken into account before getting a valid model for each subsystem. First at all, the control subsystem is the objective of the design and consequently it will be considered in the next section. A block named $\underline{R}(z)$ will represent it. With respect to the communication links, they can be considered from the dynamic point of view as small delays that can be integrated in the vision or in the robot time periods. Therefore, no dynamic models are required for them. In summary, it is only necessary to have accurate dynamic models for the sensor used (the vision subsystem) and for the plant to be controlled (the robotic subsystem). These two models are described in next paragraphs.

Vision Subsystem Model

Vision subsystem is in charge of providing a measurement of the 3D object position every sampling period. We have implemented a fast algorithm to estimate object position, based on the simplification of a generic camera calibration model described in (Gonzalez, 1998). This model efficiently compensates non-negligible radial distortion effects introduced by standard lenses, and it has been widely tested in complete 3D object pose measurement. Nevertheless, when object orientation estimation is not required the total on-line computation time can be highly reduced, allowing to achieve smaller T_v values.

Figure 9 illustrates the basic geometry of the considered camera model and robotic environment. This figure also shows the two-point pattern attached to the moving object which is used to estimate its 3D position. Taking away the method for correcting distortion, three main coordinate systems are involved in the model used in this eye in hand configuration:

- $({}^wO, {}^wX, {}^wY, {}^wZ)$: 3D world coordinate system. It is assumed that it coincides with the intrinsic coordinate system associated to the robot, i.e. with the robot coordinate system.
- $({}^cO, {}^cX, {}^cY, {}^cZ)$: 3D camera coordinate system. It is assumed that it is located at the robot's end-effector, i.e., at the actual robot position $\underline{p}_r$.
- $({}^fO, {}^fX, {}^fY)$: 2D image coordinate system. It is the coordinate system of the frame memory where the digitized image is stored and processed.

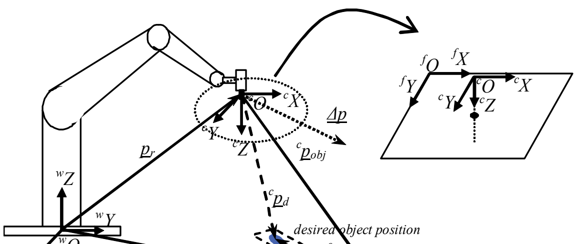


Fig. 9. Basic geometry of the robotic and vision subsystems in the considered configuration.

Let (x_i, y_i, z_i) be the world coordinates of a given point $\underline{p}_i$ located anywhere in the camera workspace, $({}^cx_i, {}^cy_i, {}^cz_i)$ be the camera coordinates of this point, and $({}^fx_i, {}^fy_i)$ be the coordinates of the projection of the point in the image coordinate system. The last two coordinates are related to each other by a (3x4) transformation matrix A which includes the perspective projection elements and the scale factors, as well as the relative translation and rotation between the two coordinate systems when they exist. In short, they are related by the expression:

$$\begin{bmatrix} \omega \cdot {}^f x_i \\ \omega \cdot {}^f y_i \\ \omega \end{bmatrix} = A \cdot {}^c \underline{p}_i = \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \end{bmatrix} \cdot \begin{bmatrix} {}^c x_i \\ {}^c y_i \\ {}^c z_i \\ 1 \end{bmatrix} \tag{2}$$

where the 12 parameters of the A matrix are estimated in the off-line calibration process.
To estimate the object position we take advantage of the known two-point geometry pattern. Both points are black circles in a plane being D the constant distance between them, and being one circle double the size of the other. We assume that the pattern plane always remains perpendicular to the camera optical axis, i.e., the two points are at the same ${}^c z$ coordinate. Let $({}^c x_1, {}^c y_1, {}^c z_1)$ and $({}^c x_2, {}^c y_2, {}^c z_2)$ be the camera coordinates of the big point pattern and the small one respectively in a given position, and let $({}^f x_1, {}^f y_1)$ and $({}^f x_2, {}^f y_2)$ be their corresponding coordinates in the projected image. We will identify the actual object position in camera coordinates with the biggest point position, i.e., ${}^c \underline{p}_{obj} = ({}^c x_1, {}^c y_1, {}^c z_1)$. Let ${}^c \underline{p}_d$ be the desired object position in camera coordinates, i.e. the reference position of object with respect to the camera, which remains constant during the experiment. The actual object position can be determined by solving the next system with five equations and five unknowns:

$$\begin{cases} (a_{11} - a_{31} \cdot {}^f x_1) \cdot {}^c x_1 + (a_{12} - a_{32} \cdot {}^f x_1) \cdot {}^c y_1 = -(a_{13} - a_{33} \cdot {}^f x_1) \cdot {}^c z_1 + (a_{34} \cdot {}^f x_1 - a_{14}) \\ (a_{21} - a_{31} \cdot {}^f y_1) \cdot {}^c x_1 + (a_{22} - a_{32} \cdot {}^f y_1) \cdot {}^c y_1 = -(a_{23} - a_{33} \cdot {}^f y_1) \cdot {}^c z_1 + (a_{34} \cdot {}^f y_1 - a_{24}) \\ (a_{11} - a_{31} \cdot {}^f x_2) \cdot {}^c x_2 + (a_{12} - a_{32} \cdot {}^f x_2) \cdot {}^c y_2 = -(a_{13} - a_{33} \cdot {}^f x_2) \cdot {}^c z_1 + (a_{34} \cdot {}^f x_2 - a_{14}) \\ (a_{21} - a_{31} \cdot {}^f y_2) \cdot {}^c x_2 + (a_{22} - a_{32} \cdot {}^f y_2) \cdot {}^c y_2 = -(a_{23} - a_{33} \cdot {}^f y_2) \cdot {}^c z_1 + (a_{34} \cdot {}^f y_2 - a_{24}) \\ ({}^c x_1 - {}^c x_2)^2 + ({}^c y_1 - {}^c y_2)^2 = D^2 \end{cases} \tag{3}$$

Error vector ${}^c \Delta\underline{p} = ({}^c \Delta x, {}^c \Delta y, {}^c \Delta z)$ is then computed by subtracting the actual position to the reference position, ${}^c \Delta\underline{p} = {}^c \underline{p}_d - {}^c \underline{p}_{obj}$. This vector is equal but opposite to the robot incremental movement $\Delta\underline{p}$ required to cancel such as error: $\Delta\underline{p} = -{}^c \Delta\underline{p} = {}^c \underline{p}_{obj} - {}^c \underline{p}_d$. From figure 1bis it can be seen that ${}^c \underline{p}_{obj}$ is the object position in the camera system and it coincides with $(\underline{p}_{obj} - \underline{p}_r)$ in the world system, i.e. $\Delta\underline{p} = \underline{p}_{obj} - \underline{p}_r - {}^c \underline{p}_d$. Therefore, the vision subsystem provides every sampling period a triplet of values $\Delta\underline{p} = (\Delta x, \Delta y, \Delta z)$ in the world coordinate system representing the position increment that the robot has to be moved to make the target object be at the right position with respect to the camera. To reach those values, every sampling period the vision subsystem must acquire an image of the pattern, digitize it, process it to obtain $({}^f x_1, {}^f y_1)$ and $({}^f x_2, {}^f y_2)$, evaluate and solve the system of equations (3) to obtain ${}^c \underline{p}_{obj}$, and finally compute $\Delta\underline{p} = {}^c \underline{p}_{obj} - {}^c \underline{p}_d$. Because of the required computation time, this information is not available until the next sampling instant. In that sense, when the overall system is working with T_v period, the vision subsystem can be considered as a pure delay. So its transfer functions matrix is:

$$V(z) = diag\,(z^{-1}, z^{-1}, z^{-1}) \tag{4}$$

Figure 10 shows the representative block of the vision subsystem. This diagram also considers a noise signal $\underline{r}_s$ which can be produced, for example, because of bad illumination conditions or just in the digitizing process.

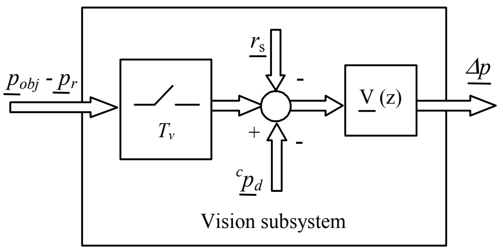


Fig. 10. Representative block of the vision subsystem.

Robotic Subsystem Model

When the manipulator works through the ALTER line it behaves as a decoupled multivariable system. It can be considered as composed of six independent loops (three of them concerning the Cartesian position and the other three concerning the orientation). As it has been mentioned above, in this work we are only taking into account the three loops relative to the Cartesian position. Thus, the manipulator with its actuators and their current feedback loops can be considered as a Cartesian servo device.

In principle, the robotic subsystem can be modeled as an integrator for each Cartesian coordinate: its output is the actual robot position that is equal to the previous one plus the incremental position input signal. From the experimental responses obtained when exciting the used robot with step signals for all the coordinates we have observed that they do not behave as pure integrators, and that some additional dynamic features appear. In order to model this behavior, we consider that the transfer function for each Cartesian coordinate is a second order system with a pole at z = 1 (representing the integrator) and another one at z = β_i which determines the characteristics of the transient response of the robotic subsystem. So, the transfer functions matrix of the robotic subsystem is:

$$\underline{G}(\hat{z}) = diag\left(\frac{n_x}{(\hat{z}-1)(\hat{z}-\beta_x)}, \frac{n_y}{(\hat{z}-1)(\hat{z}-\beta_y)}, \frac{n_z}{(\hat{z}-1)(\hat{z}-\beta_z)}\right) \tag{5}$$

Figure 11 shows the representative block of the robotic subsystem. The sampling time is T_r and $\underline{e}$ is a noise signal that represents small errors due to data acquisition, which will be used in the identification process. That noise is modeled as a sequence of independent and identically distributed random variables with zero mean.

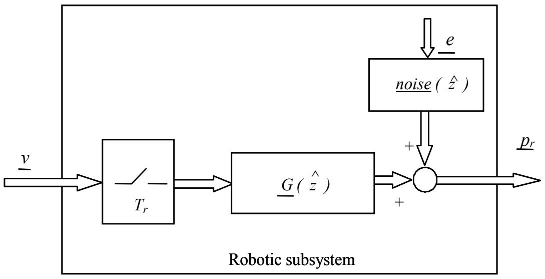


Fig. 11. Representative block for the robot subsystem.

Notice that the input signal to this block (or control signal) can be considered as a velocity reference ($\underline{v}$) since the robotic subsystem must make the specified displacement in T_r milliseconds.

4.2 System´s Block Diagram Construction

If we join all the subsystems in accordance with the position based dynamic look and move structure we obtain a block diagram as shown in figure 12, where the control signal $\underline{v}$ is a velocity reference. For that reason we will refer to this configuration as velocity control scheme.

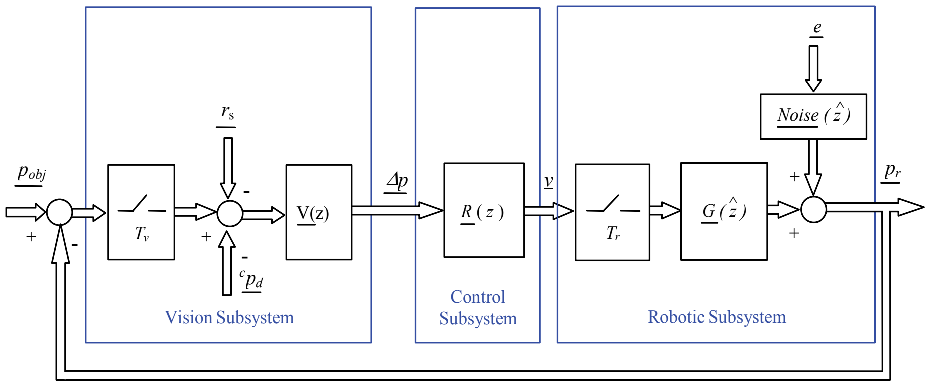


Fig. 12. Block diagram for velocity control.

Using the equation (1) and doing some manipulations this block diagram can be converted into the equivalent one shown in figure 13, where $\underline{T}(\hat{z})$ is given by:

$$\underline{T}(\hat{z}) = diag\left(\frac{1-\hat{z}^{-n}}{1-\hat{z}^{-1}}, \frac{1-\hat{z}^{-n}}{1-\hat{z}^{-1}}, \frac{1-\hat{z}^{-n}}{1-\hat{z}^{-1}}\right) \tag{6}$$

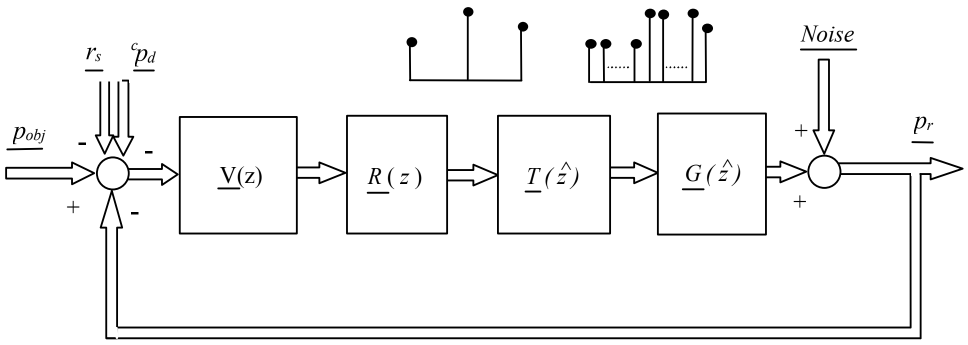


Fig. 13. Equivalent representation of the block diagram for velocity control.

At this point it is important to remark that previous block diagrams correspond to multirate control systems, because two different sampling periods appear in all the representations. In order to design the controllers at the slowest sampling rate, we should have the transfer functions matrix of the robotic subsystem corresponding to a sampling period T_v. However, we know these transfer functions sampled with period T_r. To obtain the desired transfer functions the following two step procedure can be applied: (1) Estimate the continuous transfer functions matrix from the discrete transfer functions matrix sampled with period T_r. (2) Calculate the discrete transfer functions matrix by sampling the continuous transfer functions matrix with period T_v. Applying this procedure to $\underline{G}(\hat{z})$, we obtain $\underline{G}(z)$. Then the block diagram of figure 13 can be converted into a more generic block diagram as shown in figure 14. Notice that T_v is the sampling period of this new diagram.

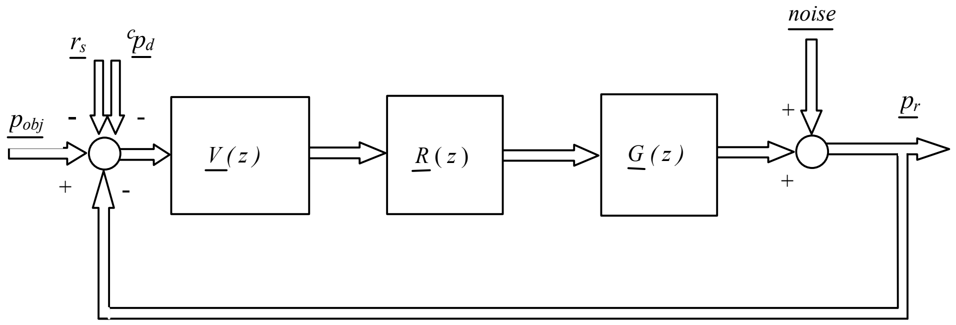


Fig. 14. Generic block diagram of the system.

4.3 Identification

Once a model has been selected to represent each subsystem, an identification process is required to estimate the unknown parameters. In general terms, an identification experiment can be performed by exciting the system (using some kind of input such as step, sinusoid or random signals) and observing the input and the output over a time interval. These signals are normally registered in a computer for information processing. Some statistically based method can be used to estimate the model unknown parameters such as the coefficients of the difference equation. The model obtained is then tested to verify whether it is an appropriate representation for the system. If that is not the case, some more complex model must be considered, its parameters estimated and the new model validated. Concerning our experimental platform the identification process can be summarized in the following comments.

As it has been exposed in the modeling section the vision subsystem behaves as a pure delay. Therefore no unknown parameter must be estimated here. The robotic subsystem is working through its ALTER line. This command allows us to introduce new reference values each 16 milliseconds, which means that in this platform T_r = *16 milliseconds*. It has already been mentioned that we are considering a second order linear system to model each coordinate of the robotic subsystem. To identify the respective parameters we have excited each Cartesian coordinate of the robot with different input signals (steps, ramps and parabolas) through its ALTER line. In order to take into account the possible errors produced in the data acquisition process, an autoregressive moving average with exogenous model (ARMAX) has been used to fit the robotic subsystem parameters, due to this model a random term is included that can be interpreted as noise. Parameters are finally identified by minimizing the estimated square error function using an iterative Gauss-Newton algorithm. The obtained transfer functions matrix is given by:

$$\underline{G}(\hat{z}) = diag\left(\frac{0.6325}{(\hat{z}-1)(\hat{z}-0.3810)}, \frac{0.6840}{(\hat{z}-1)(\hat{z}-0.3194)}, \frac{0.7081}{(\hat{z}-1)(\hat{z}-0.2953)}\right) \tag{7}$$

Before starting the control subsystem design, and as it has been mentioned in the previous section, it is first necessary to estimate the continuous transfer functions matrix from the equation

(7). To this aim, we have used the method described in (Feliu, 1986) The obtained $\underline{G}(s)$ is:

$$\underline{G}(s) = diag\left(\frac{-35.689s + 3851.45}{s^2 + 60.309s}, \frac{-42.634s + 4485.437}{s^2 + 71.4102s}, \frac{-45.992s + 4791.679}{s^2 + 76.298s}\right) \quad (8)$$

The resulting continuous transfer functions matrix is then sampled with a T_v period using a zero order hold. Considering *n=10* in the equation (1) and applying this discretization procedure to the transfer functions matrix of robotic subsystem, the following $\underline{G}(z)$ is obtained:

$$\underline{G}(z) = diag\left(\frac{8.5673z + 1.6499}{z(z-1)}, \frac{8.5734z + 1.4765}{z(z-1)}, \frac{8.6224z + 1.459}{z(z-1)}\right) \quad (9)$$

5. Second phase: Control Subsystem Design

In order to design the control subsystem ($\underline{R}(z)$) and to study the behavior of the full system, it is necessary to define the task that the robot has to do. It is evident that the controller will depend on the type of task to perform and the physical constraints of the robot. In general, the problem of robotic visual tracking/servoing is defined as in (Papanikolopoulos, 1992): "*Move the manipulator (with the camera mounted on its end effector) in such a way that the projection of a moving or static object is always at the desired location in the image*". This states the main objective of the control subsystem, but some differences appear if the object moves or remains static.

In that sense, we have considered and studied two different tasks. In the first one, denoted as task 1, the objective is to pose the robot with respect to a static object. The specifications in this case are to reach the reference with no oscillations and within a short period of time during the movement of the robot end effector when a step input is applied. Provided that this step input can be of great amplitude, it is very important to study the Cartesian acceleration value in order to avoid the saturation of some motors. The second task, denoted as task 2, consists of tracking a moving object. Specifications now are that the steady state error becomes zero in a minimum setting, in such a way that the robot will track perfectly and quickly the object when it follows a ramp, parabolic or other type of input.

In the following paragraphs the two types of tasks will be analyzed. Several controllers will be designed and tested. Some comparative observations will be added at the end.

5.1 Task 1: Tracking of a static object

In this instance, the objective is to move the robot, with the camera mounted on its end effector, in such a way that the projection of a static object is in the desired location ${}^f\underline{p}_d$ of the image (see figure 15).

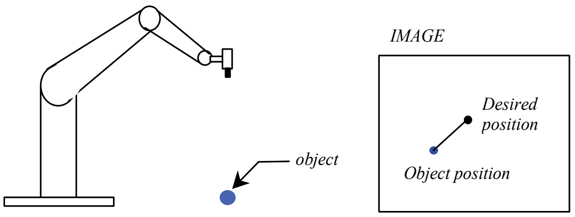


Fig. 15. Task 1 symbolic drawing.

For this task we are going to study various conventional and new methods to design the visual controller. System performance will be evaluated afterwards.

Root locus based controller

The first controller considered can be designed via root locus analysis. These controllers have the smallest performance but they also produce the smallest values of the Cartesian acceleration reference. All the transfer functions of the matrix $\underline{G}(z)$ have one pole at z = 1 in the open-loop chain (they are type 1 systems), so the steady state error is zero for a unit step input in all of them. Then a proportional controller is enough to control the robotic subsystem from the point of view of the steady state performance. The controller gains are chosen so that all the closed-loop poles will be real, with the intention of avoiding oscillations while obtaining the fastest possible response. The obtained controllers are given by:

$$\underline{R}(z) = diag(0.0214, \quad 0.0220 \quad 0.0221) \tag{10}$$

Pole placement controller

The second tested controller is designed by applying the pole placement methodology (Norman, 1995) that has already been used in other previous works like (Corke, 1993), (Vargas et al., 1999). In order to choose the closed-loop poles, an optimization process which minimizes the overshoot of the robot output waveform considering that the settling time would be smaller than a given n_s. The best results have been obtained with n_s = 6 samples. The closed loop poles for each Cartesian coordinate are located respectively at $p_1 = 0.1607$, $p_2 = -0.0366$ and $p_3 = -0.0001$. In this case, the settling interval is less than the value obtained with the proportional controller.

Optimal controller

We have also taken into account optimal control techniques to design the visual controller. In our case it has been applied as follows. Let us consider the block diagram represented in figure 16. It can be obtained from the general block diagram of figure 14 always that the transfer functions given by $\underline{G}(z)$ be linear and stable. In figure, $\underline{p}_{obj}$ represents the object position after deducing the fixed reference value of ${}^{c}\underline{p}_{d}$. Vision and robotic subsystems have been grouped into a single block whose transfer function matrix can be expressed as:

$$\underline{VG}(m,z) = \underline{V}(m,z) \cdot \underline{G}(m,z) \tag{11}$$

where modified z-transform notation is used as normal in this type of approach. Finally, $\underline{W}(z)$ has the following expression:

$$\underline{W}(z) = \frac{\underline{R}(z)}{1 + \underline{R}(z)\underline{VG}(z,m)} \tag{12}$$

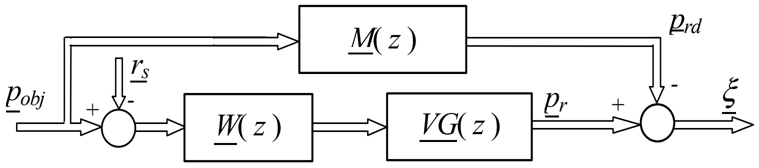


Fig. 16. Scheme used for optimal control.

The goal of the optimal control approach is to find the transfer function matrix $\underline{W}(z)$ so that it minimizes a function of the error vector $\underline{\xi}$, more specifically the integral squared error

(ISE). The error vector usually considered is $\underline{\xi} = \underline{p}_r - \underline{p}_{obj}$, the difference between the input applied and the output achieved. Obviously, the final solution will depend on the type of input chosen. For the task 1 under consideration we propose to modify the applied input by means of a smoothing transfer function matrix $\underline{M}(z)$ as indicated in figure 16. This modified input, denoted as $\underline{p}_{rd}$, tries to be a more realistic reference to be compared with the robot response than the applied input itself. Notice that high-valued step responses are not physically feasible for the robotic subsystem because they involve high torques that could surpass the maximum allowed values, provoking saturation effects in the manipulator actuators. Therefore we have tested the following transfer function matrix:

$$\underline{M}(z) = diag\left(\frac{1-a_x}{(z-a_x)\cdot z}, \frac{1-a_y}{(z-a_y)\cdot z}, \frac{1-a_z}{(z-a_z)\cdot z} \right) \tag{13}$$

where the (a_x, a_y, a_z) values are chosen to obtain the best output behavior with respect to speediness and cancellation of oscillations under the existing constrains. In the simulations the best value was equal to 0.3 for the three Cartesian coordinates. In the experimental test the same values are kept. Controller obtained is:

$$\underline{R}(z) = diag\left(\frac{0.0446z^3+0.0435z^2}{z^3+0.9849z^2+0.5180z+0.0717}, \frac{0.0441z^3+0.0449z^2}{z^3+0.9777z^2+0.5163z+0.0663}, \frac{0.0437z^3+0.0451z^2}{z^3+0.9754z^2+0.5159z+0.0644} \right) \tag{4}$$

Deadbeat controller

Another design possibility is to use a deadbeat controller, which produces the minimum settling interval. The computation of this controller for the three coordinates offers the following transfer functions matrix:

$$\underline{R}(z) = diag\left(\frac{0.0979z^2}{z^2+z+0.1615}, \frac{0.0995z^2}{z^2+z+0.1469}, \frac{0.0995z^2}{z^2+z+0.1419} \right) \tag{15}$$

5.2 Task 2: Tracking of a moving object

Now the robot has to track a moving object as fast and precisely as possible. The task of the control subsystem is to keep the relative position between the end effector and the object by generating a robot movement in order to compensate the movement of the object (see figure 17).

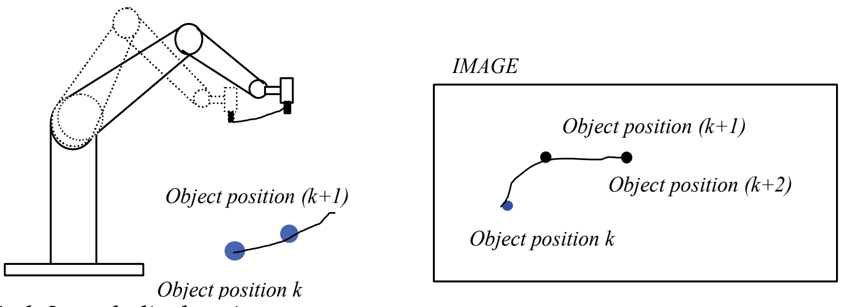


Fig. 17. Task 2 symbolic drawing.

Due to the delay introduced by the vision subsystem, in the majority of the reported experimental systems some type of estimator is used to determine the future position of the object from previous visual information (Houshangi, 1990), (Wang & Varshney, 1991), (Westmore & Wilson, 1991), (Corke, 1993). Nevertheless, we will demonstrate in this work that it is possible to eliminate the estimator while achieving performances similar to those obtained using an estimator, just by selecting and tuning an adequate control subsystem. Therefore, we consider two types of control subsystems in this task. An estimator and a controller form the first one, and it will be called *control subsystem with estimator*. Only a controller forms the second one, and it will be called *control subsystem without estimator*. Both are analyzed in the following sections.

5.2.1 Control subsystem with estimator

The one sample delay introduced by the vision subsystem leads to a delayed sensing of the target position. Making some kind of prediction to estimate the object position one sample in advance comes out like an interesting strategy to cancel the pure delay transfer function of the vision subsystem. There are different techniques to obtain the prediction of the future position, velocity and acceleration of the object. (Brown et al., 1992) compare a set of methods for velocity estimation. We have studied several types of estimator (Bachiller, 2003), such as first and second order linear regression predictors, Kalman filter based on the assumptions constant acceleration target motion and, finally, an auto-regressive discrete time model. These estimators have been used successfully for position estimation. However, they produce considerable errors when evaluating tracking performance for sinusoidal or triangular profiles of the target motion.
Once the estimator has been selected it is still necessary to design the controller in order to complete the control subsystem (see figure 8). We have examined some techniques described in the previous task but now the pure delay transfer functions of the vision subsystems are not considered because of the existence of the estimator.

Pole placement controller

For each system a pole placement compensator is designed to place the closed loop poles at $p_{1,2}$=*-0.1532±0.7618i* for X Cartesian coordinate, $p_{1,2}$=*-0.1532±0.7618i* for Y Cartesian coordinate and $p_{1,2}$=*-0.1532±0.7618i* for Z Cartesian coordinate. The method used to place these poles is the same that has been presented in task 1.

Optimal controller

The procedure used to obtain the optimal controller in this case is the same exposed for task 1. The only difference is that for task 2 no smoothing transfer function matrix $\underline{M}(z)$ is required. The obtained controller is:

$$\underline{R}(z) = diag\left(\frac{0.181z^3 - 0.2182z^2 + 0.0497z}{z^3 - 1.2661z^2 + 0.175z + 0.0911}, \frac{0.1817z^3 - 0.2181z^2 + 0.0491z}{z^3 - 1.2804z^2 + 0.1998z + 0.0806}, \frac{0.181z^3 - 0.2169z^2 + 0.0486z}{z^3 - 1.2852z^2 + 0.2081z + 0.0771} \right) \quad (16)$$

5.2.2 Control subsystem without estimator

In this work we propose a new control strategy consisting of eliminating the estimator while designing a more efficient controller. We pretend to show that similar and even better performance can be achieved by using this strategy, while some other advantages are obtained. For example, no time is consumed in evaluating the estimators and consequently the overall execution time can be reduced. Error produced in the estimation process is also eliminated. In practice, this strategy will allow us to reduce the vision period (T_v) and in addition to measure visual information more exactly.
In order to get a good tracking performance, it is necessary to cancel the steady state error in the least possible number of samples bearing in mind the torque capabilities. For this reason

we propose to design the control subsystem as a deadbeat controller system, following the conventional methodology for type of controller. Imposing as design specification that the steady state tracking error to ramp reference must be cancelled in a minimum number of samples, the transfer function matrix obtained is shown in table 1.

Notice that these transfer functions have poles outside the unit circle. In order to guarantee the stability of the output signal, we propose a modification of the design criterion to obtain the deadbeat controller in the way that the steady state tracking error must be zero in a finite number of samples. When applying the Truxal method with these specifications a higher number of unknowns than equations appear. A minimization process with restrictions is required to solve this system of equations. Specifically we have imposed the additional constraint that all the poles of the regulator have to be inside the unit circle. Table II presents the controllers finally obtained.

Minimum number of samples	$\underline{R}(z) = diag\left(\frac{0.3094z^3 - 0.2116z^2}{z^3 + z^2 - 1.651z - 0.349}, \frac{0.3094z^3 - 0.2116z^2}{z^3 + z^2 - 1.651z - 0.349}, \frac{0.3094z^3 - 0.2116z^2}{z^3 + z^2 - 1.651z - 0.349}\right)$
Finite number of samples	$\underline{R}(z) = diag\left(\frac{0.1954z^4 + 0.016z^3 - 0.1128z^2}{z^4 + z^3 - 0.6737z^2 - 1.1332z - 0.1931}, \frac{0.1954z^4 + 0.016z^3 - 0.1128z^2}{z^4 + z^3 - 0.6737z^2 - 1.1332z - 0.1931}, \frac{0.1954z^4 + 0.016z^3 - 0.1128z^2}{z^4 + z^3 - 0.6737z^2 - 1.1332z - 0.1931}\right)$

Table 1. Controllers obtained without estimator in task 2 .

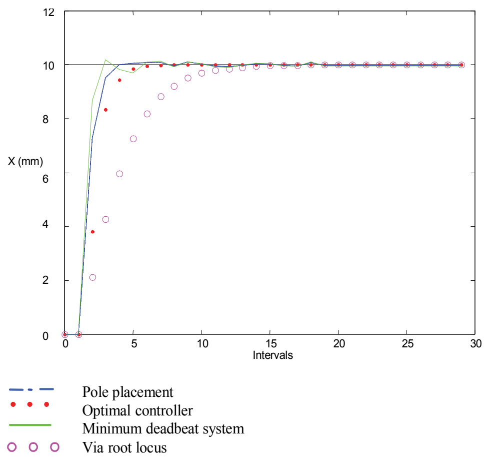
6. Third phase: Test results and performance evaluation

The next step in our plan is to evaluate the different strategies of control in order to select the most efficient controller. In this section we show how performance evaluation has been carried out in our work. We have used simulation software from MATLAB to carry out the different simulations. The following paragraphs summarize the main results obtained from simulation of the designed controllers. In general terms, plots showing the system response to the same input with the different controllers are used to compare controllers' performance.

6.1 Evaluation of the Task 1 controllers

Figure 18 shows the response associated to the X coordinate of the robotic subsystem when the input is a step of 10 millimeters. Notice that the proportional controller response is the slowest one. The three others show a similar rapidness. In order to evaluate performance some criterion must be established. The best way of doing so is to compute a set of parameters from the output signal. These parameters should give quantitative measurements to compare the behavior of the system for the different controllers used. In the present work we have chosen two significant parameters: n_s (magnitude of the settling interval) and r_a (maximum value of the Cartesian acceleration reference).

Table 2 shows the values corresponding to the controllers for the X coordinate when an input step of 10 mm is applied. No relevant differences have been observed in the two other coordinates, so they are not included here. It is noticeable that concerning the settling interval the deadbeat controller seems to be the best strategy. Nevertheless the value of Cartesian acceleration is the highest in all these solutions. The minimum value of this parameter occurs with a conventional controller, but the settling time is excessive. A trade off is achieved with the optimal controller. Using this controller the steady state error becomes zero in 6 samples, besides the maximum Cartesian acceleration is reduced to half. It means that it can be considered the best strategy for this task.



	n_s	r_a *(mm/sec²)*
Classic controller	12	13.375
Pole Placement	5	53.2255
Optimal controller	6	27.8702
Dead Beat	4	61.1710

Table 2: Performance evaluation of the designed controllers for the task 1

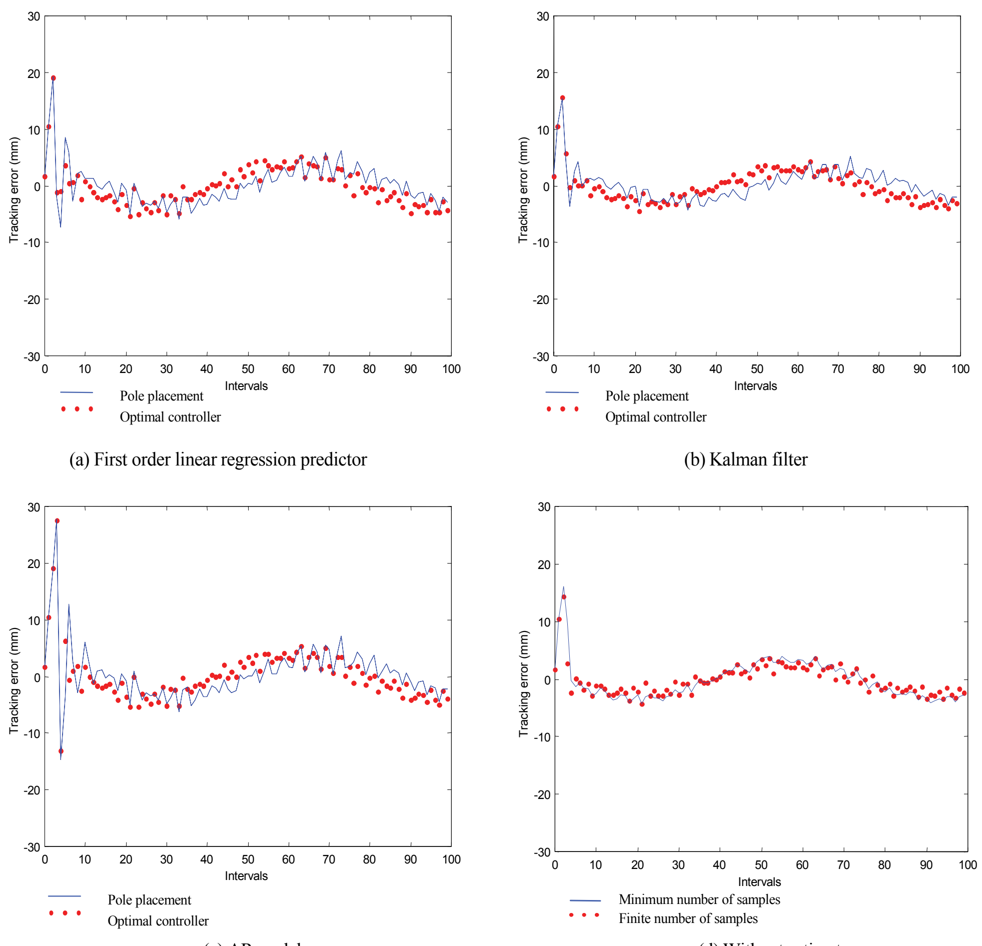
Fig. 18. Step responses of X coordinate.

6.2 Evaluation of the Task 2 controllers

In order to evaluate performance in this task it is also necessary to consider some normalized trajectory of the mobile object. In this paper we have tested several types of object movements to evaluate the behavior of the different designed controllers. First, we have considered the target object moving with a constant velocity in one of the Cartesian axis (v_x = 62.5 mm/sec). In the second place, we have considered the target tracing a circular path of 100 mm radius, with an angular velocity of 5.2 revolutions per minute (r/min). Finally, a triangular target motion has been considered to study the behavior of the system concerning the settling time after each peak.

Figure 19 shows the tracking error of the X coordinate after deducing the fixed reference value of ${}^c\underline{p}_{dx}$, i.e. $(\underline{p}_{obj} - \underline{p}_r)_x$, when the object is moving along a circular trajectory for different combinations of estimators and controllers. Notice that the tracking error presents an initial oscillation due to the acceleration at start-up, followed by smooth tracking behavior once the controller compensates for the initial disturbances. The conclusion is that the control subsystem without estimator presents the best results for the starting up movement as well as for the tracking movement. Nevertheless, the control subsystem composed of a Kalman filter and an optimal controller presents similar results.

Once the trajectory has been chosen, tracking performance can be measured in terms of bandwidth of the closed loop system. While high bandwidth is desirable to reduce error, it can also lead to a system that is sensitive to noise and non-modeled dynamics. Other commonly used performance metrics such as settling time, overshoot and tracking errors are more appropriate. In this work, we propose to consider a quantitative criterion valid for comparing the tracking responses corresponding to the different controllers under the same trajectory profile conditions. This parameter is the magnitude of the square root of the mean square error for each Cartesian coordinate measured in millimeters. This parameter allows for comparing tracking performance in terms of trajectory fitting. However, it would be convenient to have another criterion to compare all the control strategies. As in the task 1 case, it can be done by taking into account the values of the Cartesian acceleration reference. Notice that sometimes, due to torque capability constraints, it is necessary to limit the magnitude of the Cartesian acceleration reference so that they remain low enough.



(a) First order linear regression predictor

(b) Kalman filter

(c) AR model

(d) Without estimator

Fig. 19. X coordinate tracking error when evolving through a circular trajectory.

Table 3 shows the values obtained for the X coordinate under three different trajectories for all the control strategies, with and without estimator, considered in the previous section for this task. Table 4 shows the values of the Cartesian acceleration reference corresponding to each case. No relevant differences have been observed in the two other coordinates, so they are not included here. Analyzing the results of these tables it can be observed that, in general, the deadbeat controller without estimator presents the best results as for tracking error in practically all the trajectory profiles just as for values of the Cartesian acceleration. Nevertheless, the control subsystem composed of the Kalman filter and an optimal controller also gives good results for tracking error although its values of Cartesian acceleration are higher.

		Ramp	**Sinusoid**	**Triangular Profile**
Pole Placement	1st L.R.	4.5895	3.6214	5.1655
	Kalman	4.4088	3.1421	5.8296
	AR	5.6858	4.8866	5.6705
Optimal	1st L.R.	2.6057	3.5623	4.7244
	Kalman	2.5803	3.3114	5.517
	AR	4.0926	4.7006	5.1734
Minimum deadbeat system		2.4259	2.5945	4.6648
Finite deadbeat system		2.5678	3.2612	5.3475

Table 3. Root of the mean square tracking error (in mm) for several controllers.

		Ramp	**Sinusoid**	**Triangular Profile**
Pole Placement	1st L.R.	225.8460	216.0358	232.7056
	Kalman	166.4976	109.3901	190.8991
	AR	424.6092	461.6932	284.0734
Optimal	1st L.R.	218.6440	209.1466	256.7252
	Kalman	161.1881	100.9431	180.5773
	AR	367.0614	356.5014	275.3864
Minimum deadbeat system		186.8742	105.0531	194.4872
Finite deadbeat system		118.0194	82.6676	126.0523

Table 4. Maximum Cartesian acceleration (mm/sec^2) for several controllers.

In conclusion, if we consider both criteria, the finite deadbeat system is the more suitable strategy for this task. This study demonstrates that the control subsystem without estimator produces competitive results besides having a number of advantages: (1) No estimator, neither more nor less complicated, is required and (2) the control subsystem is simple and of easy implementation.

7. Experimental platform validation

This final step of this work is devoted to verifying experimentally the results obtained previously by simulations, with the purpose of demonstrating its design capabilities. On the other hand, real results obtained in this step will allow comparing our methodology with other published approaches. In any case, a real platform is required to carry out such experimentation.

We have developed a visual feedback control system consisting of a Staübli RX90 manipulator, a PC, a camera and a MATROX image processing hardware. The vision computer is connected to the robotic subsystem through a RS 232 series line working to 19200 bauds and it calculates the position increment each 160 msec. However, it would be possible to reduce the value of the vision period using a more powerful image processing hardware or improving the real time image processing software. The

manipulator trajectory is controlled via the Unimate controller's Alter line that requires path control updates every 16 msec. A photograph of the experimental platform is shown in figure 20.
In all the experiments presented in this work, we have kept the reference for the Z-depth, i.e., the vertical distance from the camera to the target, at 330 mm. Furthermore, to achieve a particular motion, the object has been grasped by the end effector of a second robot, a Staubly RX60 manipulator. This allows defining any trajectory in the space as required in the second task. It also allows keeping the two-point target at constant orientation during the trajectory evolution. Camera calibration and inverse calibration algorithms have been specifically developed for the camera and environment, and are described in (Gonzalez, 1998). Anyway, the vision model used for these experiments has been explained in section 4 of this paper.

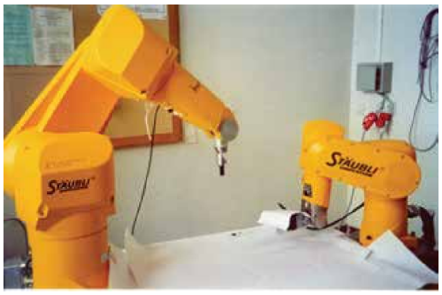


Fig. 20. Experimental setup.

The experimentation carried out over this platform will be shown and commented in the next paragraphs. In general, although all the presented controllers have been tested on the platform, only the results corresponding to the best control strategy selected in the previous step will be shown and studied here.

7.1 Experimental results of Task 1

In the static object case we have demonstrated with simulations that the best control strategy was the optimal controller. This regulator has been implemented on the platform and the final pose of the robotic manipulator with respect to the target object for a step input is achieved in 6 samples (0.96 sec.). Besides it produces small values of the Cartesian acceleration reference avoiding saturation of joint motors, contrary to the deadbeat controller. Figure 21 shows the plot of the error relative to the X coordinate registered by the vision subsystem when the input is a step of 40 mm.

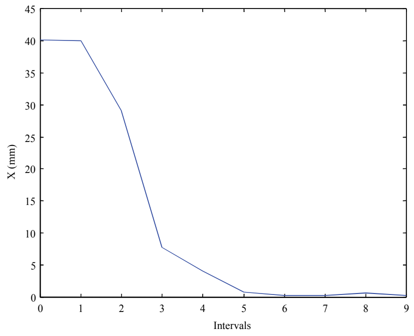
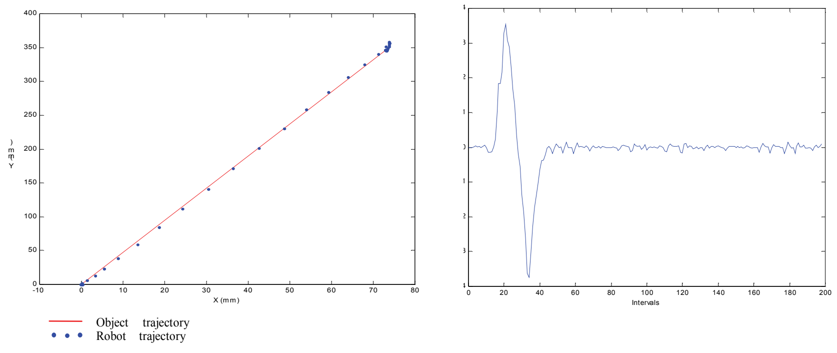


Fig. 21. Error registered by the vision subsystem when the input is a step of 40 mm.

7.2 Experimental results of Task 2

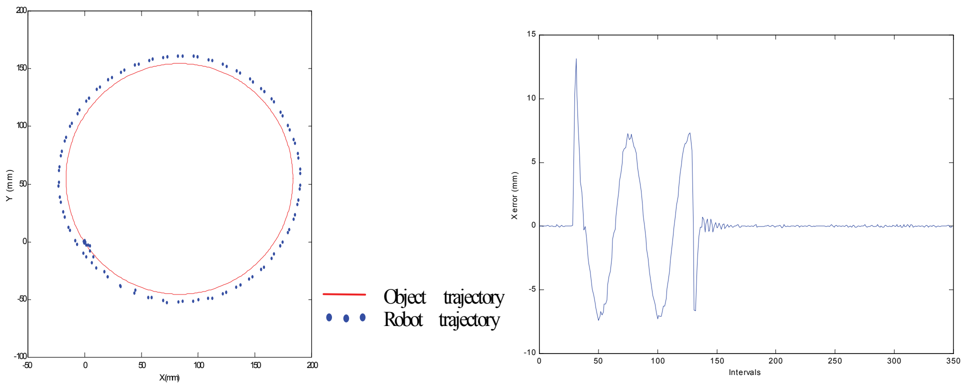
We have accomplished multiple experimental runs for tracking the target object following unknown translation motions into the perpendicular plane of the camera. During these experiments, we have tested the system using mainly linear and curved trajectories for the target.

The first type of experiments was conducted using an object moving along a linear trajectory. The system was able to track successfully the target moving at 185 mm/sec. Figure 22 shows a plot of this experiment relative to the X coordinate of the movement. The next set of experiments was performed making the target follow a circular path with a radius of 150 mm. This path produces a smooth variation of the velocity curve. The system was able to track the object moving with a circular speed of 7.5 radians/min exhibiting a tracking error of ±17 pixels, equivalent to ±7 mm at a depth of 330mm. These results are shown in figure 23. Finally, figure 24 shows an example of a trajectory in the 3D space. The object is moving along a spiral path composed by a circular movement in the XY plane and a linear movement in the Z direction.



(a) Object and robot linear trajectory (b) X coordinate tracking error

Fig. 22. Measured tracking performance for target moving along a linear trajectory with the finite deadbeat system.



(a) Object and robot circular trajectory (b) X coordinate tracking error

Fig. 23. Measured tracking performance for target moving along a circular trajectory with the finite deadbeat system.

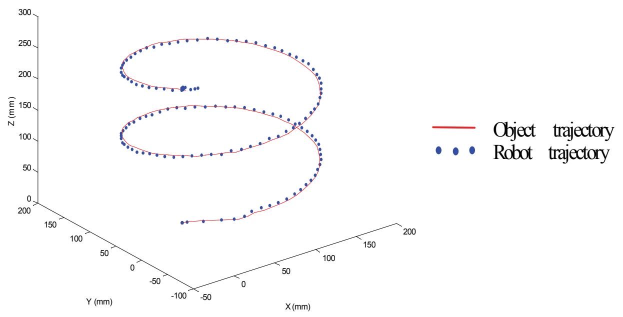


Fig. 24. Object and robot spiral trajectory.

8. Conclusions and new tendencies

This article has presented a complete working plan for designing and performance evaluation of position based dynamic look and move systems using a 6 DOF industrial manipulator and a camera mounted on its end effector. The robot used works using the ALTER line facility, allowing a direct and uncoupled control in Cartesian coordinates.
In the first stage, the dynamic models proposed for the robotic and vision subsystems have been presented, identified and validated by experimental tests. Special attention has been paid to describe the vision model, which has been optimized to cope only with the three-dimensional Cartesian position measurement, in order to reduce the total on-line computation time. The overall block diagram has also been discussed. It corresponds to a multirate control system scheme and a method to calculate its equivalent control scheme at the vision sampling period which has been proposed and applied.
The design of diverse control strategies has been considered in a second stage. Two different control tasks have been taken into account, concerning respectively the tracking of a static object and of a moving one. In both cases, several controllers have been designed and a comparative study of the different behaviors of the overall system has been presented. It is particularly remarkable the optimal control proposal made for task 1. It has been designed so that the possible saturation effects in the manipulator actuators are taken into account by introducing a smoothing transfer function, which is estimated in the optimization process. Simulations of all the considered controllers have been carried out. Selection of the most efficient control strategy for each task has been possible using this strategy. The selected controllers have been implemented and validated on an experimental platform. Let us remark that these controllers are very simple, and consequently more than satisfactory for real time implementation, whereas they achieve high performance functionality.
Finally this work has shown that, following an adequate design methodology like the one presented, a well-designed controller without using estimator can be more efficient than the combination of the estimator plus a classical controller in the considered tracking problem. Specifically, it has been demonstrated that the control subsystem without estimator improves the behavior of the entire system output as for tracking error and value of Cartesian acceleration. The results presented in this paper demonstrate the effectiveness of the designed position based dynamic look and move system to track both static and moving objects.

The visual servoing structure used in this work has the characteristic of being integrated by independent subsystems with its corresponding advantages. Its principal disadvantage is that the torque vector acting over the robot's motors does not appear explicitly and, therefore, it is not possible to know when a referenced movement can saturate any of the motors. For this reason, future experiments will focus on aspects as the study of different schemes that could maintain the exerted torque small and, in any case, under control.
Only one camera has been used in this work, considering that the information supplied such vision subsystem does not introduce uncertainties because of its simplicity. However, when the robot is operating in an unstructured environment, such sensor information and such approach are not always satisfactory. The problem increases when three-dimensional information must be managed, like in the multiple cameras case.
From this point of view, other visual servoing classification not considered in this work should be taken into account, having in mind the number of sensors used. Following cases could be distinguished: (1) monocular visual servoing that uses one camera which can be a fixed camera (fixed camera configuration) or mounted at the end effector the robot (eye-in-hand configuration); (2) multi camera vision system where multiple cameras placed in the work-space are used to get the task specific information. The monocular visual servoing is the simplest solution; however, the depth information of the work-space is lost. This information can be calculated when the vision system uses multiple cameras but not always is possible to extract all the 6-DOF information. Additional difficulties appear when the extrinsic camera parameters (position and orientation of the camera within the work-space) are inexactly obtained due to uncertainties in the camera calibration process.
Motivated by this, the focus in visual servoing has shifted towards 3D problems and compensating the uncertainties. Saedan & Ang (Saedan & Ang, 2002) design a PI controller based 3D pose servoing scheme. This system gives accurate position and orientation control for stacionary target applications, but was unable to avoid error fluctuations for tracking problems. Sahin & Zergeroglu (Sahin & Zergeroglu, 2005) use an adaptative controller based 3D position servoing. This controller is designed to compensate the uncertainties associated with the mechanical parameters of the robot manipulador and intrisic parameters of the cameras. However, this work only presents simulation results to illustrate the performance of the proposed controller. Another solution (Malis, 2004) presents a new visual servoing scheme which is invariant to changes in camera intrinsic parameters. This work shows how to position a camera, with respect to non-planar object, even if the intrinsic parameters change.

9. Acknowledgements

This work has been supported by the Spanish Government under contract through the DPI2005-03769 project, and by the Autonomous Community of Madrid under the RoboCity2030 project, referenced as S-0505/DPI/0176.

10. References

Allen, P. K.; Timcenko, A.; Yoshimi, B. & Michelman, P. (1993). Automated tracking and grasping of a moving object with a robotic hand-eye system, *IEEE Transactions on Robotics and Automation*, Vol. 9, No. 2, 1993, pp. 152-165.

Bachiller, M; Cerrada, J. A. & Cerrada, C. (2003). A Modular Scheme for Controller Design and Performance Evaluation in 3D Visual Servoing, *Journal of Intelligent & Robotic*

Systems, Vol. 36, No. 3, Marzo 2003, pp. 235-264.

Brown, R.; Schneider, S. & Mulligan, M. M. (1992). Analysis of algorithms for velocity estimation from discrete position versus time data, *IEEE Transactions Industrial Electronics*, Vol. 39, No. 1, 1992, pp. 11-19.

Corke, P. I. (1993). Visual control of robot manipulators –A review in Visual Servoing, K. Hashimoto, Ed. Singapore: World Scientific.

Chaumette, F.; Rives, P. & Espiau, B. (1991). Positioning of a robot with respect to an object, tracking it and estimating its velocity by visual servoing, *Proceedings of IEEE Int. Conf. on Robotics and Automation*, 1991, pp. 2248-2253.

Craig, J. J. (1989). Introduction to robotics. Mechanics and control. *Addison-Wesley Publishing Company.*

Espiau, B.; Chaumette, F. & Rives, P. (1992). A new approach to visual servoing in robotics, *IEEE Transactions on Robotics and Automation*, Vol 8, 1992, pp. 313-326.

Feddema, J. T. & Mitchell, O. R. (1989). Vision guided servoing with feature based trajectory generation, *IEEE Transactions on Robotics and Automation*, Vol 5, 1989, pp. 691-700.

Feliu, V. (1986). A Transformation Algorithm for Estimating System Laplace Transform from Sampled Data, *IEEE Transactions on Systems, Man, and Cybernetics*, 1986, pp 168-173.

Gangloff, J. A.; Mathelin, M. & Abba, G. (1998). 6 dof high speed dynamic visual servoing using gpc controllers, *Proceedings of the 1998 IEEE Int. Conf. on Robotics and Automation*, 1998, pp. 2008-2013.

Gonzalez, Y. (1998). Aportaciones a la calibración de cámaras con distorsión geométrica. Aplicación al control con realimentación visual, *PhD thesis* UNED, 1998.

Hashimoto, K.; Ebine, T. & Kimura, H. (1996). Visual servoing with hand eye manipulator optimal control approach, *IEEE Transactions on Robotics and Automation*, 1996, pp. 766-774.

Houshangi, N. (1990). Control of robotic manipulator to grasp a moving target using vision, *In Proceedings of. IEEE Int. Conf. Robotics and Automation*, 1990, pp. 604-609.

Khosla, P. K.; Papanikolopoulos, N. P. & Kanade, T. (1993). Visual tracking of a moving target by a camera mounted on a robot: A combination of control and vision, *IEEE Transactions on Robotics and Automation*, Vol. 9, 1993, pp. 14-35.

Koivo, A. J. & Houshangi, N. (1991). Real time vision feedback for servoing robotic manipulator with self tuning controller, *IEEE Transactions on Systems, Man, Cybernetics*, Vol. 1, No. 1, 1991, pp. 134-141.

Malis, E.; Chaumette, F. & Boudet, S. (1999). 2-1/2-d visual servoing. *IEEE Transactions on Robotics and Automation*, Vol. 15, No. 2, April, 1999, pp. 1352-1359.

Malis, E. (2004). Improving vision-based control using efficient second-order minimization techniques. *IEEE Int. Conf. on Robotics and Automation*, 2004.

Nelson, B. & Papanikolopoulos, N. (1998). Special Issue on Visual Servoing, *IEEE Robotics and Automation Magazine,* Vol. 5, No. 4, December, 1998.

Norman, S. N. (1992). Control Systems Engineering. Addison-Wesley, 1995.

Papanikolopoulos, N. P. (1992). Controlled Active Vision, *PhD thesis, Carnegie Mellon University*, 1992.

Petersson, L.; Jensfelt, P.; Tell, D.; Strandberg, M.; Kragic, D. & Christensen, H. I. (2002). System integration for real-world manipulation task. *Proceedings of IEEE Int. Conf. Robotics and Automation*, 2002, pp. 2500-2505.

Saedan, M. & Ang, M. H. (2002). 3D Vision-Based Control On An Industrial Robot. *Proceedings of IASTED International Conference on Robotics and Applications,* Florida, USA, November, 2002, pp. 19-22.

SaHin, H. T. & Zergeroglu, E. (2005). Adaptative Visual Servo Control of Robot Manipulators via Composite Camera Inputs. *Fifth International Workshop on Robot Motion and Control ROMOCO05,* Dymaczewo-Poland, June, 2005, pp.219-224.

Vargas, M.; Rubio, F. R. & Malpesa, A. R. (1999). Pose-estimation and control in a 3D visual servoing system, *IFAC,* 1999, pp. 317-323.

Wang, T. C. & Varshney, P. K. (1991) A measurement preprocessing approach for target tracking, *IFAC*, Vol. 7, 1991, pp. 199-202.

Weiss, L. (1984). Dynamic Visual Servo Control of Robots: an Adaptative Image Based Approach, *PhD Thesis, Carnegie Mellon University*, 1984.

Westmore, D. B. & Wilson, W. J. (1991). Direct dynamic control of a robot using an end point mounted camera and Kalman filter position estimation, *Proceedings of IEEE Int. Conf. Robotics and Automation*, 1991, pp. 2376-2384.

Wilson, W.; Williams, C. C. & Bell, G. S. (1996). Relative end-effector control using Cartesian position based visual servoing, *IEEE Transactions on Robotics and Automation*, Vol. 12, No. 5, 1996, pp. 684-696.

Robot Calibration: Modeling Measurement and Applications

José Maurício S. T. Motta
University of Brasilia
Brazil

1. Introduction

Most currently used industrial robots are still programmed by a teach pendant, especially in the automotive industry. However the importance of off-line programming in industry as an alternative to teach-in programming is steadily increasing. The main reason for this trend is the need to minimize machine downtime and thus to improve the rate of robot utilization. Nonetheless, for a successful accomplishment of off-line programming the robots need to be not only repeatable but also accurate.

In addition to improving robot accuracy through software (rather than by changing the mechanical structure or design of the robot), calibration techniques can also minimize the risk of having to change application programs due to slight changes or drifts (wearing of parts, dimension drifts or tolerances, and component replacement effects) in the robot system. This is mostly important in applications that may involve a large number of task points.

Theoretically, with the availability of robot calibration systems integrated with off-line programming systems, it would be possible to implement off-line programming in industry (currently they are not ready to be used in large scale), where multiple robots are used, by programming only one robot and copying the programs from one robot to the others. One of the most evident advantages of this type of programming is the reduction in maintenance expenditure, since the robot can return faster to service and easy reprogramming means the software becomes operational with little effort.

Robot calibration is an integrated process of modeling, measurement, numeric identification of actual physical characteristics of a robot, and implementation of a new model. The calibration procedure first involves the development of a kinematic model whose parameters represent accurately the actual robot. Next, specifically selected robot characteristics are measured using measurement instruments with known accuracy. Then a parameter identification procedure is used to compute the set of parameter values which, when introduced in the robot nominal model, accurately represents the measured robot behavior. Finally, the model in the position control software is corrected.

Many factors such as numerical problems, measurement system deficiencies, cumbersome setups and commercial interests have defied the confidence of the industry in such systems, especially if one realizes that a robot will not normally need full calibration more than once or twice a year.

The main objective of this chapter is to present and discuss several theoretical and practical aspects involving methods and systems for robot calibration. Firstly, it is discussed the implementation of techniques to optimize kinematic models for robot calibration through numerical optimization of the mathematical model. The optimized model is then used to compensate the model errors in an off-line programming system, enhancing significantly the robot kinematic model accuracy. The optimized model can be constructed in an easy and straight operation, through automatic assignment of joint coordinate systems and geometric parameter to the robot links. Assignment of coordinate systems by this technique avoids model singularities that usually spoil robot calibration results. Secondly, calibration results are discussed using a Coordinate Measuring Arm as a measurement system on an ABB IRB 2000 Robot.
Further in the chapter it is presented and discussed a 3-D vision-based measurement system developed specifically for robot calibration requirements showing up to be a feasible alternative for high cost measurement systems. The measurement system is portable, accurate and low cost, consisting of a single CCD camera mounted on the robot tool flange to measure the robot end-effector's pose relative to a world coordinate system. Radial lens distortion is included in the photogrammetric model. Scale factors and image centers are obtained with innovative techniques, making use of a multiview approach. Experimentation is performed on three industrial robots to test their position accuracy improvement using the calibration system proposed: an ABB IRB-2400, IRB 6400 and a PUMA-500. The proposed off-line robot calibration system is fast, accurate and ease to setup.
Finally, many theoretical and practical aspects are discussed concerning the relationship between measuring volumes, positions, robot types and measurement systems and the final accuracy expected after the calibration process.

2. Robot Calibration Models for Off-line Programming

Off-line programming is, by definition, the technique of generating a robot program without using a real machine. It presents several advantages over the on-line method. However, there are inevitably differences between the computer model used to perform the graphic simulation and the real world. This is because the effective use of off-line programming in industrial robots requires, additionally, a knowledge of tolerances of the manufacturing components in order to enable realistic planning, i.e. to reduce the gap between simulation and reality. In an actual robot system programming this is still a relatively difficult task and the generated programs still require manual on-line modification at the shop floor. A typical welding line with 30 robots and 40 welding spots per robot takes about 400 hours for robot teaching (Bernhardt, 1997). The difficulties are not only in the determination of how the robot can perform correctly its function, but also for it to be able to achieve accurately a desired location in the workspace. Robot pose errors are attributed to several sources, including the constant (or configuration-independent) errors in parameters (link lengths and joint offsets), deviations which vary predictably with position (e.g., compliance, gear transmission errors) and random errors (e.g., due to the finite resolution of joint encoders). Constant errors are referred to as geometric errors and variable errors are referred to as non-geometric errors (Roth, Mooring and Ravani, 1987). According to Bernhardt (1997) and Schröer (1993), constant errors represent approximately 90% of the overall robot pose errors. Industrial robots usually show pose errors from about 5 to 15mm, even when they are new, and after proper calibration these error can be reduced to about less than 0.5mm (Bernhardt, 1997, Motta, Carvalho and McMaster, 2001).

The Robot Calibration problem has been investigated for more than two decades, but some of its obstacles are still around. Usually, one can tackle the problem implementing model or modeless methods. Modeless methods does not need any kinematic model, using only a grid of known points located in the robot workspace as a standard calibration board. The robot is moved through all the grid points, and the position errors are stored for future compensation by using a bilinear interpolation method and polynomial fitting (Zhuang & Roth, 1996, Park, Xu and Mills, 2002) or error mapping (Bai and Wang, 2004). Although modeless methods are simpler, the calibrated workspace region is small, and each time the robot is to work off that region the calibration process has to be repeated. On the other side, model methods allow a large workspace region to be calibrated, leading to full model calibration. However, important to model methods is an accurate kinematic model that is complete, minimal and continuous and has identifiable parameters (Schröer, Albright and Grethlein, 1997). Researchers have used specific kinematic models that depend on a particular robot geometry and/or calibration method. Model identifiability has already been addressed (e.g., Everett and Hsu, 1988, Zhuang, 1992), and Motta and McMaster (1999) and Motta, Carvalho e McMaster (2001) have shown experimental and simulation results using a rational technique to find an optimal model for a specific joint configuration, requiring a few number of measurement points (for a 6 DOF robot only 15 measurement points) for a model with only geometric parameters (30), in opposition to hundreds of measurement points claimed by other authors (Drouet et al., 2002, Park, Xu and Mills, 2002). A model with singularities or quasi-singular parameterization turns the identification process to be ill-conditioned, leading to solutions that cannot satisfy accuracy requirements when the manipulator is to move off the measurement points. Furthermore, time to convergence increases or may not exist convergence at all, and the number of measurement points may be ten-fold larger than the necessary (Motta, 1999).

3. A Singularity-Free Approach for Kinematic Models

Single minimal modeling convention that can be applied uniformly to all possible robot geometries cannot exist owing to fundamental topological reasons concerning mappings from Euclidean vectors to spheres (Schröer, 1993). However, after investigating many topological problems in robots, concerning inverse kinematics and singularities, Baker (1990) suggested that the availability of an assortment of methods for determining whether or not inverse kinematic functions can be defined on various subsets of the operational spaces would be useful, but even more important, a collection of methods by which inverse functions can actually be constructed in specific situations. Another insightful paper about robot topologies was published by Gottlieb (1986), who noted that inverse functions can never be entirely successful in circumventing the problems of singularities when pointing or orienting.

Mathematically, model-continuity is equivalent to continuity of the inverse function T^{-1}, where T is the product of elementary transformations (rotation and translation) between joints. From this, the definition of parameterization's singularity can be stated as a transformation $T_s \in E$ (parameterization's space of the Euclidean Group - 3 rotations and 3 translations), where the parameter vector $p \in \mathrm{R}^6$ (p represents distance or angle) exists such that the rank of the Jacobian $J_s = dT_s/dp$ is smaller than 6. In other way, each parameterization T can be investigated concerning their singularities detecting the zeroes of determinant $\det(J^T.J)$ considered as a function of parameter p.

Thus, investigating the main kinematic modeling conventions one can represent the transformation between links in the Euclidean Group as

$$T = T_x(p_x).T_y(p_y).T_z(p_z).R_z(\gamma).R_y(\beta).R_x(\alpha) \quad (1)$$

where p_x, p_y, p_z are translation coordinates and α, β, γ are rotation coordinates for the axes x, y and z respectively. Then,

$$T = \begin{bmatrix} C\gamma.C\beta & C\gamma.S\beta.S\alpha - S\gamma.C\alpha & C\gamma.S\beta.C\alpha + S\gamma.S\alpha & Px \\ S\gamma.C\beta & S\gamma.S\beta.S\alpha + C\gamma.C\alpha & S\gamma.S\beta.C\alpha - C\gamma.S\alpha & Py \\ -S\beta & C\beta.S\alpha & C\beta.C\alpha & Pz \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (2)$$

where C=cos( ) and S=sin( ).
In a more simple symbolic form it can be represented as

$$T = \begin{bmatrix} n_x & o_x & a_x & p_x \\ n_y & o_y & a_y & p_y \\ n_z & o_z & a_z & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3)$$

which after the decomposition of the singularities through the Jacobian determinant results in a manifold of singularities

$$T_s = \begin{bmatrix} 0 & o_x & a_x & p_x \\ 0 & o_y & a_y & p_y \\ \pm 1 & 0 & 0 & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (4)$$

This manifold represents $\gamma = \pm\pi/2$ and $\beta = \pm\pi/2$ (eq. 2), which means there is a singularity when y and z axes are parallel.
The Denavit-Hartemberg (D-H) convention (parameterization) (Paul, 1981) leads to an elementary transformation represented by

$$T(\theta,p_z,p_x,\alpha) = R_z(\theta).T_z(p_z).T_x(p_x).R_x(\alpha) \quad (5)$$

Following the same procedure as before the manifold of singularities is

$$T_s = \begin{bmatrix} n_x & o_x & 0 & p_x \\ n_y & o_y & 0 & p_y \\ n_z & o_z & \pm 1 & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (6)$$

This result consists of all elements represented as parallel rotary joints. This can be verified by observing the third column showing the representation of one joint axis (z) into the previous one.
The Hayati-convention (Hayati & Mirmirani, 1985) can be represented by

$$T(\theta,p_x,\alpha,\beta) = R_z(\theta).T_x(p_x).R_x(\alpha).R_y(\beta) \quad (7)$$

There are two manifolds of singularities,

$$T_s = \begin{bmatrix} n_x & o_x & a_x & p_x \\ n_y & o_y & a_y & p_y \\ n_z & o_z & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \text{ or } T_s = \begin{bmatrix} n_x & o_x & \lambda p_x & p_x \\ n_y & o_y & \lambda p_y & p_y \\ n_z & o_z & \lambda p_z & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (8)$$

These representations show that if the distal *z* axis is in the same *x*-*y* plane of the proximal coordinate system (so perpendicular) or points to its origin, then there is a singularity.
The Veitschegger convention (Veitschegger & Wu, 1986) is a 5-dimensional parameterization as

$$T = R_z(\theta).T_z(p_z).T_x(p_x).R_x(\alpha).R_y(\beta) \tag{9}$$

The manifolds of singularities are

$$T_s = \begin{bmatrix} n_x & o_x & a_x & 0 \\ n_y & o_y & a_y & 0 \\ n_z & o_z & a_z & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \text{or } T_s = \begin{bmatrix} n_x & o_x & a_x & \lambda a_x \\ n_y & o_y & a_y & \lambda a_y \\ n_z & o_z & 0 & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{10}$$

These representations show that if the origin of the distal joint coordinate system is on the *z*-axis of the proximal one, or the *z* axes are perpendicular and intercept each other, then there is a singularity. However this convention is not minimal.
Using the same technique presented so far for prismatic joints sets of parameterizations can be used so fourth. The results can be outlined in a list together with their application ranges. The set is a complete, minimal and model-continuous kinematic model (Schröer et al., 1997). The transformations are modeled from a current frame C to a revolute joint, J_R, or to a prismatic joint, J_P, or a fixed and arbitrarily located target frame, TCP-frame. Some of them are not unique since other solutions can fulfill the requirements. Additional elementary transformations are possible, but any added elementary transformation is redundant and thus, cannot be identified. A non-expanded model can always be found, which describes the same kinematic structure as the expanded model.
Elementary transformations that are part of the complete, minimal and model-continuous sub-model being identified are marked bold, i.e. those elementary transformations including model parameters to be identified (errors): (symbols ⊥ and | | mean orthogonal and parallel axes respectively)

- Transformation from robot base frame (B) to first joint where joint is translational (J_T):

$$B \perp J_T: \ P = (T_X, T_Y, T_Z, \mathbf{R_Z}, \mathbf{R_X}) \tag{11}$$
$$B \parallel J_T: \ P = (T_X, T_Y, T_Z, \mathbf{R_X}, \mathbf{R_Y}) \tag{12}$$

And, if first joint is rotational (J_R):

$$B \perp J_R: \ P_X = (\mathbf{T_Y}, \mathbf{T_Z}, \mathbf{R_Z}, \mathbf{R_X}) \tag{13}$$
$$P_Y = (\mathbf{T_X}, \mathbf{T_Z}, \mathbf{R_Z}, \mathbf{R_X}) \tag{14}$$

(If joint axis is near **x**-axis of frame B)

$$B \parallel J_R: \ P_Z = (\mathbf{T_X}, \mathbf{T_Y}, \mathbf{R_X}, \mathbf{R_Y}) \tag{15}$$

(If joint axis is near **y**-axis of frame B)

- Transformations between consecutive joint frames:

$$J_R \perp J_R: \quad P = (\mathbf{R_Z}, \mathbf{T_Z}, \mathbf{T_X}, \mathbf{R_X}, T_Z) \tag{16}$$

(D-H parameterization)

$$J_R \parallel J_R: \quad P = (\mathbf{R_Z}, \mathbf{T_X}, \mathbf{R_X}, \mathbf{R_Y}, T_Z) \tag{17}$$

(assumption: joint axes are not identical)
(Hayati parameterization)

$$J_T \perp J_R: \quad P = (\mathbf{T_Z, R_Z, T_X, R_X}, T_Z) \tag{18}$$

$$J_T \,||\, J_R: \quad P = (T_Z, \mathbf{T_X, T_Y, R_X, R_Y}, T_Z) \tag{19}$$

$$J_T \perp J_T: \quad P = (T_Z, T_X, T_Y, \mathbf{R_Z, R_X}) \tag{20}$$

$$J_T \,||\, J_T: \quad P = (T_Z, T_X, T_Y, \mathbf{R_X, R_Y}) \tag{21}$$

$$J_R \perp J_T: \quad P = (\mathbf{R_Z}, T_X, T_Y, T_Z, \mathbf{R_X}) \tag{22}$$

$$J_R \,||\, J_T: \quad P = (R_Z, T_X, T_Y, T_Z, \mathbf{R_X, R_Y}) \tag{23}$$

- Transformation from last joint to TCP (Tool Center Point) :

$$J_T \perp TCP: \ P = (\mathbf{T_Z, T_Y, T_X, [R_Z, R_Y, R_Z]}) \tag{24}$$

$$J_T \,||\, TCP: \ P = (\mathbf{T_Z, T_Y, T_X, [R_Z, R_Y, R_X]}) \tag{25}$$

$$J_R \perp TCP: \ P = (R_Z, \mathbf{T_X, T_Y, T_Z, [R_Z, R_Y, R_Z]}) \tag{26}$$

$$J_R \,||\, TCP: \ P = (R_Z, \mathbf{T_X, T_Y, T_Z, [R_Z, R_Y, R_X]}) \tag{27}$$

Parameters in brackets are not identifiable without TCP-orientation measurements. As an example of the application of the equations above a case using eqs. (16) and (17) is shown below, namely, the known Denavit-Hartemberg (D-H) and Hayati parameterizations. The equations are referred to Fig. 1. For the D-H case one can define four cases:
1 – $Z_R(2)$ is parallel in the same direction as X_r:

$$P = R_Z(90^\circ).T_Z(p_z).T_X(p_y).R_X(90^\circ).T_Z(p_x) \tag{28}$$

2 - Z_R (3) is parallel in the opposite direction of X_r:

$$P = R_Z(90^\circ).T_Z(p_z).T_X(p_y).R_X(-90^\circ).T_Z(-p_x) \tag{29}$$

3 - Z_R (4) is parallel in the same direction as Y_r:

$$P = R_Z(0^\circ).T_Z(p_z).T_X(p_y).R_X(-90^\circ).T_Z(-p_y) \tag{30}$$

4 - Z_R (5) is parallel in the opposite direction of Y_r:

$$P = R_Z(0^\circ).T_Z(p_z).T_X(p_x).R_X(90^\circ).T_Z(-p_y) \tag{31}$$

For the case of the Hayati parameterization (eq. 17) (valid only for $T_X \neq 0$) if a joint subsequent position needs two translation parameters to be located, (i. e. in the X and Y direction) one of the two has to be vanished, locating the joint frame in a position such that only one remains. Four cases may be assigned:
Supposing only p_x:
1 – $Z_R(0)$ is in the same direction as Z_r:

$$P = R_Z(0^\circ).T_X(p_x).R_X(0^\circ).R_Y(0^\circ).T_Z(p_z) \tag{32}$$

2 – $Z_R(1)$ is in the opposite direction of Z_r:

$$P = R_Z(0^\circ).T_X(p_x).\ R_X(180^\circ).R_Y(0^\circ).T_Z(-p_z) \tag{33}$$

Supposing only p_y:

3 – $Z_R(0)$ is in the same direction as Z_r:

$$P = R_Z(90^\circ).T_X(p_y).\ R_X(0^\circ).R_Y(0^\circ).T_Z(p_z) \tag{34}$$

4 – $Z_R(0)$ is in the opposite direction of Z_r:

$$P = R_Z(90^\circ).T_X(p_y).\ R_X(0^\circ).R_Y(0^\circ).T_Z(-p_z) \tag{35}$$

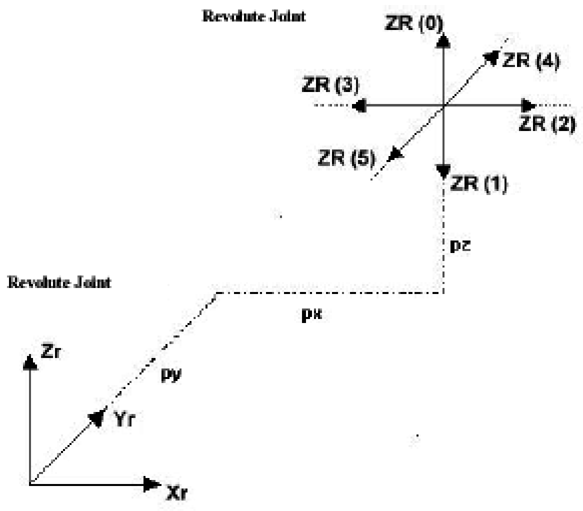


Fig. 1. Frames showing the D-H and Hayati parametrizations.

4. Kinematic Modeling - Assignment of Coordinate Frames

The first step to kinematic modeling is the proper assignment of coordinate frames to each link. Each coordinate system here is orthogonal, and the axes obey the right-hand rule.
For the assignment of coordinate frames to each link one may move the manipulator to its zero position. The zero position of the manipulator is the position where all joint variables are zero. This procedure may be useful to check if the zero positions of the model constructed are the same as those used by the controller, avoiding the need of introducing constant deviations to the joint variables (joint positions).
Subsequently the z-axis of each joint should be made coincident with the joint axis. This convention is used by many authors and in many robot controllers (McKerrow, 1995, Paul, 1981). For a prismatic joint, the direction of the z-axis is in the direction of motion, and its sense is away from the joint. For a revolute joint, the sense of the z-axis is towards the positive direction of rotation around the z-axis. The positive direction of rotation of each joint can be easily found by moving the robot and reading the joint positions on the robot controller display.
According to McKerrow (1995) and Paul (1981), the base coordinate frame (robot reference) may be assigned with axes parallel to the world coordinate frame. The origin of the base frame is coincident with the origin of joint 1 (first joint). This assumes that the axis of the

first joint is normal to the x-y plane. This location for the base frame coincides with many manufacturers' defined base frame.
Afterwards coordinate frames are attached to the link at its distal joint (joint farthest from the base). A frame is internal to the link it is attached to (there is no movements relative to it), and the succeeding link moves relative to it. Thus, coordinate frame *i* is at joint *i+1*, that is, the joint that connects link *i* to link *i+1*.
The origin of the frame is placed as following: if the joint axes of a link intersect, then the origin of the frame attached to the link is placed at the joint axes intersection; if the joint axes are parallel or do not intersect, then the frame origin is placed at the distal joint; subsequently, if a frame origin is described relative to another coordinate frame by using more than one direction, then it must be moved to make use of only one direction if possible. Thus, the frame origins will be described using the minimum number of link parameters.

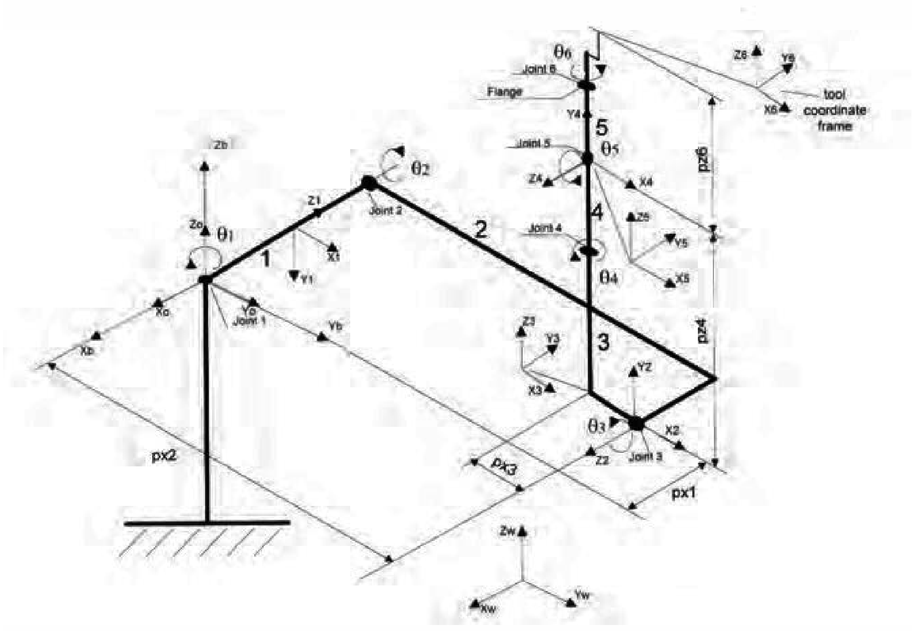


Fig. 2. Skeleton of the PUMA 560 Robot with coordinate frames in the zero position and geometric variables for kinematic modeling. (Out of scale).

The x-axis or the y-axis have their direction according to the convention used to parameterize the transformations between links (e.g. eqs. 16 to 23). At this point the homogeneous transformations between joints must have already been determined. The other axis (x or y) can be determined using the right-hand rule.
A coordinate frame can be attached to the end of the final link, within the end-effector or tool, or it may be necessary to locate this coordinate frame at the tool plate and have a separate hand transformation. The z-axis of the frame is in the same direction as the z-axis of the frame assigned to the last joint (n-1).
The end-effector or tool frame location and orientation is defined according to the controller conventions. Geometric parameters of length are defined to have an index of joint and

direction. The length *pni* is the distance between coordinate frames *i - 1* and *i* , and *n* is the parallel axis in the coordinate system *i - 1*. Figs. 2 and 3 shows the above rules applied to a PUMA-560 and an ABB IRB-2400 robots with all the coordinate frames and geometric features, respectively.

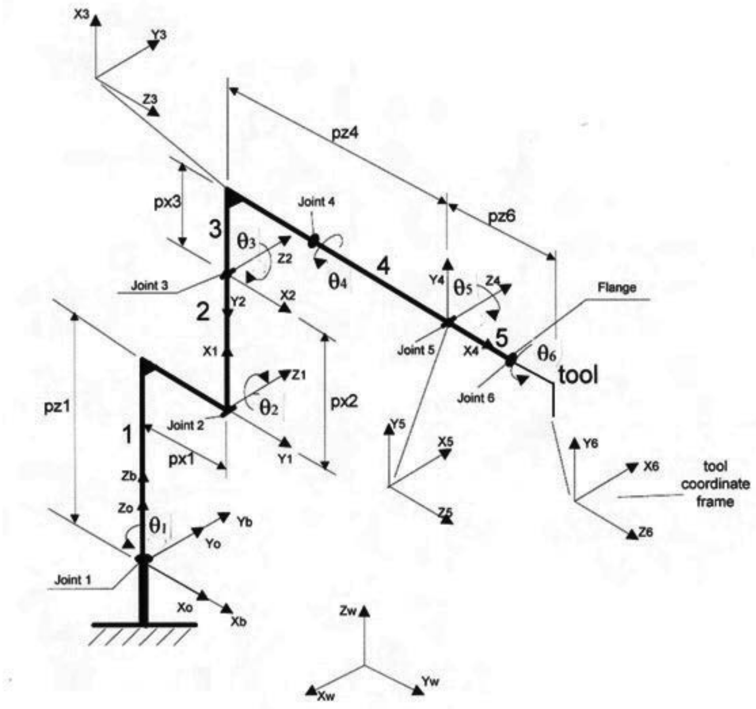


Fig. 3. Skeleton of the ABB IRB-2400 Robot with coordinate frames in the zero position and geometric variables for kinematic modeling. (Out of scale).

5. Kinematic Modeling – Parameter Identification

The kinematic equation of the robot manipulator is obtained by consecutive homogeneous transformations from the base frame to the last frame. Thus,

$$\hat{T}^0{}_N = \hat{T}^0{}_N(p) = T^0{}_1.T^1{}_2...T^{N-1}{}_N = \prod_{i=1}^{N} T^{i-1}{}_i \tag{35}$$

where N is the number of joints (or coordinate frames), $p = [p_1^T\ p_2^T\ ...\ p_N^T]^T$ is the parameter vector for the manipulator, and p_i is the link parameter vector for the joint i, including the joint errors. The exact link transformation $A^{i-1}{}_i$ is (Driels & Pathre, 1990):

$$A^{i-1}{}_i = T^{i-1}{}_i + \Delta T_i \quad , \quad \Delta T_i = \Delta T_i(\Delta p_i) \tag{36}$$

where Δp_i is the link parameter error vector for the joint i.
The exact manipulator transformation $\hat{A}^0{}_{N-1}$ is

$$\hat{A}^0{}_N = \prod_{i=1}^{N}(T^{i-1}{}_i + \Delta T_i) = \prod_{i=1}^{N} A^{i-1}{}_i \tag{37}$$

Thus,

$$\hat{A}^0{}_N = \hat{T}^0{}_N + \Delta\hat{T} \quad , \quad \Delta\hat{T} = \Delta\hat{T}(q, \Delta p) \tag{38}$$

where $\Delta p = [\Delta p_1{}^T\ \Delta p_2{}^T\ \dots\ \Delta p_N{}^T]^T$ is the manipulator parameter error vector and $q=[\theta_1{}^T, \theta_2{}^T\ \theta_N{}^T]^T$ is the vector of joint variables. It must be stated here that $\Delta\hat{T}$ is a non-linear function of the manipulator parameter error vector Δp.
Considering *m* the number of measure positions it can be stated that

$$\hat{A} = \hat{A}^0{}_N = \hat{A}(q, p) \tag{39}$$

where $\hat{A}: \Re^n \times \Re^N$ is function of two vectors with *n* and N dimensions, *n* is the number of parameters and N is the number of joints (including the tool). It follows that

$$\mathbf{\hat{A}} = \mathbf{\hat{A}}^0{}_N = \mathbf{\hat{A}}(\mathbf{q},p) = (\hat{A}(q_1,p),\dots, \hat{A}(q_m,p))^T\text{: } \Re^n \times \Re^{mN} \tag{40}$$

and

$$\Delta\mathbf{\hat{T}} = \Delta\mathbf{\hat{T}}(\mathbf{q}, \Delta p) = (\Delta\hat{T}(q_1, \Delta p),\dots,\Delta\hat{T}(q_m, \Delta p)^T \quad : \Re^n \times \Re^{mN} \tag{41}$$

All matrices or vectors in bold are functions of *m*. The identification itself is the computation of those model parameter values $p^*=p+\Delta p$ which result in an optimal fit between the actual measured positions and those computed by the model, i.e., the solution of the non-linear equation system

$$\mathbf{B}(\mathbf{q},p^*) = \mathbf{M}(\mathbf{q}) \tag{42}$$

where **B** is a vector formed with position and orientation components of $\mathbf{\hat{A}}$ and

$$\mathbf{M}(\mathbf{q}) = (M(q_1),\dots, M(q_m))^T \quad \in \Re^{\phi m} \tag{43}$$

are all measured components and ϕ is the number of measurement equations provided by each measured pose. If orientation measurement can be provided by the measurement system then 6 measurement equations can be formulated per each pose. If the measurement system can only measure position, each pose measurement can supply data for 3 measurement equations per pose and then B includes only the position components of $\hat{A}$.
When one is attempting to fit data to a non-linear model, the non-linear least-squares method arises most commonly, particularly in the case that *m* is much larger than *n* (Dennis & Schnabel, 1983). In this case we have from eq. (36), eq. (38) and eq. (42):

$$\mathbf{B}(\mathbf{q}, p^*) = \mathbf{M}(\mathbf{q}) = \mathbf{B}(\mathbf{q}, p) + \mathbf{C}(\mathbf{q}, \Delta p) \tag{44}$$

where **C** is the differential motion vector formed by the position and rotation components of $\Delta\mathbf{\hat{T}}$. From the definition of the Jacobian matrix and ignoring second-order products

$$\mathbf{C}(\mathbf{q}, \Delta p) = \mathbf{J}.\Delta p \tag{45}$$

and so,

$$\mathbf{M}(\mathbf{q}) - \mathbf{B}(\mathbf{q},p) = \mathbf{J}.\Delta p \tag{46}$$

The following notation can be used

$$\mathbf{b} = \mathbf{M}(\mathbf{q}) - \mathbf{B}(\mathbf{q},p) \in \Re^{\phi m} \tag{47}$$

$$\mathbf{J} = \mathbf{J}(\mathbf{q}, \Delta p) \in \Re^{\phi m \times n} \tag{48}$$

$$x = \Delta p \in \Re^{n} \tag{49}$$

$$\mathbf{r} = \mathbf{J}.x - \mathbf{b} \in \Re^{\phi m} \tag{50}$$

Eq. (10) can be solved by a non-linear least-square method in the form

$$\mathbf{J}.x = \mathbf{b} \tag{51}$$

One method to solve non-linear least-square problems proved to be very successful in practice and then recommended for general solutions is the algorithm proposed by Levenberg-Marquardt (LM algorithm) (Dennis & Schnabel, 1983). Several algorithms versions of the L.M. algorithm have been proved to be successful (globally convergent). From eq. (51) the method can be formulated as

$$x_{j+1} = x_j - \left[\mathbf{J}(x_j)^T.\mathbf{J}(x_j) + m_j.\mathbf{I}\right]^{-1}.\mathbf{J}^T(x_j).\mathbf{b}(x_j) \tag{52}$$

where, according to Marquardt suggestion, $\mu_j = 0.001$ if x_j is the initial guess, $\mu_j = \lambda(0.001)$ if $||b(x_{j+1})|| \geq ||b(x_j)||$, $\mu_j = 0.001/\lambda$ if $||b(x_{j+1})|| \leq ||b(x_j)||$ and λ is a constant valid in the range of $2.5 < \lambda < 10$ (Press et al., 1994).

6. Experimental Evaluation

To check the complete system to calibrate robots an experimental evaluation was carried out on an ABB IRB-2000 Robot. This robot was manufactured in 1993 and is used only in laboratory research, with little wearing of mechanical parts due to the low number of hours on work. The robot is very similar to the ABB IRB-2400 Robot, and the differences between both robots exists only in link 1, shown in Fig. 3, where p_{x1} turns to be zero.

6.1. Calibration Volumes and Positions

For this experimental setup different workspace volumes and calibration points were selected, aiming at spanning from large to smaller regions. Five calibration volumes were chosen within the robot workspace, as shown in Fig. 4. The volumes were cubic shaped. In Fig. 5 it is shown the calibration points distributed on the cubic faces of the calibration volumes. The external cubes have 12 calibration points (600mm) and the 3 internal cubes (600, 400 and 200mm) have 27 positions.

The measurement device used was a Coordinate Measuring Arm, (ITG ROMER), with 0,087mm of accuracy, shown in Fig. 6. The experimental routine was ordered in the following sequence: 1) robot positioning; 2) robot joint positions recorded from the robot controller (an interface between the robot controller and an external computer has to be available) and 3) robot positions recorded with the external measuring system. In this experiment only TCP positions were measured, since orientation measuring is not possible with the type of measuring device used. Only few measuring systems have this capacity and some of them are usually based on vision or optical devices. The price of the measuring system appears to be a very important issue for medium size or small companies.

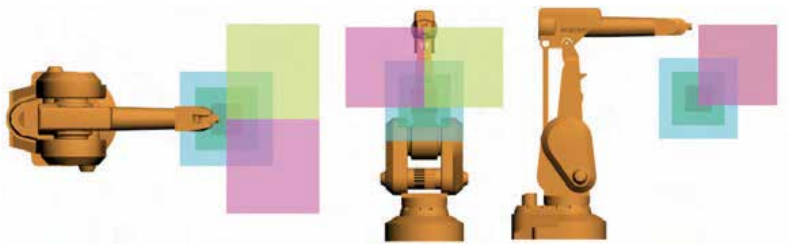


Fig. 4. Workspace Regions where the robot was calibrated.

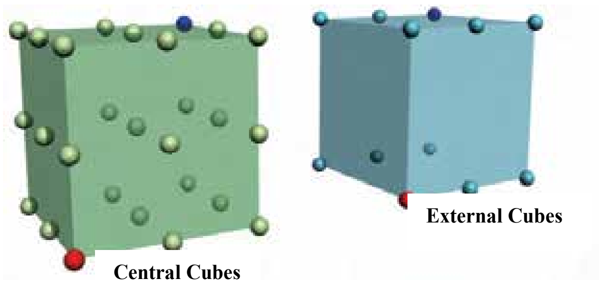


Fig. 5. Cubic calibration volumes and robot positions.

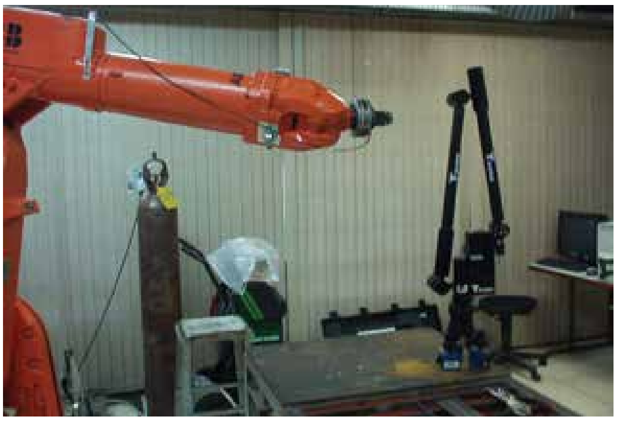


Fig. 6. Coordinate Measuring Arm - ITG ROMER and ABB IRB-2000 manipulator (University of Brasilia).

Fig. 7. represents graphically the calibration results within the regions and volumes of the workspace shown in Fig. 4 with the IRB-2000 Robot. The results presented show that the average of the position errors before and after calibration were higher when the Volumes were larger for both Regions tested. This robot was also calibrated locally, that means the robot was recalibrated in each Region.
A point that deserves attention is that if a robot is calibrated in a sufficiently large calibration volume, the position accuracy can be substantially improved compared to calibration with smaller joint motions. The expected accuracy of a robot in a certain task after calibration is analogous to the evaluation accuracy reported here in various conditions.

Every time a robot moves from a portion of the workspace to another, the base has to be recalibrated. However, in an off-line programmed robot, with or without calibration, that has to be done anyway. If the tool has to be replaced, or after an accident damaging it, it is not necessary to recalibrate the entire robot, only the tool. For that, all that has to be done is to place the tool at few physical marks with known world coordinates (if only the tool is to be calibrated not more than six) and run the off-line calibration system to find the actual tool coordinates represented in the robot base frame.

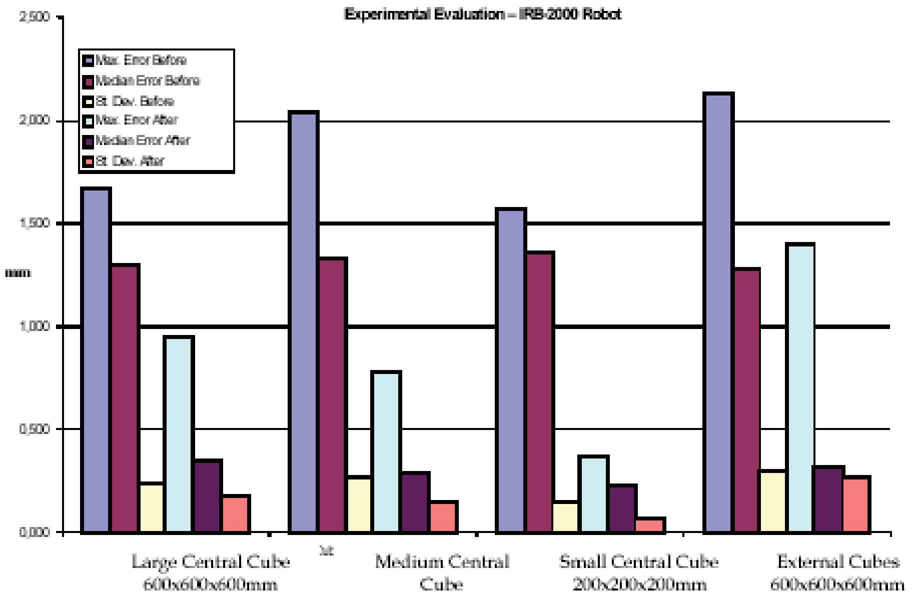


Fig. 7 Experimental evaluation of the robot model accuracy for positioning in each of the volumes.

8. A Vision-Based Measurement System

The main advantages of using a vision-based measurement system for robot calibration are: orientation measurements are feasible, measurement data can be easily recorded for further processing, good potential for high precision, measurements can be adjusted to the scale of the problem and it is a low cost system compared with the very expensive systems based on laser interferometry, theodolites and coordinate measuring arms.

A vision-based measurement system is described here, using a low cost CCD camera and a calibration board of points. The objective of this text is to describe the mathematical model and the experimental assessment of the vision system overall accuracy. The results show very good results for the application and a good potential to be highly improved with a CCD camera with more resolution and with a larger calibration board.

There are basically two typical setups for vision-based robot calibration. The first is to fix cameras in the robot surroundings so that the camera can frame a calibration target mounted on the robot end-effector. The other setup is named hand-mounted camera robot calibration. This latter setup can use a single camera or a pair of cameras. A single moving camera presents the advantages of a large field-of-view with a potential large depth-of-field, and a considerable reduced hardware and software complexity of the system. On the other

hand, a single camera setup needs full camera re-calibration at each pose.
The goal of camera calibration is to develop a mathematical model of the transformation between world points and observed image points resulting from the image formation process. The parameters which affect this mapping can be divided into three categories (Prescott & McLean, 1997, Zhuang & Roth, 1996): a) extrinsic (or external) parameters, which describe the relationship between the camera frame and the world frame, including position (3 parameters) and orientation (3 parameters); b) intrinsic (or internal) parameters, which describe the characteristics of the camera, and include the lens focal length, pixel scale factors, and location of the image center; c) distortion parameters, which describe the geometric nonlinearities of the camera. Some authors include distortion parameters in the group of intrinsic parameters (Tsai, 1987, Weng et al., 1992). Distortion parameters can be present in a model or not.
The algorithm developed here to obtain the camera parameters (intrinsic, extrinsic and distortions) is a two-step method based on the Radial Alignment Constraint (RAC) Model (see Tsai, 1987). It involves a closed-form solution for the external parameters and the effective focal length of the camera. Then, a second stage is used to estimate three parameters: the depth component in the translation vector, the effective focal length, and the radial distortion coefficient. The RAC model is recognized as a good compromise between accuracy and simplicity, which means short processing time (Zhuang & Roth, 1996). Some few modifications were introduced here in the original RAC algorithm, and will be explained later.

8.1 RAC-Based Camera Model

In Fig. 8 the world coordinate system is {xw, yw, zw}; the camera coordinate system is {x, y, z}. The origin of the camera coordinate system is centered at *Oc*, and the z-axis coincides with the optical axis. (X, Y) is the image coordinate system at *Oi* (intersection of the optical axis with the front image plane) and is measured in pixels. (u, v) are the analog coordinates of the object point in the image plane (usually measured in meters or millimeters). (X, Y) lies on a plane parallel to the *x* and *y* axes. *f* is the distance between the front image plane and the optical center.

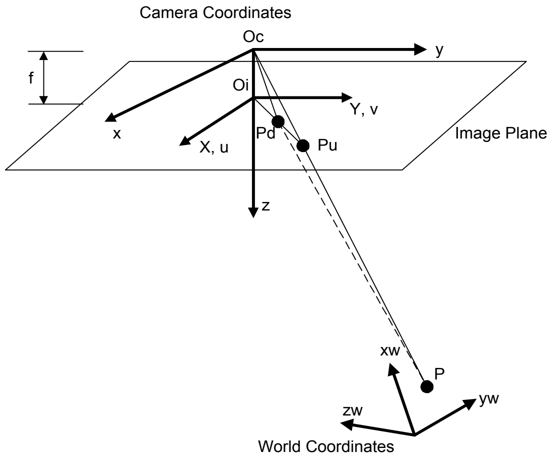


Fig. 8. Pin-hole model and the Radial Alignment Constraint hypothesis.

The rigid body transformation from the object coordinate system (xw, yw, zw) to the camera coordinate system (x, y, z) is

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = R. \begin{bmatrix} xw \\ yw \\ zw \end{bmatrix} + T \tag{53}$$

where R is the orthogonal rotation matrix, which aligns the camera coordinate system with the object coordinate system, and it can be represented as

$$R = \begin{bmatrix} r1 & r2 & r3 \\ r4 & r5 & r6 \\ r7 & r8 & r9 \end{bmatrix} \tag{54}$$

and T is the translation vector represented as

$$T = \begin{bmatrix} Tx \\ Ty \\ Tz \end{bmatrix} \tag{55}$$

which is the distance from the origin of the camera coordinate system to the origin of the object coordinate system represented in the camera coordinate system. (x_i, y_i, z_i) is the representation of any point (xw_i, yw_i, zw_i) in the camera coordinate system.
The distortion-free camera model, or "pinhole" model, assumes that every real object point is connected to its correspondent image on the image plane in a straight line passing through the focal point of the camera lens, *Oc*. In Fig. 8 the undistorted image point of the object point, *Pu*, is shown.
The transformation from 3-D coordinate (x, y, z) to the coordinates of the object point in the image plane follows the perspective equations below (Wolf, 1983):

$$u = f.\frac{x}{z} \text{ and } v = f.\frac{y}{z} \tag{56}$$

where *f* is the focal length (in physical units, e.g. millimeters).
The image coordinates (X,Y) are related to (u, v) by the following equations (Zhuang & Roth, 1996, Tsai, 1987):

$$X = s_u.u \text{ and } Y = s_v.v \tag{57}$$

where s_u and s_v are scale factors accounting for TV scanning and timing effects and converting camera coordinates in millimeter or meters to image coordinates (X,Y) in pixels. Lenz and Tsai (1987) relate the hardware timing mismatch between image acquisition hardware and camera scanning hardware, or the imprecision of the timing of TV scanning only to the horizontal scale factor s_u. In eq. (57), s_v can be considered as a conversion factor between different coordinate units.
The scale factors, s_u and s_v, and the focal length, *f*, are considered the intrinsic model parameters of the distortion-free camera model, and reveal the internal information about the camera components and about the interface of the camera to the vision system. The extrinsic parameters are the elements of R and T, which pass on the information about the camera position and orientation with respect to the world coordinate system.
Combining eqs. (56) and (57) follows

$$X = s_u.u = f.s_u.\frac{x}{z} = fx.\frac{x}{z} \tag{58}$$

$$Y = s_v.v = f.s_v.\frac{y}{z} = fy.\frac{y}{z} \tag{59}$$

The equations above combined with eq. (53) produce the distortion-free camera model

$$X = fx.\frac{r1.xw + r2.yw + r3.zw + Tx}{r7.xw + r8.yw + r9.zw + Tz} \tag{60}$$

$$Y = fy.\frac{r4.xw + r5.yw + r6.zw + Ty}{r7.xw + r8.yw + r9.zw + Tz} \tag{61}$$

which relates the world coordinate system (xw, yw, zw) to the image coordinate system (X,Y). *fx* and *fy* are non-dimensional constants defined in eqs. (58) and (59).
Radial distortion can be included in the model as (Tsai, 1987, Weng et al., 1992):

$$X.(1 + k.r^2) = fx.\frac{r1.xw + r2.yw + r3.zw + Tx}{r7.xw + r8.yw + r9.zw + Tz} \tag{62}$$

$$Y.(1 + k.r^2) = fy.\frac{r4.xw + r5.yw + r6.zw + Ty}{r7.xw + r8.yw + r9.zw + Tz} \tag{63}$$

where r = $\mu.X^2 + Y^2$ and μ is the ratio of scale to be defined further in the text. Whenever all distortion effects other than radial lens distortion are zero, a radial alignment constraint (RAC) equation is maintained. In Fig. 8, the distorted image point of the object point, Pd, is shown.
Eqs. (62) and (63) can be linearized as (Zhuang & Roth, 1993):

$$\frac{X}{1 - k.r^2} \cong fx.\frac{r1.xw + r2.yw + r3.zw + Tx}{r7.xw + r8.yw + r9.zw + Tz} \tag{64}$$

$$\frac{Y}{1 - k.r^2} \cong fy.\frac{r1.xw + r2.yw + r3.zw + Ty}{r7.xw + r8.yw + r9.zw + Tz} \tag{65}$$

This transformation can be done under the assumption that $k.r^2 << 1$, i.e. $(1 + k.r^2) \cong (\frac{1}{1 - k.r^2})$ when $k.r^2 << 1$. The first stage to solve eqs. (64) and (65) determines the rotation matrix R (eq. 54), Ty and Tx. The algorithm for that is found in Zhuang & Roth (1996) and Tsai (1987): The second stage is proposed here as

$$\begin{bmatrix} -X_i & x_i & -x_i.r_i^2 \\ -Y_i & \mu.y_i & -\mu.y_i.r_i^2 \end{bmatrix} \cdot \begin{bmatrix} Tz \\ fx \\ k.fx \end{bmatrix} = \begin{bmatrix} X_i.w_i \\ Y_i.w_i \end{bmatrix} \tag{66}$$

where $x_i = r1.xw_i + r2.yw_i + Tx$, $y_i = r4.xw_i + r5.yw_i + Ty$, $w_i = r7.xw_i + r8.yw_i$, and zw_i is made null (all calibration points are coplanar). The overdetermined linear system in eq. (66) can be solved by a linear least-square routine. The calibration points used to solve the system above were on the border of a square grid of 9x9 circular points with 1mm diameter, totting up 32 points. The grid has approximately 200x200mm. The angle between the calibration plane and the image plane should not be smaller than 30 degrees to avoid ill conditioned solutions.
The second stage of the algorithm originally proposed by Tsai (1987) was not a linear solution as it was based on eq.(63), with components only in the *y* direction. Tsai's arguments were based on the fact that *y* components were not contaminated by the mismatch between the frame grabber and camera frequencies. However, it was

experimentally observed during this research that because of the various angles and distances that the images of the target plate were to be acquired from, a linear solution considering both directions (*x* and *y*) showed to be substantially more accurate than using only one direction. The improvement in accuracy of the solutions using eq. (66) was evaluated comparing three different images at different orientations and similar distances, and observing the focus length calculated from the second stage of the RAC algorithm, using eq. (63) (only *y* components), eq. (64) (only *x* components), and eq. (66) (both *x* and *y* components). Ideally, the focus length has to be the same for all images whatever orientation the camera is placed. The use of eq. (66) resulted in considerably closer values of *fx* for the three images than using only one direction (*x* or *y*).

8.2 Calibration of the Scale Factors

A ratio of scale is defined as

$$\mu = \frac{fy}{fx} = \frac{s_v}{s_u} \tag{67}$$

and dividing eq. (64) by (65) yields

$$\frac{X}{Y} = \mu^{-1} . \frac{r1.xw + r2.yw + r3.zw + Tx}{r4.xw + r5.yw + r6.zw + Ty} \tag{68}$$

Two-step camera calibration techniques such as the RAC model can be used to determine the ratio of scale accurately if more than one plane of calibration points is used. This is accomplished by moving a single-plane calibration setup vertically with a z-stage (Zhuang & Roth, 1993). However, if only one coplanar set of calibration points is to be used, pre-calibration of the horizontal scale factor is essential.

Based on the fact that using non-coplanar and parallel planes μ can be determined from two-step calibration methods (Tsai, 1987), several images were obtained from different orientations and positions of the camera. This was accomplished when the camera was mounted on the robot hand during robot calibration measurements. Using the RAC model residuals as calculated from eq. (68) (Lenz & Tsai, 1987, Zhuang et al., 1993):

$$r4.X_i.xw_i + r5.X_i.yw_i + X_i.Ty - r1.Y_i.xw_i - r2.Y_i.yw_i - Y_i.Tx = 0 \tag{69}$$

where *i* is the index representing each calibration point in the image, and based on the fact that □□and the image center are quite independent from each other in the RAC model (a poor guess for one does not affect the determination of the optimal value for the other), the residues found for each image were averaged and then μ was searched to minimize the average of the residuals. The solution found was μ = 0.9825.

8.3 Calibration of the Image Center

The image center is defined as the frame buffer coordinates (Cx, Cy) of the intersection of the optical axis with the image plane. It is usually used as the origin of the imaging process and appears in the perspective equation. In high accuracy applications, it also serves as the center of radially modeled lens distortion (Zhuang & Roth, 1993). The image center is defined from

$$Cx = Xf - X \text{ and } Cy = Yf - Y \tag{70}$$

where (Xf, Yf) is the computed image coordinates for an arbitrary point and (Cx, Cy) is the computed image coordinates for the center Oi in the image plane.

The Radial Alignment Constraint model holds true and is independent of radial lens distortion when the image center is chosen correctly. Otherwise, a residual exists and, unfortunately, the RAC is highly non-linear in terms of the image center coordinates.
The method devised to find the image center was to search for the "best" image center as an average of all images in a sequence of robot measurements, using the RAC residuals. This method could actually find the optimal image center in the model, which could be easily checked calculating the overall robot errors after the calibration. The tests with a coordinate milling machine (explained further in the text) also showed that the determination of the image center by this method led to the best measurement accuracy in each sequence of constant camera orientation.

9 Measurement Accuracy Assessment

The evaluation of the accuracy obtained by the vision measurement system was carried out using a Coordinate Milling Machine (CMM) or any other device that can produce accurate motion on a plane. In this case a CCD camera (Pulnix 6EX – 752x582 pels) was fixed on the CMM's table and moved on pre-defined paths. The calibration board was fixed in front of the camera externally to the CMM, at a position that could allow angles from the camera optical axis to the normal of the calibration plane to be higher than 30^o degrees (avoiding ill-conditioned solutions in the RAC model).
The CMM's table motion produced variations in *z*, *x* and *y* axes of the target plate relative to the camera coordinate system. Fig. 9 shows the coordinate systems of the camera and calibration board.
There were three different measurement sequences of the 25 camera positions. Each sequence was performed with the calibration board placed at different distances and orientations from the camera. The distances were calculated using the photogrammetric model.
The calibration of the remaining intrinsic parameters of the camera, *fx* and *k*, was performed using the first sequence of 25 camera positions, which was chosen to be the closer to the target plate. For each image, values of *fx* and *k* calculated by the algorithm were recorded. Due to noise and geometric inaccuracies each image yielded different values for *fx* and *k*. The average values of the constants *fx* and *k* calculated from the 25 images were 1523 and 7.9 x 10^{-8} respectively. The standard deviation values for *fx* and *k* were 3.38 and 2.7 x 10^{-9} respectively.

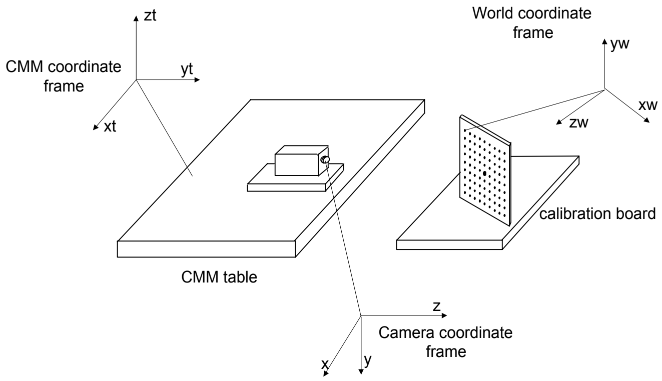


Fig. 9. Diagram showing coordinate systems and the experimental setup for the evaluation of the measurement system accuracy.

The average values for *fx* and *k* should be kept constant from then on, and considered the "best" ones. To check this assumption, the distance traveled by the camera calculated by the Euclidean norm of (XW, YW, ZW) for each camera position, and the distance read from the CMM display were compared to each other. Errors were calculated to be the difference between the distance traveled by the camera using the two methods. It was observed then that the average value of *fx* did not minimize the errors.
Subsequently, an optimal value for *fx* was searched to minimize the errors explained above, by changing slightly the previous one. The value of *k* did not show an important influence on the errors when changed slightly. The optimal values of *fx* and *k* were checked in the same way for the other two measurement sequences at different distances, and showed to produce the best accuracy. The optimal values for *fx* and *k* where found to be 1566 and 7.9 x 10^{-8}, which were kept constant from then on.
Assessment of 3-D measurement accuracy using a single camera is not straightforward. The method designed here to assess the measurement accuracy was to compare each component XW, YW and ZW corresponding to each camera position, to the same vector calculated assuming an average value for the rotation matrix. This method is justified considering that, since there are no rotations between the camera and the calibration board during the tests, the rotation matrix must be ideally the same for each camera position. Since the vector (XW, YW, ZW) depends on R in eq. (54) and T in eq. (55), and as T is calculated from R, all measurement errors would be accounted for in the rotation matrix, apart from the errors due to the ground accuracy of the calibration board. Of course, this assumption is valid only if another method to compare the camera measurements to an external measurement system (in physical units) is used to validate it. That means, the accuracy calculated from the traveled distance validates the accuracy calculated from the rotation matrix. The first method does not consider each error component independently (x, y, z), but serves as a basis to optimize the camera parameters. The second method takes into account the 3-D measurement error, assuming that the "correct" rotation matrix is the average of the ones calculated for each of the 25 positions. The maximum position errors for each measurement sequence were 0.43, 0.97, and 0.72mm respectively. Fig. 10 shows graphically the average, median and standard deviation values of the measurement system position accuracy calculated as explained before, as a function of the average distance from the camera (optical center) to the central point of the calibration board.

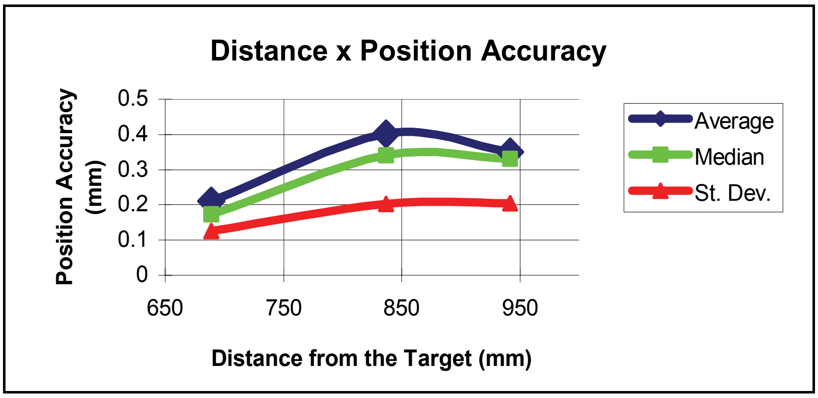


Fig. 10. Measurement system 3-D position accuracy versus the average distance from the camera focal point to the central point of the target.

10 Robot Calibration and Experimental Results

Within the IRB-2400 robot workspace three calibration Regions were defined to collect data, each one with a different Volume (V1, V2 and V3). Fig. 11 represents graphically the three Regions within the robot workspace.

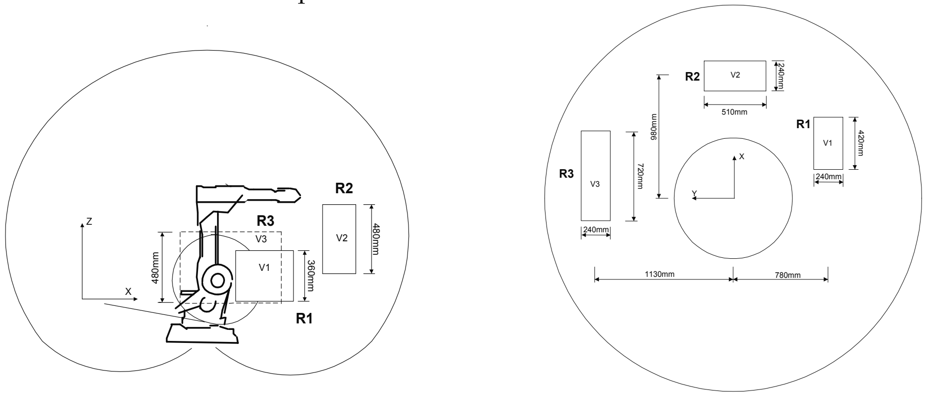


Fig. 11. Side and top view of the IRB-2400 Robot workspace showing Regions, Volumes and their dimensions and locations.

The results from the calculation of average errors and their standard deviation in each Region can be seen in the graphs shown in Fig. 12, calculated before and after the calibration. For calculated and measured data in different coordinate systems to be compared to each other, the robot base coordinate frame was moved to coincide with the world coordinate system at the measurement target plate. This procedure was carried out through a recalibration of the robot base in each Region.
The results presented in Fig. 12 show that the average of the position errors after calibration were very close in value for the three calibration regions tested.

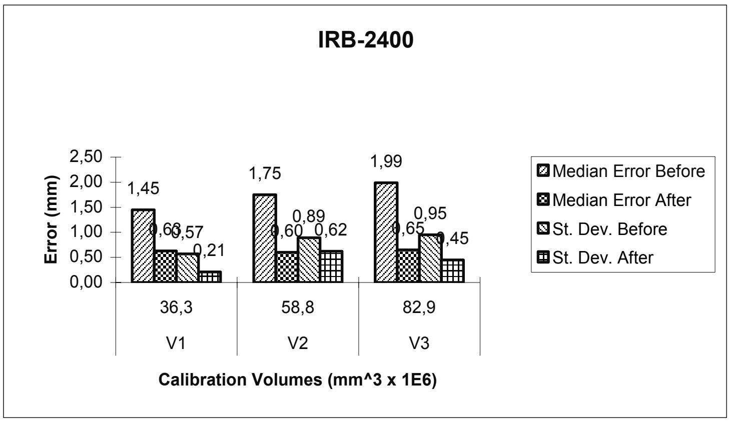


Fig. 12. Average Error and Standard Deviation calculated before and after calibration in each Region.

Within the PUMA-500 robot workspace two calibration Regions were defined to collect data. In each Region three Volumes were defined with different dimensions. Once two different Volumes had the same volume they had also the same dimensions, whatever Region they were in.
Fig. 13 represents graphically all Regions and Volumes within the PUMA-500 workspace.

The results presented in Figs. 15 and 16 show that the average of the position errors before and after calibration were higher when the Volumes were larger for both Regions tested. This robot was also calibrated locally, that means the robot was recalibrated in each Region.

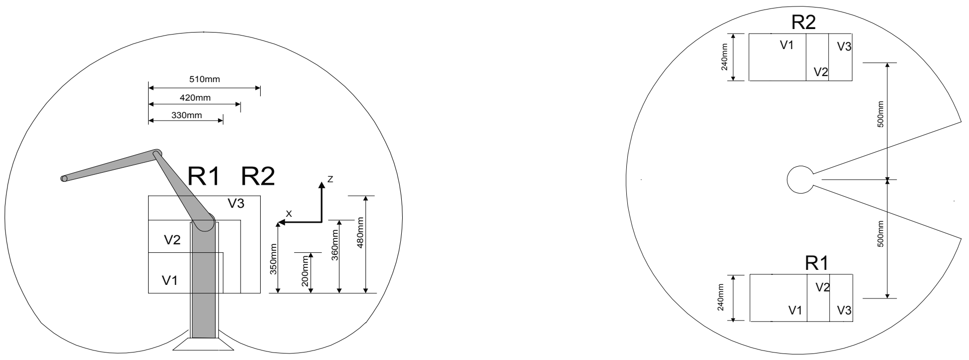


Fig. 13. Side and top view of the PUMA-500 Robot workspace showing Regions, Volumes and their dimensions and locations.

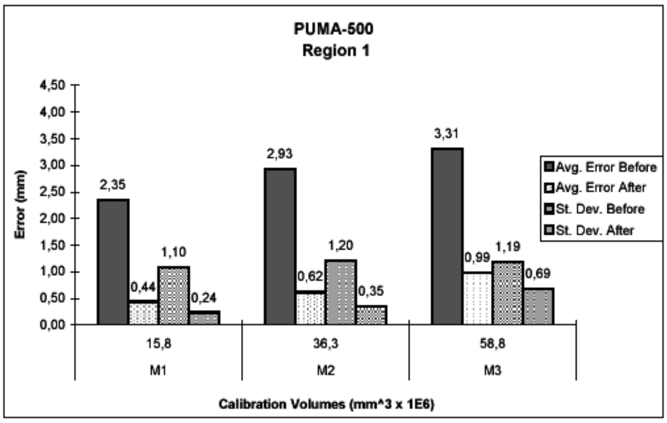


Fig. 14. Average Error and Standard Deviation calculated before and after calibration in each Volume in Region 1.

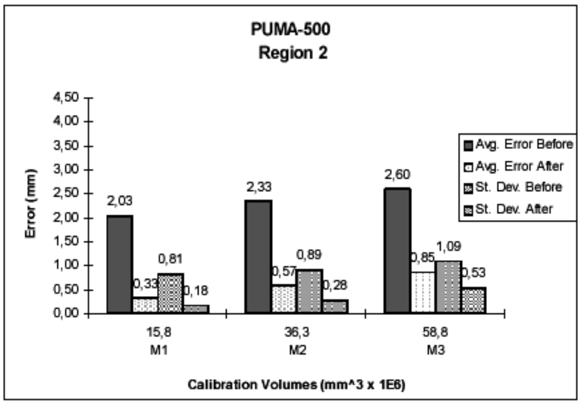


Fig. 15. Average Error and Standard Deviation calculated before and after calibration in each Volume in Region 2.

11. Conclusions and Further Work

The calibration system proposed showed to improve the robot accuracy to well below 1mm. The system allows a large variation in robot configurations, which is essential to proper calibration.
A technique was used and a straightforward convention to build kinematic models for a manipulator was developed, ensuring that no singularities are present in the error model. Mathematical tools were implemented to optimize the kinematic model parameterization, avoiding redundancies between parameters and improving the parameter identification process. A portable, ease of use, speedy and reliable Vision-based measuring system using a single camera and a plane calibration board was developed and tested independently of the robot calibration process.
The robot calibration system approach proposed here stood out to be a feasible alternative to the expensive and complex systems available today in the market, using a single camera and showing good accuracy and ease of use and setup. Results showed that the RAC model used (with slight modifications) is not very robust, since even for images filling the entire screen and captured at approximately the same distances from the target, the focus length was not constant and showed an average value shifted by approximately 3% from the exact one. This amount of error can produce 3-D measurement errors much larger than acceptable. Practically speaking, the solution for this problem developed here for a set of camera and lens was to use an external measurement system to calibrate the camera, at least once. The measurement accuracy obtained is comparable to the best found in academic literature for this type of system, with median values of accuracy of approximately 1:3,000 when compared to the distance from the target. However, this accuracy was obtained at considerable larger distances and different camera orientations than usual applications for cameras require, making the system suitable for robotic metrology.
For future research it is suggested that the target plate and the calibration board have to be improved to permit the camera to be placed at larger ranges of distances from the target, allowing larger calibration volumes to be used. One path that might be followed is to construct a much larger calibration board, with localized clusters of calibration points of different sizes, instead of just one pattern of point distribution. So, if the camera is placed at a greater distance, larger dots can be used all over the area of the calibration board. If the camera is nearer to the target, smaller dots can be used at particular locations on the calibration board. Different dot sizes make easier for the vision processing software to recognize desired clusters of calibration points.
Other sources of lens distortions such as decentering and thin prism can be also modeled, and so their influence on the final measurement accuracy can be understood.
Another issue concerns the influence orientation measured data may have on the final accuracy. Non-geometric parameters such as link elasticity, gear elasticity and gear backlash might be modeled, and a larger number of parameters introduced in the model parameterization. This procedure may improve the accuracy substantially if the robot is used with greater payloads.

12. References

Bai, Y and Wang, D. (2004). Improve the Robot Calibration Accuracy Using a Dynamic Online Fuzzy Error Mapping System, *IEEE Transactions on System, Man, And Cybernetics – Part B: Cybernetics*, Vol. 34, No. 2, pp. 1155-1160.

Baker, D.R. (1990). Some topological problems in robotics, *The Mathematical Intelligencer*,Vol.12, No.1, pp. 66-76.

Bernhardt, R. (1997). Approaches for commissioning time reduction, *Industrial Robot*, Vol. 24, No. 1, pp. 62-71.

Dennis JE, Schnabel RB. (1983). *Numerical Methods for Unconstrained Optimisation and Non-linear Equations*, New Jersey: Prentice-Hall.

Driels MR, Pathre US. (1990). Significance of Observation Strategy on the Design of Robot Calibration Experiments, *Journal of Robotic Systems*, Vol. 7, No. 2, pp. 197-223.

Drouet, Ph., Dubowsky, S., Zeghloul, S. and Mavroidis, C. (2002). Compensation of geometric and elastic errors in large manipulators with an application to a high accuracy medical system, *Robotica*, Vol. 20, pp. 341-352.

Everett, L.J. and Hsu, T-W. (1988). The Theory of Kinematic Parameter Identification for Industrial Robots, *Transaction of ASME*, No. 110, pp. 96-100.

Gottlieb, D.H. (1986). Robots and Topology, *Proceedings of the IEEE International Conference on Robotics and Automation*, pp. 1689-1691.

Hayati, S. and Mirmirani, M., (1985). Improving the Absolute Positioning Accuracy of Robots Manipulators, *Journal of Robotic Systems*, Vol. 2, No. 4, pp. 397-413.

Lenz RK, Tsai RY. (1987). Techniques for Calibration of the Scale Factor and Image Centre for High Accuracy 3D Machine Vision Metrology, *IEEE Proceedings of the International Conference on Robotics and Automation,* pp. 68-75.

McKerrow, P. J. (1995). *Introduction to Robotics*, 1st ed., Ed. Addison Wesley, Singapore.

Motta, J. M. S. T. and McMaster, R. S. (1999). Modeling, Optimizing and Simulating Robot Calibration with Accuracy Imporovements, *Journal of the Brazilian Society of Mechanical Sciences*, Vol. 21, No. 3, pp. 386-402.

Motta, J. M. S. T. (1999). *Optimised Robot Calibration Using a Vision-Based Measurement System with a Single Camera*, Ph.D. thesis, School of Industrial and Manufacturing Science, Cranfield University, UK.

Motta, J. M. S. T., Carvalho, G. C. and McMaster, R. S. (2001), Robot Calibration Using a 3-D Vision-Based Measurement System With a Single Camera, *Robotics and Computer Integrated-Manufacturing*, Ed. Elsevier Science, U.K., Vol. 17, No. 6, pp. 457-467.

Park, E. J., Xu, W and Mills, J. K. (2002). Calibration-based absolute localization of parts for multi-robot assembly, *Robotica*, Vol. 20, pp. 359-366.

Paul, R. P., (1981). *Robot Manipulators - Mathematics, Programming, and Control*, Boston, MIT Press, Massachusetts, USA.

Prescott B, McLean GF. (1997). Line-Based Correction of Radial Lens Distortion, *Graphical Models and Image Processing,* Vol. 59, No. 1, pp. 39-47.

Press WH, Teukolsky SA, Flannery BP, Vetterling WT. (1994). *Numerical Recipes in Pascal - The Art of Scientific Computer*, New York: Cambridge University Press.

Roth, Z.S., Mooring, B.W. and Ravani, B. (1987). An Overview of Robot Calibration, *IEEE Journal of Robotics and Automation*, RA-3, No. 3, pp. 377-85.

Schröer, K. (1993). Theory of kinematic modeling and numerical procedures for robot calibration, *Robot Calibration*, Chapman & Hall, London.

Schröer, K., Albright, S. L. and Grethlein, M. (1997), Complete, Minimal and Model-Continuous Kinematic Models for Robot Calibration, *Robotics & Computer-Integrated Manufacturing*, Vol. 13, No. 1, pp. 73-85.

Tsai RY. (1987). A Versatile Camera Calibration Technique for High-Accuracy 3D Machine Vision Metrology Using Off-the Shelf TV Cameras and Lenses. *IEEE International Journal of Robotics and Automation*, RA-3, No. 4, pp. 323-344.

Veitscheggar, K. W. and Wu, C. W. (1986). Robot Accuracy Analysis based on Kinematics. *IEEE Journal of Robotics and Automation*, Vol. 2, No. 3, pp. 171-179.

Weng J, Cohen P, Herniou M. (1992). Camera Calibration with Distortion Models and Accuracy Evaluation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 14, No. 10, pp. 965-980.

Wolf PR. (1983). *Elements of Photogrammetry*, McGraw-Hill, Singapore.

Zhuang H, Roth ZS, Xu X, Wang K. (1993). Camera Calibration Issues in Robot Calibration with Eye-on-Hand Configuration. *Robotics & Computer Integrated Manufacturing*, Vol. 10, No. 6, pp. 401-412.

Zhuang H, Roth ZS. (1993). A Linear Solution to the Kinematic Parameter Identification of Robot Manipulators. *IEEE Transactions on Robotics and Automation*, Vol. 9, No. 2, pp. 174-185.

Zhuang H, Roth ZS. (1996). *Camera-Aided Robot Calibration*, CRC Press, Boca Raton.Fla, USA.

Zhuang, H. (1992). A Complete and Parametrically Continuous Kinematic Model for Robot Manipulators, *IEEE Transactions on Robotics and Automation*, Vol. 8, No. 4, pp. 451-63.

Accuracy and Calibration Issues of Industrial Manipulators

Mohamed Abderrahim, Alla Khamis, Santiago Garrido, Luis Moreno
University Carlos III of Madrid,
Spain

1. Introduction

Since initial stages of the adoption of industrial robot, the reason for their development was to replace humans in repetitive tedious and hazardous manual work. However, due to rise of production and falling prices, the use of robots becomes an economically viable solution in more and more applications (Hefele & Brenner, 2001). Nowadays, industrial robot manipulators are an important component of most automated manufacturing systems, and especially in the automotive industry. They have been used for many years in industry efficiently in mass production applications, where simple sequences of movements are pre-recorded in the memory of the robot controller. Nevertheless, manipulators are still rarely used in industrial applications with frequently changing operations where continuous reprogramming is needed, since this requires teaching the points for each operation and specifying the trajectories for the new sequences of motion using robots teach-pendants. This problem can be overcome only if the programs and the points are designed off-line using manipulators' models and simulated work cells. The implementation of off-line programming (OLP) requires very good accuracy of manipulators, which can be achieved by a process of robot calibration to reduce the effects of the kinematic model errors due to the manufacturing tolerances of the robot elements and the assembly processes, in addition to changing working environment conditions and effects of tear and wear. This means that the model of the manipulator in the controller should be a precise representation of the mechanisms and allows the system to achieve the desired manipulator *pose* (position and orientation) with minimum offsets.
Calibration is therefore used to improve robot positioning accuracy, through software rather than changing the mechanical structure or design of the robot itself (Elatta et al., 2004). Since robots are mechanical devices, they can be affected by slight changes due to manufacturing tolerances, changes of working environment temperature, wear of parts and component replacement (Elatta et al. 2004). Calibration is therefore meant to minimise these effects and avoid having to modify operations' programs and allows the use of off-line programming and effective use of manipulators in more industrial and non industrial applications.
To achieve high accuracy, the kinematic model used in the controller, should be a faithful mathematical description of the relationship, which relates the end-effector pose to the individual joint values. Ideally, this model should account for geometrical and no

geometrical parameters, which can affect this relationship. However, all industrial manipulator controllers in the market contain ***nominal*** kinematic representations which are common to all manipulators of the same model. It is for this reason that improving manipulator accuracy passes through improving the kinematic model, by generating the *real* kinematic parameters for each particular manipulator after its assembly.
The work described in this chapter focuses on kinematic model improvement as a means of improving the accuracy of robots (Calibration) without having to impose stricter manufacturing tolerances, and presents an overview of the major research achievements in the field. The chapter discusses the issue of calibration in general and achieved research results, while it describes the procedure developed by the authors in previous work (Abderrahim & Whittaker, 2000) for the direct identification of the D-H kinematic model parameters. In addition the chapter describes the recent developments concerning the improvement of robot manipulators accuracy, which did make their way through to the market and made "*Absolute Accuracy*" of industrial robots a reality nowadays ((Renders, 2006) and (Fixel, 2006)).
The remainder of this chapter is organized as follows. The next section describes the repeatability and the accuracy as the most important values that determine the precision performance of an industrial manipulator. Error sources are also highlighted in this section. Section 3 gives an overview on industrial manipulators calibration followed by describing in details the calibration methods in section 4. Examples for the most commonly used robot simulation software are presented in section 5. In section 6, two examples for commercial calibration procedures are described and finally conclusions are summarized in section 7.

2. Accuracy and Error Sources

The important two values describing the precision performance of manipulator specified in the international standard ISO 9283, which sets the performance criteria of industrial manipulator are the *pose repeatability* and *pose accuracy*. Until recently and given that most of manipulator programming has been done on-line, where command points are acquired in teach mode, the most relevant performance specification was the repeatability. Pose repeatability is a measure of the ability of the robot to move back to the same position and orientation. Pose Accuracy, on the other hand, is defined as the ability of the robot to precisely move to a desired position in 3-D space. Fig. 1 illustrates the repeatability and accuracy graphically, by showing how close and scattered the achieved positions relative to the desired location at the centre of the circles.
The achievable repeatability of current industrial manipulators is very high and reaches values bellow 0.1 mm by most robots, while standard robot accuracy can range between 5 and 15 mm (Fixel, 2006) and can be higher depending on the make and model (Abderrahim & Whittaker, 2000). The repeatability errors are mainly caused by the inability of the controller to achieve exactly the same joint values at different runs, and the possible presence of backlash in the manipulator reduction gears. However, the latter effect is very unlikely due to the availability nowadays of zero-backlash gears. The pose accuracy errors are affected by numerous geometric and non-geometric parameters. The accuracy involves using the inverse kinematic model for the calculation of the joint values that correspond to the commanded 3D pose and therefore errors are caused by defects in this model. The main geometric parameters that affect this model are the dimensions of the manipulator links and

orientation of the joints. During the manufacturing, the parts variation of dimensions is inevitable from one robot to the other due to tolerances. According to (Kevin et al., 2000), in revolute 6 DOF manipulators the length of first 3 links contribute to the position and joints 4, 5, and 6 (wrist) contribute primarily to the orientation of the tool frame. Other parameters that are not included in kinematic models and affect the accuracy include: Elasticity of joints and links, load proportion and joint deflection, actuator heating, sensors stability and drifts, gearbox flexibility and eccentricity and environment working temperature (Hefele & Brenner, 2001). In addition, end-effector fixing position and the determination of the world coordinate frame also affect the pose accuracy in the work environment.

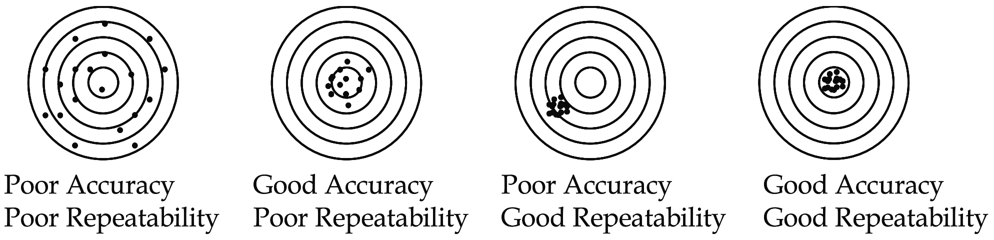


Fig. 1. Achieved position according to repeatability and accuracy qualities.

3. Calibration Overview

There has been a lot of work on the subject of improving the positioning accuracy of industrial robot manipulators ((Hayati & Mirmirani, 1985), (Mooring & Pack, 1987), (Driels & Swaze, 1994) and (Roth et al., 1987) among many others). Most of the authors considered the main source of errors to be only geometric, with the exception of (Chen at al., 1987) and Whitney et al. (Whitney et al., 1986), who included explicitly non-geometric errors as well. Although some introduced their own models (Whitney et al., 1986), the majority used models that are universally valid and widely accepted, such as the Denavit-Hartenberg or modified versions of it ((Khalil et al., 1991), (Driels, 1993) and (Harb & Burdekin, 1994)). Much previous work is based only on computer simulations, but some validation work used real measurements ((Chen at al., 1987), (Whitney et al., 1986), (Stone, 1987) (Stanton & Parker, 1992) (Khalil et al., 1991),(Driels, 1993),(Driels & Pathre, 1994) and (Abderrahim & Whittaker, 2000)), and very recently there is even commercial application dedicated to robot calibration ((Renders, 2006) and (Fixel, 2006)). Stone (Stone, 1987) developed a model and also a novel general identification method to estimate his S-model parameters, based on the circle-point analysis (Mooring et al., 1991) which lead to what he designated as *joint features*. These joint features are the *plane of rotation, centre of rotation* and *radius of rotation*. If required, the D-H parameters can then be extracted from the S model. Stone's motivation for developing his model was because "the D-H Model is not amenable to direct identification" (Stone, 1987). However, with small modification, the D-H Model has been used successfully for calibration in several of the papers referred to above. Although the S-Model can be identified using the method proposed in (Stone, 1987), it still suffers the same disproportion as the D-H model when consecutive joint axes are parallel or near parallel. The real problem here is that the control software of existing industrial manipulator does not use the S-model parameters. The physically explicit significance of the D-H model and its widespread use in robot control software make it very desirable to develop identification methods that obtains the D-H parameters directly, whilst at the same time is able to deal with the case where consecutive joint axes are parallel. Our procedure proposed in (Abderrahim & Whittaker, 2000) makes use of the features introduced by Stone, but differs from Stone's work by making more use of the *radius of rotation* and also translating the *plane of rotation* along the *axis of rotation* to make it coincide with the D-H X-Y plane. Using the idea developed by (Hayati & Mirmirani, 1985), a new parameter was introduced to modify the model to be able to deal with the disproportion at parallel or near parallel joint axes. In this way the proposed method can be viewed as a combination of Stone's and Hayati's methodologies. In the method, the position of the end-effector caused by the movement of only one joint axis is measured, and the process is repeated for each joint in an inward sequence, starting from joint n and ending at joint 1. In a similar sequence the measured Cartesian positions of the end-effector are then used for the estimation of the features for each link (joint). These are in turn used for the estimation of link parameters.

4. Calibration Methods

Robot calibration is a process by which manipulator real parameters' values affecting its accuracy are established through direct or indirect measurement, and used to improve its accuracy by modifying the positioning software. The calibration procedure involves modelling, measurement, parameter identification and implementation of compensation. The first step is establishing a model to relate the 3D Cartesian position of the end-effector in terms of joint values and all other kinematic *parameters* that affect this function. The next step is to perform an external accurate measuring of the end-effector Cartesian pose corresponding to each set of joint values. The measurement should be performed in a number of positions sufficient for the process of identification. The identification method is chosen according to the model and how the measurements were taken. Using the set poses and the corresponding joint angles, the true kinematic values should be estimated through and identification process. These new parameters, which deviate from their nominal values, shall improve the manipulators accuracy when implemented in the controller. It has to be noted that in the case of establishing the non-geometric sources of errors and evaluating their effects a similar approach and steps have to be followed. Nowadays, calibration techniques and absolute accuracy are provided by major manipulator manufacturers such as ABB (Fixel, 2006) and KUKA robots. Access to modify the controller is not obvious, and may not be possible for third parties. In this case, the alternative method of compensation can be established off-line, where compensated end-effector Cartesian positions are calculated using the "real" estimated inverse kinematic model. The latter method is offered nowadays by calibration and metrology equipment manufacturers such as Metris (Metris, 2005). In the rest of this section we develop the mentioned steps covering the method described in our previous work and other new developments by others with emphasis on existing industrial solutions.

4.1. Modelling

Coordinate frame kinematic models are based upon the assignment of Cartesian coordinate frames fixed relative to each link of the robot arm. The Denavit-Hartenberg (D-H) model has been widely adopted in robotics due to its explicit physical interpretation of the mechanisms and the relatively easy implementation in the programming of robot manipulators. Other models are mentioned in the overview section above, where the mentioned references can be consulted for in-depth information. Since the D-H model was used in our work and due to its practical relevance it is the only one covered in this section. The D-H model is presented as 4X4 matrix that results from the following product:

$$T = A_1 \cdot A_2 \cdot \ldots \cdot A_n \ , \tag{1}$$

where T denotes the transformation of the link n coordinate frame into the base coordinate frame of the robot arm. A_i designates the D-H transformation matrix relating frame i to frame i-1. The nth link frame coincides with the end-effector's coordinate frame. Fig. 2 illustrates the spatial relative position of two consecutive links and their associated coordinate frames.

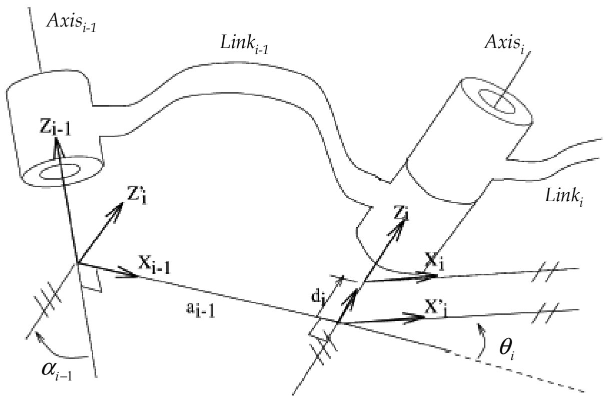


Fig. 2. The D-H parameters illustration in case of a revolute joint.

The spatial transformation between two consecutive links is a function of the joint type that connects them together. It is caused by a number of rotations and translations summarised by

$$A_i = A_i(q_i) \equiv Rot(z,\theta_i)\ Trans(0,0,d_i)\ Trans(a_{i-1},0,0)\ Rot(x,\alpha_{i-1}) \tag{2}$$

$$A = \begin{bmatrix} \cos\theta_i & -\sin\theta_i \cos\alpha_{i-1} & \sin\theta_i \sin\alpha_{i-1} & a_{i-1}\cos\theta_i \\ \sin\theta_i & \cos\theta_i \cos\alpha_{i-1} & -\cos\theta_i \sin\alpha_{i-1} & a_{i-1}\sin\theta_i \\ 0 & \sin\alpha_{i-1} & \cos\alpha_{i-1} & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3}$$

When this model is used as a basis for kinematic and inverse kinematic calculations the errors affecting the parameters, a_i, d_i, α_i and θ_i should be estimated during the calibration process. If non-geometric effects such as temperature and joint compliance are considered, then the model should account for them directly or indirectly.

4.2. Measurement

This section is dedicated to describe the systems that are capable of providing accurate measurements fit for application in the calibration process. The first system described is the system used in our previous work (Abderrahim & Whittaker, 2000), and the second is chosen because it represents one of the few existing industrial effective solution in the market.

During the mentioned work the instrument used to measure the Cartesian end-effector position of the Puma manipulator, relative to the robot base frame, was a laser tracking system called "*Optotrac*" designed and built at the University of Surrey (Mayer & Parker, 1988). The instrument consists of two tracking sub-systems that each directs a laser beam towards a retro-reflective target, attached to the end-effector of the robot arm, by using two orthogonally mounted optical scanner units. The resultant tracking errors from driving the scanners are used to calculate the 3-dimensional position of the target. The instrument has a repeatability of $0.1 mm (Stanton & Parker, 1992). Simultaneously with the target position measurements, the angular positions of the first three links were measured by extracting the relevant encoder signals from the robot controller, using hardware developed at Glasgow University (Abderrahim & Whittaker, 1996) fig. 3. Hand-shaking signals were established to synchronise the measurements taken by the two systems. The *optotrac* set up takes very long time, which makes using it for industrial applications much less effective than existing industrial solutions nowadays, such as the *Krypton K600* and *K400* (Renders, 2006) and the *Leica Lasertracker LTD 500* (Fixel, 2006). It is worth mentioning that, all these systems would have served well our measurement requirements.

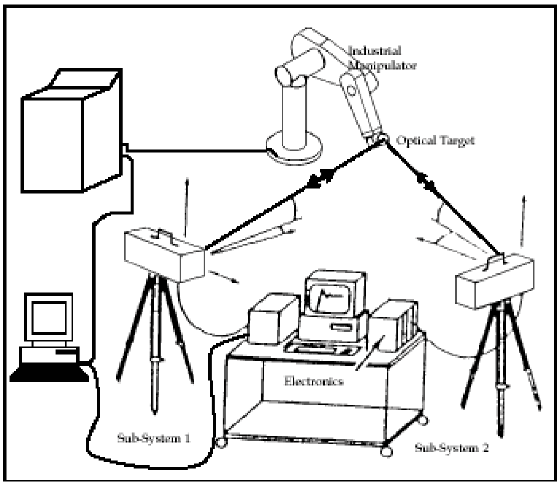


Fig. 3. The measurement system including the Optotrac & motor angles and currents acquisition hardware.

The second measurement system to describe here is the Krypton *K600* solution. The main piece of the *K series* measurement system is the camera system, consisting in three linear CCD cameras, shown in fig 4. The camera system relies on infra red light active LEDs, and therefore they cannot be seen by the human eye. When a LED is picked by the three linear cameras the computer calculates its exact position in the 3D space.

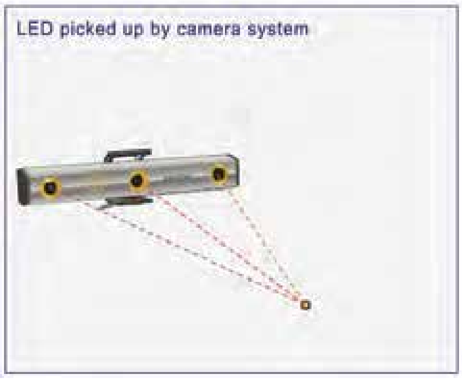


Fig. 4. The Krypton K600 Camera system (*Courtesy of Metris*).

The calculation is achieved by comparing the image of the 3 linear CCD cameras, from the effect of having 3 planes intersecting on the LED position, which is then calculated relative the pre-calibrated camera as illustrated in fig 5. According to the manufacturer the system is capable of tracking up to 256 LEDs simultaneously, through computer controlled strobing. This simultaneous multiple point tracking allows the measurement of the position and orientation of objects by attaching to them 3 or more LEDs and measuring their positions simultaneously. The system has a single point accuracy starting at 60 μ and is capable of measuring targets at 6 m distance from the camera.

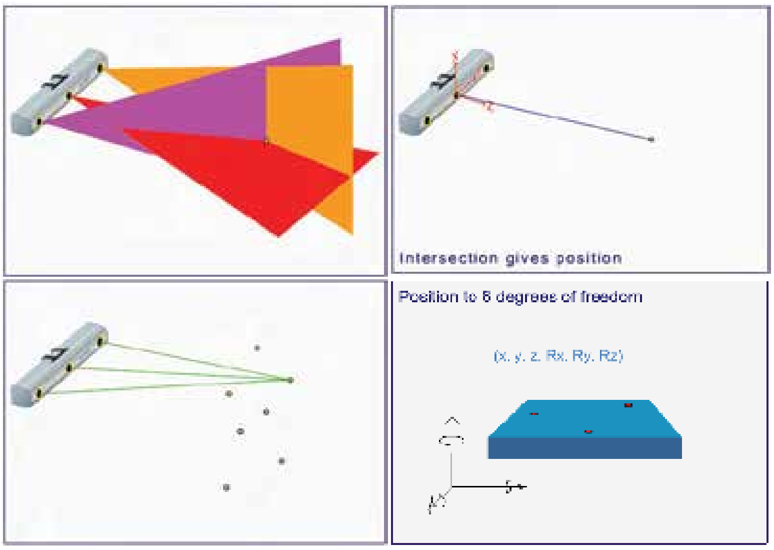


Fig. 5. Illustration of the measurement steps of the K600 (*Courtesy of Metris*).

This measurement system is a fundamental piece of metrology, which can be used for robot calibration and other application like motion analysis and 3D CMM inspection. This system is then valid for any calibration procedure that relies on measuring the pose of the end-effector of the manipulator. All is needed is fixing the LEDs precisely on the end-effector and start the measurement, which takes only minutes. Until recently the measurement step was often a lengthy and the main hurdle for achieving effective calibration.

4.3. Parameters Identification

Among the variety of methods of parameters identification, this section is limited to describing a method where the authors have done research and experimental tests (Abderrahim & Whittaker, 2000). In the current work, only robots with revolute joints have been considered. Prismatic joints have only one feature, the line of translation. Stone (Stone, 1987) gives a simple approach for the estimation of this feature. Concerning the required measurements, Stone (Stone, 1987) determines the model parameters through direct measurement of the joints' motions by attaching to the moving link a target whose position is measured. Stone allows free positioning of the target on the moving link while Stanton (Stanton & Parker, 1992), due to the nature of the measurement system, specifies a target fixed to the end-effector. The measurement of the target Cartesian position in the current work uses the same system used by Stanton.

The joint features are identified link by link. In a similar manner to Stone, we identify first the *plane of rotation*: the plane containing a circle described by a point on a rotating link, followed by the *centre of rotation*: the centre of that circle which will be situated on the joint axis. We also explicitly identify the *radius of rotation*. We next introduce a novel translation of the *plane of rotation* along the *axis of rotation* so that it coincides with the D-H X-Y plane. This translation, along with the three identified features, allows the D-H model parameters to be extracted.

4.3.1. Features Identification

Identifying the *plane-of-rotation* is a straightforward task. When joint *i* is made to rotate and the rest of the joints (1 to *i-1* and *i+1* to *n*) are locked, the target fixed relative to the end-effector then traces a circle in space. The coefficients of the plane of rotation can be estimated by fitting to a plane the *m* measured Cartesian positions of the target corresponding to *m* deferent positions of joint *i*. Several methods have been suggested to solve this problem, as stated in (Stone, 1987) and (Stanton & Parker, 1992). For reasons of simplicity, Stone's linear least square technique is used in our work. The method attempts to minimise the sum of the perpendicular distances between the measured points and the estimated plane of rotation. Though not essential, it aids visualisation, when measuring the target motion due to movement of joint *i*, all joints 1 to *i-1* and *i+1* to *n* are locked in their zero positions. Independently, each joint of the manipulator is made to move through the required *m* positions, maintaining a correct sense of rotation. It is assumed that the joint angles $q_{i,j}$ are ordered such that $q_{i,j} < q_{i,j+1}$ where *i* is the joint number and *j* is the order of the corresponding measured end-effector position $\vec{p}_j = \left[x_j y_j z_j\right]^T$.

The equation of a plane can be written as:

$$Ax + By + Cz + D = 0 \tag{4}$$

where *x*, *y* and *z* are the coordinates of the points of the plane and the coefficients A, B, C and D are to be identified. This equation can be rewritten as:

$$z = Ex + Fy + G = \phi^T \Theta \tag{5}$$

where the $\phi = [x, y, 1]^T$ and $\Theta = [E, F, G]^T$, which are defined as the information and parameters vectors. The *z* coordinate is defined to be the output of Eq. (4). A simple regression of *z* on *x*, *y* and 1 corresponds to the minimisation of the sum of the squared errors in the *z* coordinate.

$$\Xi = \sum_{j=1}^{m} (z - z_j)^2 = \sum_{j=1}^{m} (\phi^T \Theta - z_j)^2 \tag{6}$$

The minimisation of this expression by equating it to zero yields the linear least squares solution (Hsia, 1977), where it is assumed that the coordinates of the information vector are independent variables and measured without error. Three points of the measured target positions are used for the formation of the new coordinate frame. These three points uniquely form an initial approximation of x-y plane, hence the plane of rotation. The points are chosen to be mutually most distant from one another and are denoted by $\vec{p}_k$, $\vec{p}_l$ and with $k < l < m$ in order to preserve the sense of rotation. Fig. 6 assists in the understanding of the formation of the new coordinate frame where the plane of rotation is estimated.

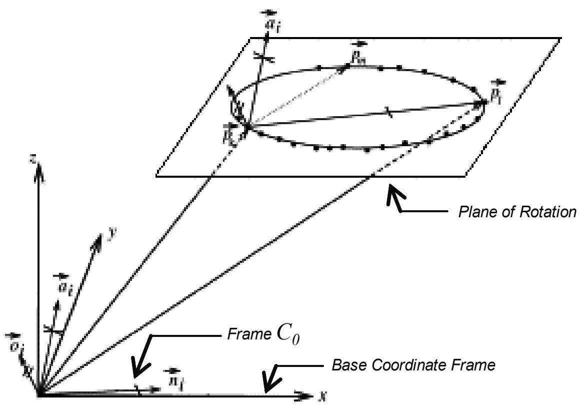


Fig. 6. The formation of new coordinate frame from the measured points.

The x-axis is chosen to be parallel to the line joining $\vec{p}_k$ to $\vec{p}_l$. The z-axis is perpendicular to this axis and to the line joining $\vec{p}_k$ to $\vec{p}_m$. The y-axis completes the orthogonal system. The origin of this initial coordinate frame is coincident with the origin of the frame in which the data are measured (presented), which in our case is the robot arm base coordinate frame. The homogeneous transformation matrix between the measurement frame and the new frame C_0 is given by:

$$R_i = \begin{bmatrix} \vec{n} & \vec{o} & \vec{a} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{7}$$

where $\vec{n}$, $\vec{o}$ and $\vec{a}$ are he unit vectors of the newly formed coordinate frame, which forms a pure rotation T. The measured data can be transferred to the newly formed frame C_0 where a least square regression is executed to calculate the coefficients of the plane of rotation, which in turn are transferred back the measurement coordinate frame. The detailed procedure and justification are treated in (Abderrahim & Whittaker, 2000). In the procedure used in the current work, unless the target position is already on the nominal D-H X-Y plane, the plane of rotation for each link should now be translated to coincide with the D-H X-Y plane. This is first done by using the offsets between the nth axis and the target attached to the nth link. Consequently, accurate knowledge of the target location with reference to the last link is essential. In the case of other links, the previously estimated parameters are used. If for some reason this information is not available, nominal lengths could be used. The translation is along the z-axis of the C_i for each link.

By a similar least squares procedure, the coordinates of the *centre of rotation* $\vec{p}_i$ can be identified. The *radius of rotation* is then easily established.

The next step is the coordinate frames construction and the D-H parameters identification. This part makes use of the set of the estimated n normal vector to the *planes of rotations* $\vec{a}_i$ and n centre of rotations $\vec{p}_i$ to specify the locations and orientations of the D-H model link coordinate frames. First, partial D-H models are specified to define the location and orientation of the individual D-H coordinate frames in terms of the base frame of the robot arm. These are given by:

$$T_i = A_1 \cdot A_2 \cdot \ldots \cdot A_i \tag{8}$$

From above we deduce:

$$\begin{cases} T_i = T_{i-1} \cdot A_i \\ \text{and} \\ A_i = T_{i-1}^{-1} \cdot T_i \end{cases} \tag{9}$$

where the expression of the individual T_i are given by:

$$T_i = \begin{bmatrix} \vec{n}_i & \vec{o}_i & \vec{a}_i & \vec{p}_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{10}$$

Using the result of expression (9) and (10) and equating them to the expression of A_i given in (3) the parameters forming the matrices can be calculated in a backward iteration from the last joint to the first. The errors affecting the kinematic model parameters can be extracted. This method was tested experimentally and results proved its effectiveness (Abderrahim & Whittaker, 2000).

4.4. Compensation Implementation

Compensation is the last important step in manipulator calibration, and consists of replacing the nominal kinematic model parameters by the new calculated "true" values in the robot's position control software. However this operation is neither easy nor always possible, and usually implemented by the robot manufacturer such as the case of ABB. The other option is the compensation off-line of all points of the programs using the established correct kinematic model. The 3D Cartesian positions are used with the inverse correct kinematic model to calculate the joint values for the real robot. These values are then used in the nominal forward kinematic model to calculate new 3D compensated end-effector positions. These compensated positions are then loaded in the robot controller, with the nominal kinematic model, and when executed achieve the real desired positions. In this process, the robot is commanded to an offset pose from the intended, and because of its model errors it reaches the correct pose. This solution is the easiest to implement, but attention should be paid to avoid *relative* 3D command positions, which include on-line calculations that involve the nominal kinematic model. This option is implemented in the calibration software, *ROCAL*, provided by Metris (Renders, 2006).

5. Robot Simulation Software and Off-Line Programming

With the rapid development of information and computer technology, recent years have seen an increase of computer simulation environments of complete manufacturing cells and Robotics Computer Simulation modules that are also offered by some robot manufacturers. Robot Simulation modules are provided for Off-Line Programming (OLP) systems which incorporate original path planning and interpolation algorithms of the robot controller (Beyer and Wulfsberg, 2004). Tremendous time saving can be achieved and costly programming mistakes can be spared if the tasks can be planed and simulated before implementation (Elatta et al., 2004). These simulation tools can also be used to quickly investigate robot applications using accurate 3D computer models and path planning algorithms. Off-Line prepared programs can then be loaded directly in the robot controller, but due to differences between the real robot and nominal model in the simulation software, position points have to be corrected. This process can be

reduced or completed eliminated if the robot is calibrated and the "true" compensated model is transferred to the simulation environment. In this case the whole production line can be simulated and robot programs can be simply downloaded from the simulation environment to the actual robot on the shop floor with no "touch-up". This also means that a failing robot can be replaced by another "absolute accuracy" robot with minimum disruption to the production line (Hefele & Brenner, 2001).
As samples of these computer simulations software, a brief description of the three most known to the authors is given next:

- **IGRIP (Interactive Graphics Robot Instruction Program):** developed by the company Delmia, is a Robotic Simulation and Off-Line Programming Solution. According to the company, it permits the design and simulation and analysis of the complete manufacturing cell. IGRIP includes the most comprehensive library of robot models available, with more than 500 industrial robot models, including the latest from FANUC, ABB, Motoman and Nachi. Application-specific devices such as weld guns, work piece positioners, rails, gantries, and workpiece clamps are available in options such as Arc, Spot, and Paint (Delmia, 2003).
- **RobotStudio:** is a simulation software developed by ABB and reflects the group's great effort to produce such a mature product. RobotStudio is built on the ABB VirtualRobot, which is an exact copy of the real software that runs the robots in production, and can generate exact RAPID programs. It allows very realistic simulations to be performed, using real robot programs and configuration files identical to those used on the shop floor. It is therefore easy to transfer robot programs and configuration parameters between a robot and the computer. According to ABB, virtual robot technology concept makes the company the only supplier offering true off-line programming, with no translation or touch-up. In addition, the company's calibration software is built on the RobotStudio platform, and that is how it uses a virtual model identical to the real calibrated robots (see www.abb.com/ for more details). RobotStudio provides the tools to increase the robot systems profitability by allowing the users perform tasks such as training and optimization without disturbing production.
- **Festo COSIMIR Professional:** is a PC-based software package for 3D modelling (including fault finding and design optimization), programming and performance emulation of various industrial robot systems. Modelling with COSIMIR Professional is made simple by virtue of the libraries of robots, end effectors and automation components. The program also imports several CAD formats. The programming feature supported by COSIMIR Professional, permits off-line programming of most commonly used Robots, allowing syntax check and automatic integration of position lists, automatic creation of complex parameter assignment, and the up load and download to and from Kuka, ABB, and all Mitsubishi controllers. The simulation mode permits the testing of cell and tool designs. All motion sequences and handling processes can be instantly simulated in the modelled work cell to avoid collisions and optimise cycle times (see www.festo.com/ for more details). Compilers of more robot models can be integrated in the software at any time to produce realistic simulations and analysis of the designed process (Karras, 2003). According to recent consultation with *Festo GmbH,* the software has the capability of connecting the real robot controller to the simulation PC to produce precise cycle time to permit a *real* optimisation and analysis.

6. Existing Absolute Accuracy and Industrial Calibration examples

In both examples presented in this section, to the best knowledge of the authors, no mathematical theory behind the choice of the measurement point or identification algorithms has been published. Therefore the intention is to bring to the reader's intention some of the sample calibration tools available in the market and describe how they work.
The absolute calibration of ABB robots (**AbsAcc**) is implemented as a model based solution, highly integrated in the standard controller software. The solution is developed in-house by ABB and uses the standard kinematic description as a base. AbsAcc covers two parts, geometric compensation and deflection compensation. There is a set of parameters (approx 40) describing the individual properties for a calibrated robot. Once the parameters are loaded and activated, the operator can use the robot as normal. All compensations are done automatically without any further adjustments or special handling. The positioning accuracy of a calibrated robot depends on the robot size and variant and is in average between 0,25 mm and 0,55 mm for a robot handling between 5 and 500 kg.
The calibration is performed for each individual manipulator that requires AbsAcc and normally this is done in the factory. The calibration software ***CalibWare*** and the procedure require the robot to be measured and for this purpose a 3D measuring station is needed. ABB uses the Leica Lasertracker (LTD 500) in order to have a measuring accuracy that supports the calibration and verification. The process compensates for geometric and no geometric errors as shown in fig. 7.

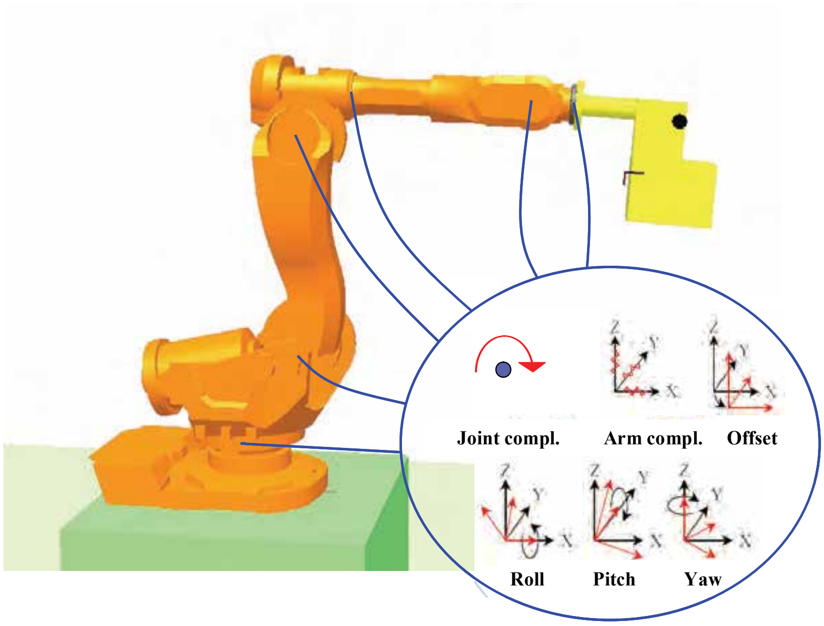


Fig. 7. Geometry deviations and joint deflection due to load *(Courtesy of ABB).*

The calibration itself contains 100 positions randomly distributed within the robot working area and giving excitation to all robot axes. From the result a set of parameters that best minimizes the error between the model and real robot are calculated. The robot positioning accuracy is then verified by running 50 more positions and calculating the difference between the ordered position and the measured, according to the set up shown in fig. 8.

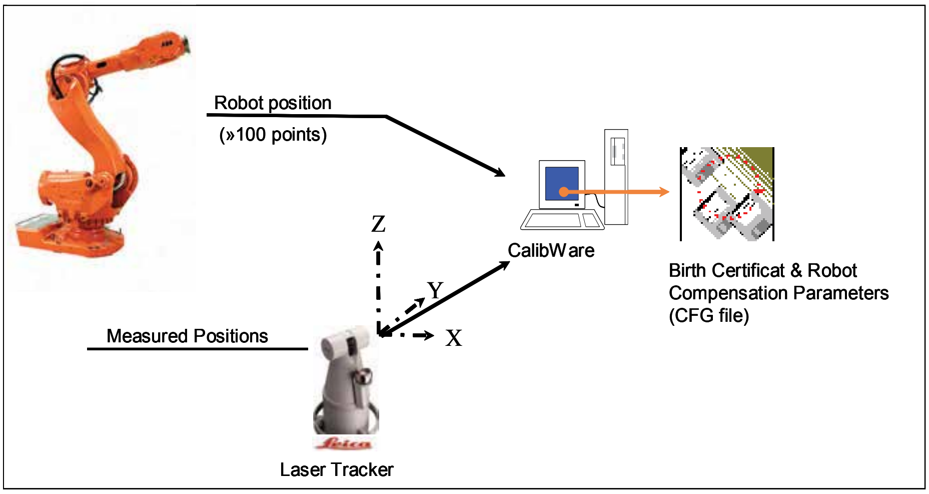


Fig. 8. ABB calibration process set up. *(Courtesy of ABB).*

On the other hand, the calibration procedure provided by **Metris** employs the Krypton K600 camera to measure between 25 to 100 calibration poses. These poses are automatically generated in the field of view of the camera and in the working volume of the robot. The ROCAL software will thus create a set of poses evenly spread. This set of poses is then converted into a robot program, which is loaded to the robot and executed. In each of the poses, at least 3 infrared LED's attached to the robot tool frame are measured fig. 9. Measuring three LED's, gives position and orientation of the robot tool frame, which is called "6D". With most other systems, only 3D or 1D position is acquired. Acquiring position and orientation during the measurement reduces the number of robot positions that are required to identify a higher number of robot parameters.
The set of estimated robot positions and the measured poses are used to calculate a new robot model. This robot model contains: the real link lengths and angles, deflection of links, joints deflection, base deflection, coupling errors and encoder offsets.

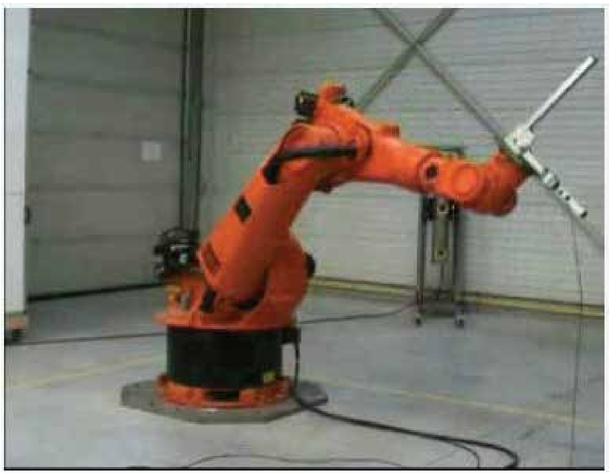


Fig. 9. Minimum of 3 LEDs are attached the tool frame during the calibration to obtain complete pose (6D). (*Courtesy of Metris*).

Since Metris is not a robot manufacturer, their system of measurement and calibration targets all robots makes and models. According to (Renders, 2006), it takes only about 1 hour of work to include a new robot model. All is needed is a description of all joint axes with respect to the base frame of the robot. As for the compensation of the errors, this is done on the target positions of the program (off line). The compensation on the controller level is more difficult and requires more effort. If this is required, another software module (***RETRAC***) and the new robot model are provided by Metris to be integrated in the controller path planner. Therefore any robot that connects to that controller can run the Absolute Accuracy version.
The combinations of ROCAL software and the *K series* measurements system are able to calibrate the entire cell and hence provide the link between simulations and the real world. The software provides three types of calibration routines, robot calibration, tool calibration and environment calibration. In the tool calibration LEDs are fixed to the tool and measurements are performed to establish the tool TCP frame, as illustrated in fig 10. In the case of the environment calibration, the CMM capabilities of the measurement system are used to register reference points of the environment relative to the robot and establish a fixture frame.

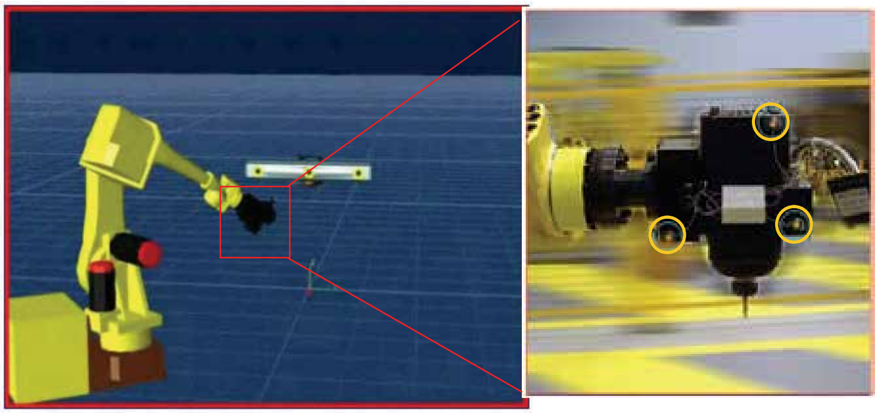


Fig. 10. During tool calibration LEDs are fixed to the tool (*Courtesy of Metris*).

For illustration a real example of the achieved accuracy after calibration of a 159 Kg payload industrial robot manipulator holding full load is presented next. The original pose accuracy before calibration was 3.25 mm and 5.43 mrad. After complete calibration these figures are brought down to 0.29 mm and 0.35 mrad.
If a subset of parameters is calculated in partial calibration the achieved accuracy is a little worse than the mentioned above, but still improve the Cartesian positioning of the robot a great deal. For the same manipulator mentioned above, encoder offset calibration achieves a complete pose accuracy of 1.44 mm and 3.87 mrad, while geometrical only calibration guarantees an accuracy of about 0.72 mm and 2.12 mrad.
The Rocal software and the *K600* system make a powerful tool, which makes it possible to move a 150kg robot load under 0.5 mm average error in its entire working volume.

7. Conclusion

This chapter discussed the accuracy issues of robot manipulators and the importance of searching for absolute accuracy for industrial manipulators. It shows how there is an increased need to use off-line programming and robotic and manufacturing simulation

software to analyse and optimise the processes and trajectories of manipulators present in the process. Throughout the chapter it has been emphasised that without absolute accuracy there cannot be off-line programming of robots and without calibration there cannot be absolute accuracy. Therefore the combination of OLP and absolute accuracy, production line design and development in manufacturing can be done in record times through realistic simulations. Programs can be downloaded directly to the robots without touch-up and corrections, which can be interpreted in reduction of downtime, great efficiency, easy and rapid duplication of production lines or even complete cells.
The text presents therefore an overview of the work and methods of accuracy improvement and explains the main steps to be followed in this process. Existing commercial solutions have been described to give the reader an idea of the state of the art of the technology and where to find it. Obviously these are not the only commercial solutions but they have been chosen because, to the best knowledge of the authors, they are among the best solutions and they provided all information to make our evaluation.

8. Acknowledgements

The authors are grateful for the feedback and the constructive discussion with Peter Fixel from ABB and Steven Renders from Metris especially for providing documentation and permission to use the graphics. Thanks are extended to Fred Zierau from Festo USA for his assistance.

9. References:

Abderrahim M, Whittaker AR. (1996). Design and construction of a low cost instrumentation system for an industrial robot. *Proc of the Int Conf MECHATRONICS and M2VIP*, pp. 369-74, Guimaraes, Portugal, September 1996.

Abderrahim, M. & Whittaker, A. R. (2000) Kinematic model identification of industrial manipulators, Robotics and Computer-Integrated Manufacturing, Vol. 16, N° 1, February 2000, pp 1-8

Beyer, L. & Wulfsberg, J. (2004). Practical robot calibration with ROSY, Robotica, Vol. 22, pp.505-12, Sep 2004, Cambridge University Press (UK), ISSN: 0263-5747

Chen J, Chao L-M. (1987). Positioning error analysis for robot manipulators with all rotary joints, IEEE J of Robotics and Automat , Vol. 3, N° 6, 1987, pp. 539-45.

DELMIA Corporation, (2003). IGRIP Documentation. Auburn Hills, Mich.: DELMIA Corporation, 2003. www.delmia.com

Driels MR, Pathre US. (1994). Robot calibration using an automated theodolite. *Int J Adv Manuf Technol*, Vol. 9, N° 2, pp. 114-25. 1994.

Driels MR, Swaze WE.(1994). Automated partial pose measurement system for manipulator calibration experiments. *IEEE Trans Robotics Automat*, Vol. 10, N° 4, 1994, pp. 430-440.

Driels MR. (1993). Using passive end-point motion constraints to calibrate robot manipulators. *J Dyn Systems Meas Control, Trans ASME* , Vol. 115, N°. 3, 1993, pp. 560-6.

Elatta, A.Y. ; Gen, L.P. ; Zhi, F.L. ; Daoyuan Y. & Fei, L. (2004). An Overview of Robot Calibration, *Information Technology Journal*, Vol. 3, N° 1, 2004, pp. 74-78, ISSN 1682-6027

Fixel, P. (2006). Absolute Accuracy Marketing Presentation, *ABB Automation Technologies AB*, peter.fixell@se.abb.com, June 2006.

Harb SM, Burdekin M. (1994). A systematic approach to identify the error motion of an n-degree of freedom manipulator. *Int J Adv Manuf Technol*, Vol. N° 2, 1994, pp. 126-33, Hayati S, Mirmirani M. (1985). Improving absolute accuracy of robot manipulators. *J Robotic Systems*, Vol. 2, N° 4, 1985, pp. 394-413.

Hefele, J. & Brenner, C. (2001). Robot pose correction using photogrammetric tracking. *Proceedings of SPIE -: Machine Vision and Three-Dimensional Imaging Systems for Inspection and Metrology*, Vol. 4189, Feb. 2001, pp. 170-178.

Hsia TC. (1977). *System identification: least squares method*. Lexington, MA: Lexington Books.

Karras, U. (2003). *COSIMIR Profesional Manual*, Festo Didactic, GmbH. Germany

Khalil W, Gautier M, Enguehard C. (1991). Identifiable parameters and optimum configurations for robots calibration. *Robotica*, Vol. 9, 1991, pp. 63-70.

Mayer R, Parker GA. (1988). Calibration and assessment of a laser based instrument for robot dynamic measurement. *In IMEKO XI, 11th Triennial World Congress of the International Measurement Confederation*, Paper no. 172, Houston, Texas, October 1988.

Metris (2005). Krypton interactive CD, http://www.metris.com/

Mooring B, Roth Z, Driels M. (1991). *Fundamentals of manipulator calibration*, Wiley, New York.

Mooring BM, Pack TJ. (1987). Aspects of robot repeatability, *Robotica*, Vol. 5, 1987, pp. 223-30.

Pathre US, Driels MR. (1990). Simulation experiments in parameter identification for robot calibration. *Int J Adv Manuf Technol*, Vol. 5, 1990, pp. 13-33.

Renders, S. (2006). Private Presentation about the Krypton System. steven.renders@metris.com

Roth ZS, Mooring BW, Ravani B. (1987). An overview of robot calibration. *IEEE J Robotics Automat*, Vol. 3, N° 5, 1987, pp. 377-85.

Stanton D, Parker GA. Experimental kinematic calibration using a modified S-model. Proc. of ICARCV'92: The Second Int Conf on Automation, Robotics and Computer Vision, Vol. 3, September 1992.

Stone HW. (1987). *Kinematic modeling, identification, and control of robotic manipulators*. Kluwer Academic Publishers.

Whitney DE, Lozinski CA, Rourke JM. Industrial robot forward calibration method and results. Trans ASME: J Dyn Systems, Meas Control, Vol. 108, 1986; pp. 1-8.

Calibration of Serial Manipulators: Theory and Applications

Irene Fassi[1], Giovanni Legnani[2], Diego Tosi[2], Alberto Omodei[2]
[1]*ITIA-CNR Institute of Industrial Technology and Automation, National Research Council V.le Lombardia 20/A 20131 Milano - Italy*
[2]*Dip. Ing. Meccanica, Università di Brescia Via Branze 38, 25123 Brescia, Italy*
giovanni.legnani@ing.unibs.it

1. Introduction

The kinematic calibration is a procedure to improve the manipulator accuracy without mechanical means by acting on the manipulator controller.
Although manipulators are composed by accurate mechanical components, the precision of their motion is affected by many sources of error (Mooring et al, 1991). The final position accuracy is mainly influenced by: kinematic inaccuracy (due to manufacturing and assembly errors in both actuated and passive joints), load deformation (due to external forces including gravity) and thermal deformation (Reinhar et al, 2004). This is true for serial (Mooring et al, 1991) as for parallel (Wildenberg, 2000) manipulators as well. Each of these factors should be addressed with an appropriate compensation or calibration methodology. This work deals with kinematic inaccuracy, related to robot geometry, assembly and joint motion.
One possibility to compensate for geometrical errors is to perform a *kinematic calibration*. The robot is requested to reach some *desired* poses and the reached *actual* poses are measured. Then, the exact robot geometry is estimated analyzing the difference between the desired and the reached poses. This procedure requires a *parametric* identification of the manipulator which consists in the formulation of a geometrical model of the robot in which each source of error is represented by a parameter. The parameter set includes link lengths, joint axes inclination and joint coordinate offsets. The calibration consists in identifying the actual values of all these parameters. Once this operation is performed, it is possible to predict the pose error for any robot configuration and so it is possible to compensate for them by suitable joint motions.
The aim of this work is to address all the steps of the procedure which includes:

1. The development of a suitable kinematic model of a general serial manipulator;
2. One example of collection of experimental data;
3. The estimation of the numerical value of the manipulator parameters;
4. The error compensation.

For each phase different alternatives are described and critically compared.
A relevant part of the chapter summarises, compares and extends the existing techniques used to generate a suitable parametric kinematic model for a general serial manipulator. Rules for automatic generation of models with different characteristics are developed. These results will be used in a next chapter for the development of a parametric kinematic model for a general parallel manipulator (PKM).

An effective model for robot calibration must describe all the possible sources of error (it should be *Complete*) and, to avoid singularities, little geometrical errors must be described by small changes in the values of the corresponding parameters (*Parametrically Continuous* or *Proportional*). Such a model is often referred as CPC (Zhuang & Roth, 1992, Mooring et al, 1991). Furthermore, if each source of error can be described by one parameter only (absence of redundant parameters), the model is defined *Minimum* and this allows to obtain an unique numerical solution from the calibration process. In this paper such models are called MCPC (*Minimum Complete and Parametrically Continuous*).
For a given robot more than one valid MCPC model can be defined; generally speaking they are equivalent to each other, but each of them has different characteristics. Some comparisons about the different models are contained in the following Sections. Discussion about the accuracy achievable with the different models in presence of noise or measuring errors is outside the scope of this paper.
Being the aim of this work the analysis of geometrical errors, it will be assumed that the robot is composed by rigid links connected by 'ideal' joints (without backlash). All the possible sources of error will be considered constant. It will be also assumed that actuators are directly connected to the manipulator joints and so errors in the kinematics of the transmissions will be here neglected. These hypotheses are reasonable for many industrial manipulators.
In Sections 2 and 3 the basic concepts for the calibration procedure are presented. In Section 4 the general formula for the determination of the parameters number is discussed. Some different approaches used for serial robots are reordered (Sec.s 5 and 6), compared (Sec. 7) and a modified one is proposed (Sec. 8). After the explanation of an elimination procedure for the redundant parameters (Sec. 9) the calibration procedure for two different robots is discussed (Sec. 10 and 11). Eventually, Sec. 12 draws the conclusions.

J_i : i -th joint	$T(u,a)$: translation of a in $\vec{u}$ axis direction	$//$: parallel
R : revolute joint	$R(u,\varphi)$: rotation φ around axis $\vec{u}$	$\not\parallel$: not parallel
P : prismatic joint	$R(x,y,z,\alpha,\beta,\gamma)$: 3D rotation	$\perp$: orthogonal
R: number of revolute joints	Δa , Δb , Δc displacements along x, y, z	
P: number of prismatic joints	$\Delta\alpha$, $\Delta\beta$, $\Delta\gamma$ rotations around x, y, z	

Table 1. Symbols and abbreviations.

2. Methodological Bases

When choosing a parameter set to describe errors in a manipulator geometry, many different approaches can be followed; two of the most common are:

- Extended Denavit and Hartenberg approach ('ED&H');
- Incremental approach ('Inc').

When an 'ED&H' approach is adopted, a specific set of parameters is chosen to describe the robot structure (Denavit & Hartenberg, 1955) and errors are represented by variations of these parameters. The direct kinematics is represented as

$$S = F(Q, \Lambda_n + \Delta\Lambda) \tag{1}$$

where $S = [x, y, z, \alpha, \beta, \gamma]^t$ represents the gripper pose, $Q = [q_1, \ldots q_n]^t$ is the vector of the joint coordinates, $\Lambda_n = [\lambda_1, \lambda_2, \ldots, \lambda_N]^t$ is the vector of the nominal structural parameters and $\Delta\Lambda$ is

the vector of their errors.
In many cases Λ consists in the set of the Denavit & Hartenberg parameters. However, for calibration purposes, the classic D&H approach must be extended to assure the generation of a MCPC model in any situation (Sec. 5).
Conversely, if an 'Inc' approach is adopted, the nominal geometry of the robot is described by any parametric formulation $\overline{\Lambda}$ without requirements of minimality, completeness and proportionality. Errors are then represented by a suitable number of other parameters describing the difference between the nominal manipulator geometry and the actual one. This second set of parameters $\Delta\Lambda$ must be defined taking into account minimality, proportionality and singularity issues. There is no need that the number of the parameters in $\overline{\Lambda}$ equals that of those in $\Delta\Lambda$. In this case the direct kinematics is represented as

$$S = F(Q, \overline{\Lambda}, \Delta\Lambda) \tag{2}$$

3. Identifying the Parameter Values

Calibration can be defined as the procedure to estimate the numerical value of $\Delta\Lambda$ which better describes the kinematics of a given robot. It can be done using two different approaches: *Pose Measuring* and *Pose Matching* (Cleary, 1997).
Using the pose measuring approach, the robot is requested to reach a predefined '*desidered*' pose S_d and the calibration process is performed elaborating the difference between S_d and the '*real*' pose S_r reached by the gripper.
In the *pose matching* approach, the robot gripper is driven to a number of know poses and the corresponding joint rotations are measured. The difference between the expected joint coordinates and the actual ones is used as input for the identification procedure.

3.1 'Pose Measuring'

If we ask the robot gripper to move to a certain desired pose S_d, the gripper will reach the actual pose S_a:

$$\mathbf{S}_a = F(\mathbf{Q}_d, \Lambda_n + \Delta\Lambda) \quad \text{with} \quad \mathbf{Q}_d = G(\mathbf{S}_d, \Lambda_n) \tag{3}$$

where Q_d are the joint coordinates evaluated using the inverse kinematics model $G(\ldots)$, based on the nominal robot parameters Λ_n. The error in the gripper pose will be:

$$\Delta\mathbf{S} = \mathbf{S}_a - \mathbf{S}_d \tag{4}$$

Assuming that the magnitude of the parameter errors is sufficiently small, the equations can be linearized as:

$$\Delta\mathbf{S} \cong \mathbf{J}_\Lambda \cdot \Delta\Lambda \qquad \mathbf{J}_\Lambda = \frac{\partial\mathbf{F}}{\partial\Lambda} \tag{5}$$

where J_Λ is the jacobian matrix evaluated for $Q = Q_d$ and $\Lambda = \Lambda_n$.
If the value of ΔS can be measured for a sufficient number of poses, it is possible to estimate the value of $\Delta\Lambda$. The required equations are obtained rewriting Eq. (5) for each pose. A graphical representation of the procedure is presented in Figure 1.

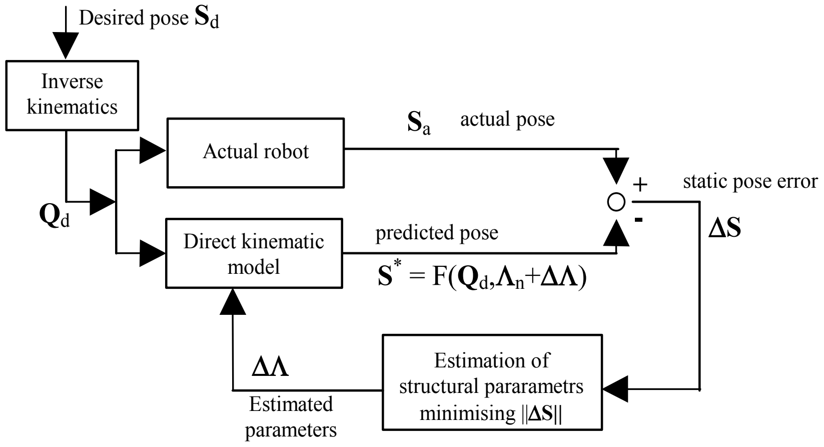

Desired pose $\mathbf{S}_d$
Inverse kinematics
Actual robot
$\mathbf{S}_a$ actual pose
+
static pose error
-
$\Delta\mathbf{S}$
$\mathbf{Q}_d$
Direct kinematic model
predicted pose
$\mathbf{S}^* = F(\mathbf{Q}_d, \Lambda_n + \Delta\Lambda)$
$\Delta\Lambda$
Estimated parameters
Estimation of structural pararametrs minimising $\|\Delta\mathbf{S}\|$

Fig. 1. Calibration of a robot using the Pose Measuring approach.

3.2 'Pose Matching'

When we force the robot gripper to reach a certain desired pose S_d, we predict the joint rotations Q_d using the inverse kinematics based on the nominal values Λ_n of the structural parameters:

$$\mathbf{Q}_d = G(\mathbf{S}_d, \Lambda_n) \tag{6}$$

However, the actual joints values Q_a are different from the predicted ones:

$$\mathbf{Q}_a = G(\mathbf{S}_d, \Lambda_n + \Delta\Lambda) \cong \mathbf{Q}_d + \frac{\partial G}{\partial \Lambda}\Delta\Lambda \tag{7}$$

or also:

$$\Delta \mathrm{Q} = \mathrm{Q}_a - \mathrm{Q}_d \cong \frac{\partial G}{\partial \Lambda}\Delta\Lambda \tag{8}$$

$\Delta\Lambda$ is the vector containing the geometrical parameter errors and $\frac{\partial G}{\partial \Lambda}$ is evaluated for $S = S_d$ and $\Lambda = \Lambda_n$. Eq. (8) for pose matching is the equivalent of Eq. (5) for pose measuring. Since $\frac{\partial G}{\partial \Lambda}$ is generally not available, an alternative formulation can be used.

Linearizing the direct kinematics equation we have:

$$\mathbf{S}_a \cong F(\mathbf{Q}_d, \Lambda_n) + \frac{\partial F}{\partial \Lambda}\Delta\Lambda + \frac{\partial F}{\partial \mathbf{Q}}(\mathbf{Q}_a - \mathbf{Q}_d)$$

since S_a has been forced to be equal to $F(Q_d, \Lambda_n)$, and remembering Eq. (8) we get:

$$\frac{\partial F}{\partial \Lambda}\Delta\Lambda + \frac{\partial F}{\partial \mathrm{Q}}\Delta \mathrm{Q} \cong 0 \quad \Rightarrow \quad \frac{\partial G}{\partial \Lambda} = \left(\frac{\partial F}{\partial \mathrm{Q}}\right)^{-1}\frac{\partial F}{\partial \Lambda} \tag{9}$$

The number of the geometrical parameters N is greater than the number of the joint parameters that can be measured for each gripper pose, but if the value of ΔQ can be measured for a sufficient number of poses, it is possible to estimate the value of $\Delta\Lambda$. The required equations are obtained rewriting Eq. (9) for all the poses. A graphical representation of the procedure is given in Figure 2.

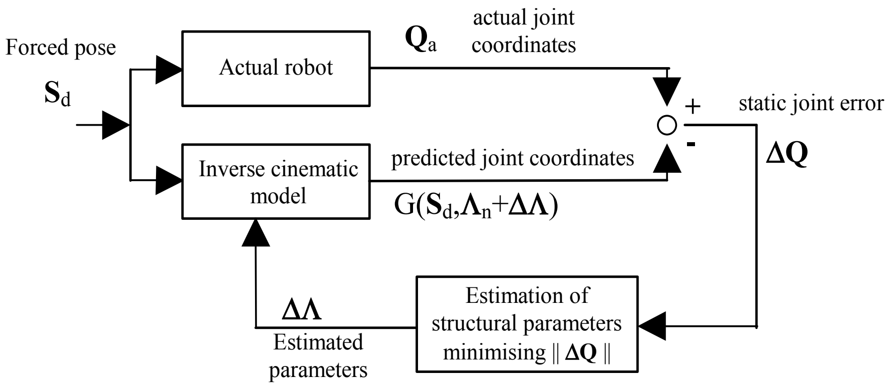


Fig. 2. Calibration of a robot using the Pose Matching Method.

3.3 Equivalence between Pose Matching and Pose Measured

An equivalence between the '*pose matching*' and the '*pose measuring*' can be established remembering that for a given value of the structural parameters we have:

$$\Delta S \cong \frac{\partial F}{\partial Q} \Delta Q$$

As already stated in Section 3.1, the pose measuring approach for calibration is based on the difference between the predicted and the measured poses. The same procedure can be used any time that a set of corresponding pair of joint rotation Q and of gripper poses S are known for the actual robot.

In Section 3.1 the joint coordinates were forced to known values and the corresponding gripper pose was measured. An alternative approach is to force a known gripper pose and to measure the corresponding joint rotations.

Moreover, calibration programs written for the pose measuring approach can be used also for the pose matching case using the following guidelines.

The robot gripper is forced to a known pose $\bar{S}$. The actual joint coordinates Q_a are measured. The theoretical gripper position S^* is predicted using the direct kinematics for $Q=Q_a$. Then, the robot calibration for pose measuring is performed assuming $\bar{S}$ as measured pose S_a and $Q_d = Q_a$. A graphical representation of the resulting algorithm for pose matching is presented in Figure 3.

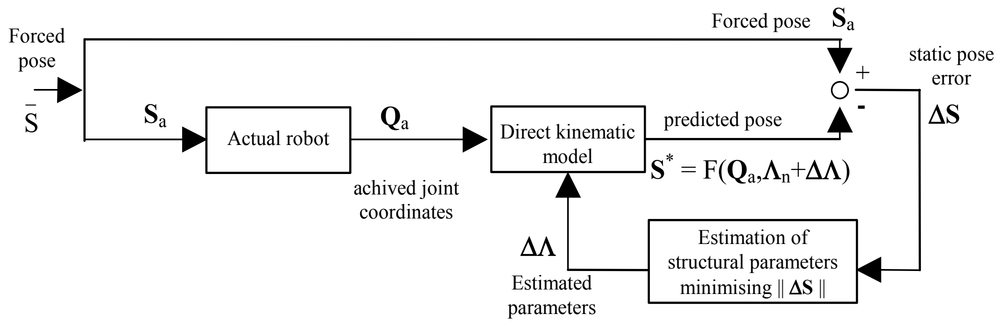


Fig. 3. Calibration of a robot using the Pose Matching Method (alternative approach).

3.4 Estimation of the Structural Parameters

Calibration Procedure: For the estimation of the structural parameters, three different identification procedures based on the guidelines presented in Sections 3.1 or 3.2 are here presented and compared:

- A non linear optimisation procedure;
- An iterative linearisation of the equations;
- An extended *Kalman* filter.

Their practical application will be presented in Section 11: a robot is driven to a set of known poses S_{ah} (h = *1, 2, ..., k*) and the corresponding joint rotation Q_{ah} are recorded. The mathematical procedures described in Sections 3.1 or 3.2 are then applied to estimate the structural parameters.

A non linear optimisation procedure ('amoeba'): The first method, experimented in this study to estimate the geometrical parameter errors, consists in writing Eq. (3) for a sufficient number of poses and in using a general purpose optimisation algorithm to find the value of $\Delta\Lambda$ which minimises the average error E_{op} based on the Euclidean norm:

$$E_{op} = \frac{1}{k}\sum_{h=1}^{k}\left\|\mathbf{S}_{ah} - F(\mathbf{Q}_{ah}, \Lambda_n + \Delta\Lambda)\right\|$$

where the subscript h is used to scan the k measured poses, S_{ah} is the *h-th* imposed gripper pose, Q_{ah} is the corresponding joint rotations, and *F(.)* is the predicted value for S. In the theoretical error free case, if the value of $\Delta\Lambda$ is exactly evaluated, E_{op} would be exactly null.

The algorithm, after several (many) iterations, gives an estimation of the value of $\Delta\Lambda$ which minimises E_{op}; E_{op} is also called residual.

A*moeba*, the optimization procedure we adopted, is adapted from (Flannery et al. 1992).

Iterative linearization of the equations ('linear'): The second method experimented consists in writing Eq. (5) for a sufficient number of poses and grouping all the equations in a linear system that can be solved to find $\Delta\Lambda$*:*

$$\mathbf{A}\cdot\Delta\Lambda = \mathbf{b} \quad \mathbf{A} = \begin{bmatrix}\mathbf{A}_1\\ \vdots \\ \mathbf{A}_h \\ \vdots \\ \mathbf{A}_k\end{bmatrix} \quad \mathbf{b} = \begin{bmatrix}\mathbf{b}_1\\ \vdots \\ \mathbf{b}_h \\ \vdots \\ \mathbf{b}_k\end{bmatrix} \tag{10}$$

Where A_h is J_Λ evaluated for the *h-th* pose and $b_h = \mathbf{S}_{ah} - F(\mathbf{Q}_{ah}, \Lambda_n)$.

The measures are generally redundant and so the system is solved with least squares criteria obtaining a first estimation for $\Delta\Lambda$ which is added to Λ_n obtaining a first prediction for the structural parameter values. This make possible a new better estimation of $b_h = \mathbf{S}_{ah} - F(\mathbf{Q}_{ah}, \Lambda_n)$ and a new solution of Eq. (10). The procedure can be iterated to improve the accuracy of the result. At each iteration j the last value of the parameters Λ_{j+1} replaces Λ_j:

$$\Lambda_{j+1} = \Lambda_j + \Delta\Lambda_j \quad \text{with} \quad \Lambda_0 = \Lambda_n$$

This procedure is repeated iteratively until the average error E_{it} reaches a stable minimum:

$$E_{it} = \frac{1}{k}\sqrt{\mathbf{b}^T\mathbf{b}}$$

E_{it} is also called residual.

An extended Kalman filter ('kalman'): A *Kalman* filter is a mathematical procedure to estimate the state of a system when uncertainties are present. Assuming that the vector of the geometrical parameters ΔΛ represents the state of a stationary system, an extended *Kalman* filter can be defined to give an estimation of ΔΛ starting from ΔS (Legnani and Trevelyan, 1996), (Clearly 1997).

The filter gives a new estimation of ΔΛ each time a new measure of ΔS is considered. The estimation $\Delta\Lambda_{h+1}$ of ΔΛ, after the *h-th* pose has been measured, is:

$$\begin{aligned} \Delta\Lambda_{h+1} &= \Delta\Lambda_h + \mathbf{M}_h\left(\mathbf{S}_{ah} - F(\mathbf{Q}_{ah}, \Lambda_n + \Delta\Lambda_h)\right) \\ \mathbf{M}_h &= \mathbf{P}_h\mathbf{C}_h^T\left(\mathbf{R} + \mathbf{C}_h\mathbf{P}_h\mathbf{C}_h^T\right)^{-1} \qquad \Delta\Lambda_0 = 0 \\ \mathbf{P}_{h+1} &= \left(\mathbf{1} - \mathbf{M}_h\mathbf{C}_h\right)\mathbf{P}_h \end{aligned}$$

Where C_h is the jacobian J_Λ evaluated in the *h-th* pose, M_h is the filter matrix gain after *h* steps, P_h is the matrix of the parameters covariance. R is the matrix of the measures covariance; extra diagonal elements in position *i, j* of matrices P (or R*)* indicates that the *i-th* and *j-th* parameters (or the *i-th* and the *j-th* measures) are correlated. P_0 representing the initial uncertainty of Λ should be initialised with suitable large values. The diagonal value of P contains a prediction of the accuracy of the estimation of ΔΛ, while R contains an estimation of the noise present in the measuring procedure. R and a series of S_{ah} and Q_{ah} must be given to the algorithm, which estimates ΔΛ and P.

After the processing of all the poses, an error index E_{ka} can be evaluated as

$$E_{ka} = \frac{1}{k}\sum_{h=1}^{k}\left\|\mathbf{S}_{ah} - F(\mathbf{Q}_{ah}, \Lambda_n + \Delta\Lambda)\right\|$$

4. Defining the Number of the Parameters

It has been proved for a n-DOF (degrees of freedom) serial manipulator (Mooring et al., 1991, Everett et al, 1987) that a model representing the pose of the gripper frame with respect to the fixed one, to be complete and minimum, must contain the following number of parameters:

$$N = 4R + 2P + 6 \tag{11}$$

being respectively R and P the number of revolute and of prismatic joints in the kinematic chain ($n = R + P$). This formula is derived under the hypothesis that

- Serial manipulators make use of *n* revolute or prismatic one-DOF joints (no spherical or cylindrical joints);
- All the joints are actuated (and so their motion is measured by the control system);
- The measures of all the 6 coordinates of the gripper are available for a number of different manipulator poses.

As stated in (Mooring et al, 1991), the proof of Eq. (11) is based on the observation that for revolute joints it is not relevant the position of the mechanical hinge but only the location of its geometrical axis which can be expressed in terms of two translations and two rotations (Fig. 4a). For prismatic joints only the axis direction is relevant and can be described with two rotations (Fig. 4b). At last, 6 parameters are necessary to define the gripper frame in an arbitrary location (Fig. 4c); this concept is directly applied in the 'Inc' approach (Sec. 6).
When only a partial measure of the gripper pose is available, the number of the identifiable parameters is reduced accordingly (Omodei et al, 2001):

$$N = 4R + 2P + G \tag{12}$$

being G the number of measurable coordinates of the gripper ($G \le 6$). In milling applications, for example, the tool pose is identified by 5 coordinates, being the rotation about the tool axis redundant with the spindle motion.

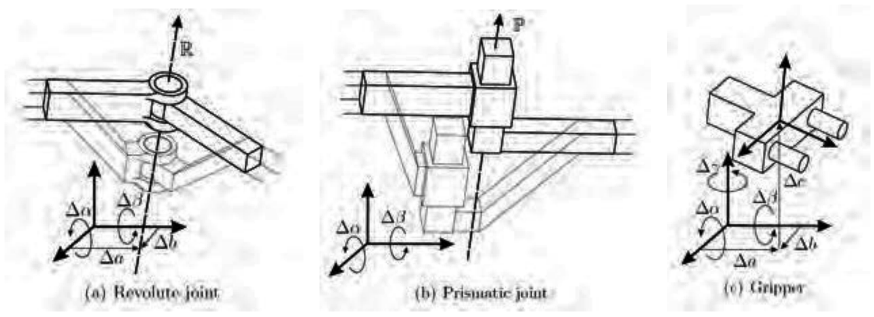


Fig. 4. The structural parameters necessary to define the location of: a) revolute joint axis (Δa, Δb, $\Delta\alpha$, $\Delta\beta$); b) prismatic joint axis ($\Delta\alpha$, $\Delta\beta$); c) gripper frame (Δa, Δb, Δc, $\Delta\alpha$, $\Delta\beta$, $\Delta\gamma$).

However, it is evident that 6 of the N parameters correspond to a wrong location of the robot base and they can be compensated simply by repositioning it. During experimental measures for the robot calibration, it is impossible to separate the effects of these errors with respect to errors in the location of the measuring instrumentation. Similarly the last 6 parameters describe the pose of the end effector with respect to the last link (i.e. the 'shape' and size of the gripper). So each time the end-effector is changed, these parameters change their value. We can conclude that only $N-12$ parameters are intrinsically related to the manipulator structure. We call them *internal parameters*; their number is

$$N_i = N - 12 = 4R + 2P - 6 \tag{13}$$

The internal parameters can be then compensated during the manipulator construction or with a proper calibration procedure performed in the factory. The other 12, called *external parameters* , can be calibrated and compensated only in the user's working cell.

5. Extended D&H Approach (ED&H)

This methodology is based on an extension of that proposed by Denavit and Hartenberg for the kinematic description of mechanisms (Denavit & Hartenberg, 1955) and widely used for serial manipulators (Paul, 1981).

As well known, the links are numbered from 0 (the base) to n (the gripper). A reference frame is embedded on each link (base and gripper included) in such a way that the i-th joint axis coincides with the axis z_{i-1}. The pose of the i-th link with respect to the previous one is expressed by a 4×4 transformation matrix.

$$A_i = \left[\begin{array}{ccc|c} & R_i & & T_i \\ \hline 0 & 0 & 0 & 1 \end{array}\right]$$

where R_i is a 3×3 rotation matrix and T_i is the vector of the origin coordinates of the i-th frame.

The direct kinematics of the manipulator can be expressed as

$$M = A_0 A_1 A_2 \ldots A_i \ldots A_n \tag{14}$$

where M is the matrix describing the gripper pose with respect to the base frame, n is the number of DOF. A_0 is a constant matrix representing the location of the first joint with respect to the base frame.

When the axes z_{i-1} and z_i are not parallel to each other, standard Denavit and Hartenberg rules can be used (Fig. 5) to position reference frames:

- Axis z_i coincides with $i+1$-th joint axis;
- Axis x_i is orthogonal both to z_i and z_{i-1} and directed from z_{i-1} to z_i;
- Axis y_i is chosen in order to create a right frame.

The base frame and the origin of the gripper frame are freely located.

The four D&H parameters of the i-th link are:

- The distance h_i between axes x_{i-1} and x_i which is called link *offset*;
- The distance l_i between axes z_{i-1} and z_i which is called link *length*;
- The angle φ_i between z_{i-1} and z_i which is called *twist*;
- The angle θ_i between x_{i-1} and x_i which is called *rotation*.

The relative location of frame i with respect to frame $i-1$ is then

$$A_i = R(z,\theta_i)T(z,h_i)T(x,l_i)R(x,\varphi_i) \tag{15}$$

where $R(u,\phi)$ is a rotation of ϕ around axis u and $T(u,t)$ is a translation t along u.

It can be noted that l_i and φ_i represent *intrinsic* geometric properties of the link, while θ_i and h_i are the joint motion and the assembly condition with respect to the previous link. This is quite evident for revolute joints, and similar considerations can be made for prismatic ones. As well known, for prismatic joints 2 out of the 4 parameters are redundant (Paul, 1981).

The D&H approach is often considered a good standardized way to describe robot kinematics. However calibration requires MCPC models and for several manipulator structures Eq. (15) must be modified as described in the following.

When two consecutive joint axes are parallel to each other the parameter h_i is not univocally defined and can be freely assigned. However if a small geometrical error equivalent to a rotation $\Delta\beta_i$ around y_i axis occurs, the joints are no longer parallel and $h_i \cong l_i/\Delta\beta_i$. So, for small variation of $\pm\Delta\beta_i$ around 0 the value of h_i changes abruptly from $-\infty$ to ∞ and therefore the model is no longer proportional.

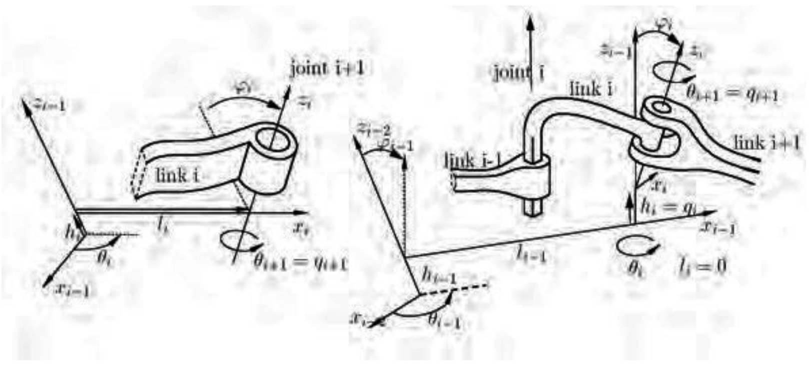


Fig. 5. Frames positions according to the Denavit and Hartenberg conventions. Case of i-th link with revolute (left) and prismatic joint (right).

In order to obtain a parametrically continuous model, when the i-th joint is revolute (Fig. 6a), the Hayati modification should be adopted (Hayati & Mirmirani, 1985)

$$A_i = R(z,\theta_i)T(x,l_i)R(x,\varphi_i)R(y,\beta_i) \tag{16}$$

while, when the i-th joint is prismatic (Fig. 6b), the **PR** modification is required:

$$A_i = T(z,l_i)T(y,h_i)R(x,\varphi_i)R(y,\beta_i). \tag{17}$$

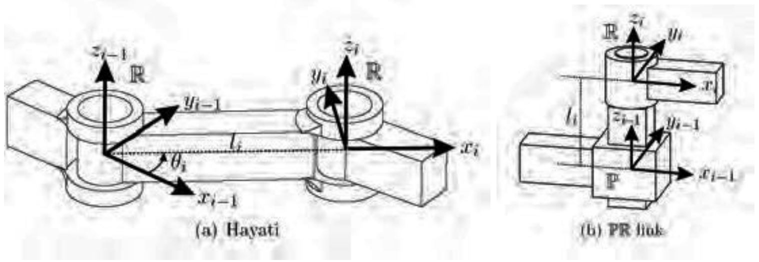


Fig. 6. Parallel joint axes: Hayati conventions for **RR** and **RP** (nearly)-parallel joint axes and the case of the **PR** link. In both cases the frames in the figures are represented for the nominal values of the rotations around x_i and y_i ($\varphi_i=\beta_i=0$).

In both cases the nominal values of φ and β are zero, but they can be used to represent errors. In brief we can write

$$A_i=\begin{cases} R(z,\theta_i)T(z,h_i)T(x,l_i)R(x,\varphi_i) & \text{'D\&H'} \quad \text{if } z_{i-1} \# z_i \\ R(z,\theta_i)T(x,l_i)R(x,\varphi_i)R(y,\beta_i) & \text{'Hayati'} \quad \text{if } z_{i-1}//z_i \text{ link RR or RP} \\ T(z,l_i)T(y,h_i)R(x,\varphi_i)R(y,\beta_i) & \text{'PR'} \quad \text{if } z_{i-1}//z_i \text{ link PR} \end{cases} \tag{18}$$

For prismatic joints, 4 parameters are used, however, to avoid redundancy, 2 of them suitably chosen for each specific robot are kept constant to their nominal value and eliminated from the calibration model (Mooring et al, 1991). The elimination process can be

performed using the algorithm described in Section 9.
Finally, in order to freely assign the gripper frame, the gripper matrix A_n must be generalized to contain 3 rotations and 3 translations. The expression of A_n depends on the *i*-th joint type J_n:

$$A_n = \begin{cases} R(z,\gamma_n)R(y,\beta_n)R(x,\alpha_n)\ T(z,c_n)T(y,b_n)T(x,a_n) & \text{'R6'} \quad \text{if } J_n = \mathbf{R} \\ T(z,c_n)T(y,b_n)T(x,a_n)\ R(z,\gamma_n)R(y,\beta_n)R(x,\alpha_n) & \text{'P6'} \quad \text{if } J_n = \mathbf{P} \end{cases} \tag{19}$$

The complete set of the geometrical parameters Λ is obtained by collecting all the variables used to describe the quoted matrices A_i. It is important to notice that some of the parameters coincide with the joints coordinates q_i:

$$\begin{array}{c|cc} & J_i = \mathbf{R} & J_i = \mathbf{P} \\ \hline i < n & q_i = \theta_i & q_i = h_i \\ i = n & q_i = \gamma_n & q_i = c_n \end{array} \tag{20}$$

6. Incremental Approach

When an incremental approach is adopted, Eq. (14) can be reformulated as

$$M = A_0 B_0 A_1 B_1 A_2 B_2 \ldots A_i B_i \ldots B_{n-1} A_n C \tag{21}$$

where matrices A_i describe the nominal geometry of the links and the joint motions while B_i and C describe geometric inaccuracy. Matrices A_i can be freely defined with the only constraint that z_i axis coincides with the $i-1$- th joint axis while B_i and C have the following form:

$$B_i = \begin{cases} R(x,\Delta\alpha_i)R(y,\Delta\beta_i)T(x,\Delta a_i)T(y,\Delta b_i) & \text{if } J_{i\text{-}1} = \mathbf{R} \\ R(x,\Delta\alpha_i)R(y,\Delta\beta_i) & \text{if } J_{i\text{-}1} = \mathbf{P} \end{cases}$$

$$C = R(x,\Delta\alpha_n)R(y,\Delta\beta_n)R(z,\Delta\gamma_n)\ T(x,\Delta a_n)T(y,\Delta b_n)T(y,\Delta c_n) \tag{22}$$

where Δa, Δb, Δc indicate translations respectively along x, y, and z axes and $\Delta\alpha$, $\Delta\beta$, and $\Delta\gamma$ describe rotations respectively around x, y, and z axes. Each matrix B_i represents the errors in the location of the axis z_i (i+1- th joint), while matrix C describes the errors in the gripper frame.
An alternative formulation for matrices B_i, proposed in (Zhuang & Roth, 1992) and (Zhuang et al 1992), is:

$$B_i(\Delta\theta_{xi}, \Delta\theta_{yi}, \Delta a_i, \Delta b_i) = \begin{bmatrix} 1-\dfrac{\Delta\theta_{xi}^2}{1+\Delta\theta_{zi}} & -\dfrac{\Delta\theta_{xi}\Delta\theta_{yi}}{1+\Delta\theta_{zi}} & \Delta\theta_{xi} & \Delta a_i \\ -\dfrac{\Delta\theta_{xi}\Delta\theta_{yi}}{1+\Delta\theta_{zi}} & 1-\dfrac{\Delta\theta_{yi}^2}{1+\Delta\theta_{zi}} & \Delta\theta_{yi} & \Delta b_i \\ -\Delta\theta_{xi} & -\Delta\theta_{yi} & \Delta\theta_{zi} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\Delta\theta_{zi} = \sqrt{1-\Delta\theta_{xi}^2-\Delta\theta_{yi}^2} \tag{23}$$

For prismatic joints $\Delta a_i = \Delta b_i = 0$.

Comparing this definition of B_i with that of Eq. (22), for small errors, we get $\Delta\theta_{xi} \cong \Delta\beta_i$ and $\Delta\theta_{yi} \cong -\Delta\alpha_i$.

The set of the robot parameter errors $\Delta\Lambda$ is obtained collecting all the parameters (Δa_i, Δb_i, Δc_i, $\Delta\alpha_i$, $\Delta\beta_i$, and $\Delta\gamma_i$) of all the matrices B_i and C of Eq. (22). The offset errors Δq_i in the joint coordinates do not enter in vector $\Delta\Lambda$ because they are redundant with the above mentioned parameters.

7. Comparison Between ED&H and Incremental Approaches

The two approaches presented in Sections 5 and 6 have different characteristics. The development of the incremental parameter set can be more easily automated because the need to deal with different situations is minimized. The only test to be performed is to distinguish between revolute and prismatic joints (Eq. 22).

On the contrary, the extended D&H approach has the advantage that the offset errors in the joint coordinates q_i are explicitly present in the models. In some cases this could be an advantage because joint coordinate errors are responsible for a great percentage of the manipulator accuracy error and they can be easily compensated even in simple controllers without specific calibration software. However in the ED&H approach to avoid singularities and redundancies two cases must be dealt properly: consecutive parallel joint axes, and prismatic joints; this approach requires a more complicated algorithm.

An approach where the joint offset errors are explicitly present could be useful when extending the methodology to manipulators where many joints are not actuated (e.g. PKM) and so the corresponding joint parameters must be removed from the model.

A methodology which combines the good characteristics of the two approaches is developed in the next Section.

8. A Modified Incremental Approach

When an incremental approach is desired and the explicit presence of the joint offsets is requested, the following two-step procedure can be adopted. First of all Eq. (21) is modified introducing for each joint a matrix D_i describing the coordinate offset errors Δq_i:

$$M = A_0 B_0 \; D_1 A_1 B_1 \; D_2 A_2 \ldots D_n A_n \; C \tag{24}$$

where matrices D_i are defined as

$$D_i = \begin{cases} R(z, \Delta q_i) & \text{if } \mathrm{J_i} = \mathbf{R} \\ T(z, \Delta q_i) & \text{if } \mathrm{J_i} = \mathbf{P} \end{cases} \tag{25}$$

Secondly, Eq. (24) is analyzed in order to remove from matrices B_i all the terms redundant to the just introduced joint offset errors. This elimination process can be performed using differential analysis (infinitesimal motions) as described in Section 9.

After introducing matrices D_i and removing redundancy from the matrices B_i, the final number of the parameters does not change matching again the value predicted by Eq. (11).

9. Elimination of the Redundant Parameters

The algorithm described in this Section has been adapted from (Khalil et al, 1991, Khalil & Gautier, 1991, Meggiolaro & Dubowsky, 2000).

The variation $\Delta\lambda_i$ of the i-th geometrical parameter causes a variation ΔM_i of the matrix $M(Q,\Lambda)$ that describes the gripper pose $\Delta M_i = \frac{\partial M}{\partial \lambda_i}\Delta\lambda_i$. The correspondent roto-translation of the gripper $\Delta S_i'$ can be expressed in the base frame as:

$$\Delta S_i' = \Delta M_i M^{-1} = \frac{\partial M}{\partial \lambda_i} M^{-1}\Delta\lambda_i = L_i\Delta\lambda_i = \left[\begin{array}{ccc|c} 0 & -d\gamma_i & d\beta_i & dx_i \\ d\gamma_i & 0 & -d\alpha_i & dy_i \\ -d\beta_i & d\alpha_i & 0 & dz_i \\ \hline 0 & 0 & 0 & 0 \end{array}\right] \tag{26}$$

with (Legnani, Casolo et al, 1996)

$$\frac{\partial M}{\partial \lambda_i} = L_i M \quad \Leftrightarrow \quad \frac{\partial M}{\partial \lambda_i} M^{-1} = L_i \tag{27}$$

where dx_i, dy_i and dz_i are the translations and $d\alpha_i$, $d\beta_i$ and $d\gamma_i$ are the rotations of the gripper generated by the parameter error $\Delta\lambda_i$. The generic form of matrices L_i is

$$L_i = \left[\begin{array}{ccc|c} 0 & -\delta\gamma & \delta\beta & \delta x \\ \delta\gamma & 0 & -\delta\alpha & \delta y \\ -\delta\beta & \delta\alpha & 0 & \delta z \\ \hline 0 & 0 & 0 & 0 \end{array}\right] \tag{28}$$

which simplifies to ${}^{r}L$ for rotational parameters and to ${}^{p}L$ for translational ones

$${}^{r}L = \left[\begin{array}{ccc|c} 0 & -u_z & u_y & t_x \\ u_z & 0 & -u_x & t_y \\ -u_y & u_x & 0 & t_z \\ \hline 0 & 0 & 0 & 0 \end{array}\right] \qquad {}^{p}L = \left[\begin{array}{ccc|c} 0 & 0 & 0 & u_x \\ 0 & 0 & 0 & u_y \\ 0 & 0 & 0 & u_z \\ \hline 0 & 0 & 0 & 0 \end{array}\right] \tag{29}$$

where $U = [u_x, u_y, u_z]^t$ is the unit vector of the axis of motion (rotation or translation). Moreover for rotations $T = -U \times P = [t_x, t_y, t_z]^t$ where $P = [p_x, p_y, p_z]^t$ is one point of the axis around which the rotation is performed.

The relation between the error $\Delta S = [dx, dy, dz, d\alpha, d\beta, d\gamma]^t$ of the gripper pose and all the N errors in the manipulator geometry Λ can be expressed by means of the jacobian J_Λ

$$\Delta S = J_\Lambda \, \Delta\Lambda \tag{30}$$

The jacobian can be constructed transforming each matrix L_i into the i-th column of J_Λ

$$J_\Lambda = \left[\begin{array}{c|c|c|c|c} j_1 & \dots & j_i & \dots & j_N \end{array}\right] \tag{31}$$

with $j_i = [t_x, t_y, t_z, u_x, u_y, u_z]^t$ for rotational parameters and $j_i = [u_x, u_y, u_z, 0, 0, 0]^t$ for the translation ones. The i-th parameter is redundant if the i-th column of J_Λ is a linear combination of some other columns.
Generally, it is not necessary to construct the whole jacobian. For simplicity it is sometimes convenient to construct only the part relative to the links under analysis expressing all the terms in any suitable (adjacent) link frame.
For example, we consider two links of a robot (Fig. 7) with three revolute joints. The pose of the $i+1$-th frame with respect to the $i-1$-th one can be expressed as

$$\begin{aligned} M= \; & T(x_{i-1},\Delta a_{i-1})T(y_{i-1},\Delta b_{i-1})R(x_{i-1},\Delta\alpha_{i-1})R(y_{i-1},\Delta\beta_{i-1})R(z_{i-1},q_i+\Delta q_i)T(x_{i-1},l_i)R(x_{i-1},\pi/2) \\ & T(x_i,\Delta a_i)T(y_i,\Delta b_i)R(x_i,\Delta\alpha_i)R(y_i,\Delta\beta_i)R(z_i,q_{i+1}+\Delta q_{i+1})T(x_i,l_{i+1})R(x_i,-\pi/2) \\ & T(x_{i+1},\Delta a_{i+1})T(y_{i+1},\Delta b_{i+1})R(x_{i+1},\Delta\alpha_{i+1})R(y_{i+1},\Delta\beta_{i+1})R(z_{i+1},q_{i+2}+\Delta q_{i+2}) \end{aligned} \tag{32}$$

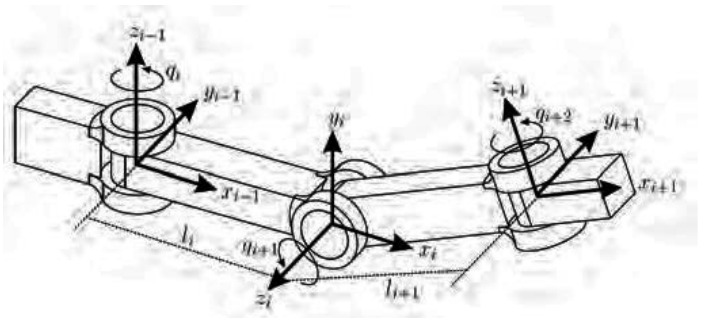


Fig. 7. Some links of a manipulator used to illustrate the elimination of the parameter redundant to Δq_i.

Suppose that we want to find the parameters redundant to Δq_i. For simplicity and according to Eq. (26) the jacobian is expressed in frame $i-1$ and so dx, dy and dz represent the displacement of one point embedded on frame $i+1$ which initially lies in the origin of frame $i-1$ (Legnani, Casolo et al, 1996). The relevant columns of J_Λ are presented in Table 2. The columns 5, 9, and 12 are linearly dependent to each other ($j_5 - j_9 - l_i j_{12} = 0$) and so one of the parameters Δq_i, $\Delta\beta_i$ or Δb_{i+1} must be eliminated from the model.
An alternative numerical approach for the elimination of the redundant parameters is presented in the example of Section 11. Another one is described in the chapter devoted to the calibration of PKM.

$J_\Lambda =$	$[j_1$	j_2	j_3	j_4	$\overset{*}{j_5}$	j_6	j_7	j_8	$\overset{*}{j_9}$	j_{10}	j_{11}	$\overset{*}{j_{12}}$	j_{13}	j_{14}	$j_{15}]$
	Δa_{i-1}	Δb_{i-1}	$\Delta \alpha_{i-1}$	$\Delta \beta_{i-1}$	Δq_i	Δa_i	Δb_i	$\Delta \alpha_i$	$\Delta \beta_i$	Δq_{i+1}	Δa_{i+1}	Δb_{i+1}	$\Delta \alpha_{i+1}$	$\Delta \beta_{i+1}$	Δq_{i+2}
dx	1	0	0	0	0	C_i	0	0	$l_i S_i$	0	$C_i C_{i+1}$	$-S_i$	...		
dy	0	1	0	0	0	S_i	0	0	$-l_i S_i$	0	$S_i C_{i+1}$	C_i	...		
dz	0	0	0	0	0	0	1	0	0	$-l_i$	S_{i+1}	0	...		
$d\alpha$	0	0	1	0	0	0	0	C_i	0	S_i	0	0	...		
$d\beta$	0	0	0	1	0	0	0	S_i	0	$-C_i$	0	0	...		
$d\gamma$	0	0	0	0	1	0	0	0	1	0	0	0	...		

Table 2. Relevant columns of the jacobian J_Λ of the manipulator of Figure 7 expressed in frame $i-1$, with: C_i=cos(q_i), S_i=sin(q_i), C_{i+1}=cos(q_{i+1}), S_{i+1}=sin(q_{i+1}).

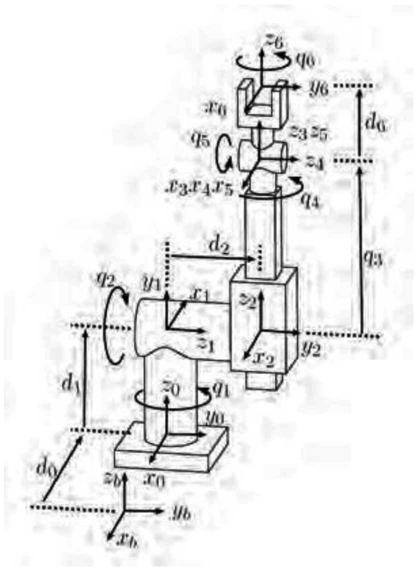


Fig. 8. The Stanford arm in its home position.

10. Example: Models of the Stanford Arm

The 6-dof Stanford arm (Fig. 8) is here analyzed as an example to construct the calibration parameters sets obtained applying the three quoted approaches.
The manipulator has 5 revolute joints and a prismatic one. The total number of parameters is $N = 6 + 4R + 2P = 6 + 4 \cdot 5 + 2 \cdot 1 = 28$.
Table 3 displays the parameter sets defined by the quoted approaches when external calibration is to be performed. The last two lines of the table displays the 12 external parameters to be removed from the model when internal calibration is required.
For the ED&H approach since the third link is prismatic, h_2 is redundant to h_3 and l_3 to h_4 and so two of them must be removed from the calibration model (in the example they are h_3 and h_4 indicated with parentheses in the table).
The modified incremental model is obtained from the incremental one observing that Δq_1 is redundant to $\Delta \beta_1$, Δq_2 is redundant to $\Delta \beta_2$, Δq_3 is redundant to Δb_4, Δq_4 is redundant to $\Delta \beta_4$, Δq_5 is redundant to $\Delta \beta_5$, and Δq_6 is redundant to $\Delta \gamma_6$.

Stanford Manipulator					
		Extended D&H		**Incremental**	**Modified Incr.**
link	**joint**	**type**	**nominal values**		
Base 0	-	Hayati	$\theta_0 = 0$, $l_0=d_0$, $\varphi_0=0$, $\beta_0=0$	$\Delta\alpha_0$, $\Delta\beta_0$ Δa_0, Δb_0	$\Delta\alpha_0$, $\Delta\beta_0$, Δa_0, Δb_0
1	R	D&H	$\theta_1=q_1$, $h_1=d_1$, $l_1=0$, $\varphi_1=90^\circ$	$\Delta\alpha_1$, $\Delta\beta_1$, Δa_1 Δb_1	Δq_1, $\Delta\alpha_1$, Δa_1, Δb_1
2	R	D&H	$\theta_2=q_2$ $h_2=d_2$, $l_2=0$, $\varphi_2=90°$	$\Delta\alpha_2$, $\Delta\beta_2$	Δq_2, $\Delta\alpha_2$
3	P	PR	$l_3=q_3$, $(h_3=0)$, $\varphi_3=0$, $\beta_3=0$	$\Delta\alpha_3$, $\Delta\beta_3$, Δa_3, Δb_3	Δq_3, $\Delta\alpha_3$, $\Delta\beta_3$, Δa_3, Δb_3
4	R	D&H	$\theta_4=q_4$, $(h_4=0)$, $l_4=0$, $\varphi_4=-90^\circ$	$\Delta\alpha_4$, $\Delta\beta_4$ Δa_4, Δb_4	Δq_4, $\Delta\alpha_4$, Δa_4
5	R	D&H	$\theta_5=q_5$, $h_5=0$, $l_5=0$, $\varphi_5=90^\circ$	$\Delta\alpha_5$, $\Delta\beta_5$ Δa_5, Δb_5	Δq_5, $\Delta\alpha_5$, Δa_5, Δb_5
Gripper 6	R	R6	$\gamma_6=q_6$, $\beta_6=0$, $\alpha_6=0$, $c_6=d_6$, $b_6=0$, $a_6=0$	$\Delta\gamma_6$, $\Delta\beta_6$, $\Delta\alpha_6$, Δc_6, Δb_6, Δa_6	Δq_6, $\Delta\beta_6$, $\Delta\alpha_6$, Δc_6, Δb_6, Δa_6
External parameters base			$\theta_0=0$, $l_0=d_0$, $\varphi_0=0$, $\beta_0=0$, $h_1=d_1$, Δq_1	$\Delta\alpha_0$, $\Delta\beta_0$, Δa_0 Δb_0, Δb_1, $\Delta\beta_1$	$\Delta\alpha_0$, $\Delta\beta_0$, Δa_0 Δb_0, Δb_1, Δq_1
External parameters gripper			$\gamma_6=\Delta q_6$, $\beta_6=0$, $\alpha_6=0$ $c_6=d_6$, $b_6=0$, $a_6=0$	$\Delta\gamma_6$, $\Delta\beta_6$, $\Delta\alpha_6$ Δc_6, Δb_6, Δa_6	Δq_6, $\Delta\beta_6$, $\Delta\alpha_6$, Δc_6, Δb_6, Δa_6

Table 3. The 28 parameters of the 3 MCPC models of the Stanford manipulator (external calibration) and the 12 to be removed for internal calibration. For the ED&H approach, the values after the '=' sign are the nominal values, parameters in parentheses are redundant.

11. Example: Calibration of a Measuring Robot

11.1. Introduction

The above described calibration procedures have been applied to a measuring robot (Fig. 9) whose kinematics is described in Section 11.2. The manipulator is used in a shoe industry.
The robot is made by steel, the structure is quite stiff, its weight is partially compensated by compressed air pistons, it is not subjected to loads during operation. High precision incremental encoders are directly connected to the 5 revolute joints. For these reasons a pure kinematics procedure was considered appropriated.

The measuring robot is used to manually measure the shape of an object by touching it with the robot end-effector. The robot is requested to measure the position of the distal point of the end-effector and the direction of its axis.
The dime displayed in Figure 10 is used to force the manipulator to known poses for calibration purposes.

11.2 The parameters set

Eq. (11) would suggest the identification of 26 parameters for a complete calibration. However the end effector rotation around its axis is negligible, then a total of 25 structural parameters must be identified (Eq. (12) with G=5).
The 5 DOF robot under analysis has a structure similar to that of a PUMA robot. The direct kinematic problem was solved using a D&H procedure. An inverse kinematic solution was also derived in analytical form for $\Lambda = \Lambda_n$. The joint displacements are measured by high resolution encoders with a resolution of 0.018 degrees for step, that gives a gripper repeatability of about ±0.17 mm.

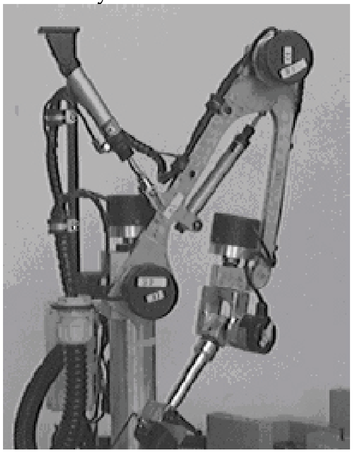


Fig. 9. The 5 DOF measuring robot.

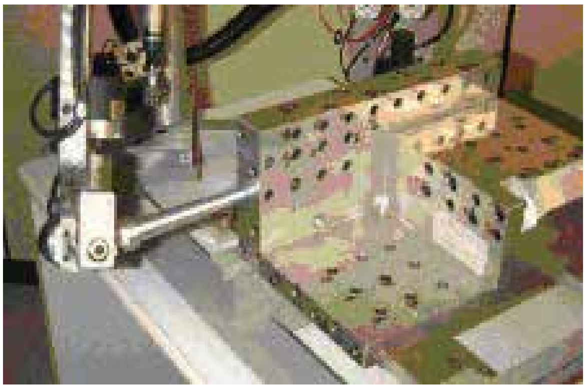


Fig. 10. Matching a pose using the dime.

Figure 11 shows a schematic view of the 5 DOF robot. Frame {0} is embedded on the robot base, while frames {1} to {5} are embedded on the corresponding link using the D&H convention (Denavit and Hartenberg, 1955) (axis z_i coincident with joint axis $i+1$).
The absolute reference frame {G} is the frame with respect to which each measure is taken. The nominal position of {G} is on the robot base with the z axis parallel to first joint axis, and it is coincident with frame {0}. z_0 is not parallel to z_G but its non-parallelism is caused by very small constant rotations δx_0 and δy_0 around x_G and y_G. The theoretical position of frame {0} with respect to {G} is $(X_0, Y_0, 0)^T$ and two position errors ΔX_0 and ΔY_0 are identified. Translations in the z direction are incorporate in the length of link 1.

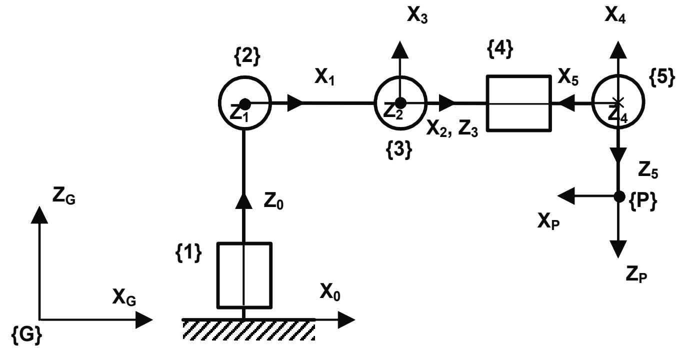


Fig. 11. Frame positions.

The pose of frame {0} with respect to {G} is then represented by A_{G0}:

$$A_0 = \begin{bmatrix} c_\psi & 0 & s_\psi & X \\ s_\chi \cdot s_\psi & c_\chi & -s_\chi \cdot c_\psi & Y \\ -c_\chi \cdot s_\psi & s_\chi & c_\chi \cdot c_\psi & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad \begin{aligned} c_\alpha &= \cos(\alpha) \\ s_\alpha &= \sin(\alpha) \\ \chi &= \chi_0 + \delta\chi_0 \\ \psi &= \psi_0 + \delta\psi_0 \\ X &= X_0 + \Delta X_0 \\ Y &= Y_0 + \Delta Y_0 \end{aligned} \tag{33}$$

According to Eq. (15), with the exception of A_2, we get:

$$A_i = \begin{bmatrix} c_\vartheta & -s_\vartheta \cdot c_\alpha & s_\vartheta \cdot s_\alpha & a \cdot c_\vartheta \\ s_\vartheta & c_\vartheta \cdot c_\alpha & -c_\vartheta \cdot s_\alpha & a \cdot s_\vartheta \\ 0 & s_\alpha & c_\alpha & d \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad \begin{aligned} \vartheta &= \vartheta_i + \delta\vartheta_i \\ \alpha &= \alpha_i + \delta\alpha_i \\ a &= a_i + \Delta a_i \\ d &= d_i + \Delta d_i \\ i &= 1,3,4,5 \end{aligned} \tag{34}$$

Since joint axes 2 and 3 are parallel, in order to avoid singularities in the calibration procedures, the transformation between frame {1} and {2} should be expressed accordingly, as described in Section 5. A suitable formulation is:

$$A_2 = R(z_1, \vartheta_2 + \delta\vartheta_2) \cdot T(z_1, d_1) \cdot T(x_2, a_2 + \Delta a_2) \cdot R(x_2, \alpha_2 + \delta\alpha_2) \cdot R(y_2, \psi_2 + \delta\psi_2)$$

We get:

$$A_2 = \begin{bmatrix} c_\vartheta \cdot c_y - s_\vartheta \cdot s_\alpha \cdot s_y & -s_\vartheta \cdot c_\alpha & c_\vartheta \cdot s_y + s_\vartheta \cdot s_\alpha \cdot c_y & a \cdot c_\vartheta \\ s_\vartheta \cdot c_y + c_\vartheta \cdot s_\alpha \cdot s_y & c_\vartheta \cdot c_\alpha & s_\vartheta \cdot s_y - c_\vartheta \cdot s_\alpha \cdot c_y & a \cdot s_\vartheta \\ -c_\alpha \cdot s_y & s_\alpha & c_\alpha \cdot c_y & d \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad \begin{aligned} \vartheta &= \vartheta_2 + \delta\vartheta_2 \\ \alpha &= \alpha_2 + \delta\alpha_2 \\ a &= a_2 + \Delta a_2 \\ d &= d_2 \\ y &= y_2 + \delta y_2 \end{aligned} \tag{35}$$

Finally, the pose of the gripper frame is:

$$A_6 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & Z_P + \Delta Z_P \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{36}$$

The gripper pose is represented by a product of the position matrixes:

$$M = A_0 \cdot \prod_{i=1}^{6} A_i$$

Therefore the complete set of the parameter to be estimated is:

$$\begin{aligned}\Lambda = [\ & X_0 \ \ Y_0 \ \ \chi_0 \ \ \psi_0 \ \ \vartheta_1 \ \ d_1 \ \ a_1 \ \ \alpha_1 \ \ \vartheta_2 \ \ a_2 \ \ \alpha_2 \ \ \psi_2 \\ & \vartheta_3 \ \ d_3 \ \ a_3 \ \ \alpha_3 \ \ \vartheta_4 \ \ d_4 \ \ a_4 \ \ \alpha_4 \ \ \vartheta_5 \ \ d_5 \ \ a_5 \ \ \alpha_5 \ \ Z_p \ \ ]^T\end{aligned} \tag{37}$$

And the nominal value of the parameters are:

$$\begin{aligned}\Lambda = [\ & 0 \ \ 0 \ \ 0° \ \ 0° \ \ 0° \ \ 252. \ \ 0 \ \ 90° \ \ 0° \ \ 0 \ \ 320. \ \ 0° \\ & 90° \ \ 0 \ \ 0 \ \ 90° \ \ 0° \ \ 330. \ \ 0 \ \ 90° \ \ -90° \ \ 0 \ \ 0 \ \ 90° \ \ 161. \ \ ]^T\end{aligned}$$

Length are given in millimetres and angles in degrees.

11.3 Results and Discussion

To verify the numerical optimization algorithms, as better described in (Omodei et al. 2001), we initially tested them with simulated measures created as follows. A number of joint rotations $Q^{\#}$ were chosen and the corresponding gripper poses $S^{\#}$ were evaluated after giving an arbitrary value to the structural parameter errors $\Delta\Lambda^{\#}$. The programs under test were asked to estimate the structural errors $\Delta\Lambda^{\#}$ assuming $Q_m = Q^{\#}$ and $S_m = S^{\#}$; $\Delta\Lambda^{*}$ should be identical to $\Delta\Lambda$. Before running the estimation procedure, the joint coordinates were corrupted adding random errors to simulate uncertainties in the measuring system. The convergence of the procedure and the achieved accuracy was estimated comparing S_d with $F(Q_d, \Lambda_n+\Delta\Lambda)$. The results show that if only geometric inaccuracy is present, it is possible to reach a robot accuracy close to the measuring error.

As a final step, experimental calibration was performed on the actual system (Fig.s 9 and 10). The measuring robot was forced to reach a set of predefined known poses and, for each of them, the corresponding joint rotation was measured.

A precision dime was manufactured by a CNC machine and holes created on different surfaces. The position and orientation of each hole is precisely known (it was measured by CMM machine). During the calibration procedure the operator inserts the gripper pin into the holes and collects the joint angles. These values are recorded together with the correspondent theoretic gripper pose position. In our case, for each pose we can measure the joint coordinates (5 data) and so we must repeat the measurement for a minimum of 25/5 = 5 poses. However to improve the calibration accuracy and to cover the whole working area with different gripper orientation the measuring dime was designed to have 72 insertion holes. Since some of the holes can be reached with two different robot configurations, so a total of 81 poses can be collected.

To estimate the robot repeatability, all the dime poses were recorded twice by two different operators recording the correspondent joint rotations. The gripper positions were evaluated with the nominal value of the parameters and the difference between the two sets was evaluated. The maximum difference was 0.176 mm, the average was $\overline{d}$ = 0.082 mm and its standard deviation was σ = 0.062 mm. Borrowing a definition from international standards (ISO 9283), we estimated the robot repeatability as $\overline{d} + 3\sigma = \pm\ 0.268$ mm.
The calibration procedure was performed as follows.
A total of 81 poses were collected (we call these data the *complete set*). The complete set was divided into two sub-sets: the *calibration set* and the *control set*. The calibration set was used as input for the program which evaluates the geometric parameters. The control set was used to verify the quality of the calibration.
In order to investigate the algorithm convergence, all the mathematical procedures were repeated 3 times using different sizes of the calibration and control set:

1. 40 calibration poses, 41 control poses;
2. 60 calibration poses, 21 control poses;
3. 81 calibration poses.

To verify the quality of the calibration, for each of the 81 poses we evaluated the distance between the known gripper positions with those estimated using the measured joint rotations and the evaluated structural parameter errors.
Table 4 contains the average residual, its maximum value and the standard deviation evaluated for all the situations considered. Considering the complete set the average position error before calibration procedure was about 4.702 mm with a standard deviation of 1.822 mm and 8.082 mm as maximum value.

	Calibration set (mm)				**Control set (mm)**			**Complete set (mm)**		
	Poses	average	s. dev.	max	average	s. dev.	max	average	s. dev.	max
a	40	4.789	1.784	8.082	4.616	1.854	7.572	4.702	1.822	8.082
b	60	4.692	1.785	7.574	4.732	1.921	8.082	4.702	1.822	8.082
c	81	4.702	1.822	8.082	-----	-----	-----	4.702	1.822	8.082

Table 4. Position errors (residual) before calibration procedure.

Table 5 presents the results of the calibration process performed using three algorithms and different number of poses. For each case we give the average position error (E_{op}, E_{it}, E_{ka}), their standard deviation and the maximum error.
The column labelled with 'time' represents the time necessary to process the data with a PC Pentium 100 MHz, highlighting the different efficiency of the algorithms.

		Calibration Set (Mm)				Control Set (Mm)			Complete Set (Mm)		
Alg.	Time	Poses	Average	S. Dev.	Max	Average	S. Dev.	Max	Average	S. Dev.	Max
amoeba	07′ 33″	40	0.310	0.196	0.894	0.301	0.178	0.775	0.305	0.187	0.865
	05′ 53″	60	0.399	0.261	1.127	0.461	0.192	0.828	0.415	0.246	1.117
	11′ 02″	81	0.316	0.249	1.069	-----	-----	-----	0.316	0.249	1.069
kalman	00′ 11″	40	6.004	4.579	11.69	5.431	4.597	11.81	5.714	4.597	11.81
	00′ 14″	60	6.676	3.048	11.54	6.931	2.611	13.99	6.742	2.945	13.99
	00′ 19″	81	5.767	2.675	11.85	-----	-----	-----	5.767	2.675	11.85
linear	00′ 06″	40	0.820	0.466	2.159	0.726	0.385	1.799	0.773	0.430	2.159
	00′ 08″	60	0.434	0.245	1.123	0.529	0.254	1.229	0.459	0.251	1.229
	00′ 13″	81	0.632	0.328	1.700	-----	-----	-----	0.632	0.328	1.700

Table 5. Comparison between calibration algorithms: residual error (25 parameters).

A comparison between the residual error between the calibration, the control and the complete set gives good information about the final results. As obvious the residuals in the control set are generally worst than those in the calibration set; the residuals of the complete set is an average of the two. However differences are small assuring that the number of the considered poses was sufficient to calibrate the robot on the considered working volume. This conclusion is supported by the fact that the results do not vary very much increasing the calibration set from 40 to 81 poses.

The results prove the effectiveness of the algorithms called '*amoeba*' and '*linear*' which reduce the gripper pose to about 1/8 of the initial value.

On the contrary the kalman filter gave worst results, though with other robots, this algorithm performed very well.

The robot is designed to operate only a subset of its working area and the calibration procedure cover just this part.

This means that some of the robot parameters could possibly be un-distinguible to others, because produce almost the same gripper pose error. This fact can reduce the effectiveness of the calibration algorithms which work badly with redundant parameters.

For these reason we decide to perform the robot calibration estimating just a reduced set of the parameter errors. We remember that complete convergence of all parameters is not necessary for a robot to be accurate within the calibrated area. However if the robot is operated out of the calibrated area, the accuracy will depend on how well each parameter has converged. The presented methodologies calculate a robot model, which best fits the measured data, which is not necessarily the correct model. We empirically selected the parameters that we consider more important for the robot accuracy. The choice of the parameters was based on a visual inspection of the robot structure. This subset (called the *reduced set*) contains the robot base position and orientation, the joint offsets, and the link length: only 14 of 25 parameters are considered

$$\begin{aligned}\Lambda^* = [\ & \Delta X_0 \quad \Delta Y_0 \quad \delta\chi_0 \quad \delta\psi_0 \quad \delta\vartheta_1 \quad \Delta d_1 \quad \delta\vartheta_2 \\ & \Delta a_2 \quad \delta\vartheta_3 \quad \delta\vartheta_4 \quad \Delta d_4 \quad \delta\vartheta_5 \quad \Delta d_5 \quad \Delta Z_p \quad]^T\end{aligned} \tag{38}$$

The algorithms supplied an estimation of the structural parameters Λ^*. These values were utilised to predict the pose of the gripper used during the measuring session. The difference between the estimated and the predicted pose positions are shown in Table 6.

		Calibration set (mm)				Control set (mm)			Complete set (mm)		
Alg.	Time	Poses	average	s. dev.	max	Average	s. dev.	Max	average	s. dev.	max
amoeba	02′ 11″	40	0.586	0.195	0.921	0.538	0.221	0.963	0.561	0.210	0.963
	03′ 52″	60	0.445	0.195	1.122	0.528	0.212	1.208	0.467	0.203	1.208
	07′ 38″	81	0.294	0.189	0.926	-----	-----	-----	0.294	0.189	0.926
kalman	00′ 04″	40	1.423	0.706	2.770	1.285	0.659	2.733	1.353	0.686	2.770
	00′ 07″	60	4.864	2.231	12.62	5.361	1.777	11.75	4.993	2.134	12.62
	00′ 11″	81	5.276	2.331	13.07	-----	-----	-----	5.276	2.331	13.07
linear	00′ 03″	40	0.711	0.214	1.256	0.669	0.214	1.129	0.690	0.215	1.254
	00′ 06″	60	0.691	0.271	1.462	0.784	0.258	1.289	0.715	0.270	1.463
	00′ 07″	81	0.680	0.266	1.485	-----	-----	-----	0.680	0.266	1.485

Table 6. Comparison between calibration algorithms: residual error (reduced set of 14 parameters).

The results are not very different from those obtained with the complete set of 25 parameters. In some cases they are even better. This mean that some redundant parameter was present obstructing the convergence of the calibration process.

As a final test we designed an improved calibration program which is able to decide which parameters should be important.

The program works in this way. Initially the program performs the calibration using the reduced set of parameters $\Delta\Lambda^*$ and the corresponding residual error E^* is evaluated. We indicate the number of parameters in the complete set as n_c, while only n_r of them are present in the reduced set. Then the calibration process is repeated n_c - n_r times considering at each time the reduced set plus one of the others additional parameters. For each of this estimation a new value of E_i^* is evaluated. The parameter which generate the lower value of E_i^* is selected. If E_i^* is significantly lower than E^*, the *i-th* parameter is added to the reduced set and the estimation procedure is repeated from beginning.

		Calibration set (mm)				**Control set (mm)**			**Complete set (mm)**		
Alg.	Time	Poses	average	s. dev.	max	Average	s. dev.	Max	average	s. dev.	max
amoeba	1h 09′	40	0.238	0.175	0.693	0.240	0.153	0.722	0.239	0.164	0.722
	1h 49′	60	0.209	0.183	0.939	0.250	0.207	0.977	0.219	0.190	0.977
	1h 44′	81	0.232	0.182	0.891	-----	-----	-----	0.232	0.182	0.891
kalman	02′ 20″	40	1.371	0.652	2.496	1.233	0.630	2.563	1.301	0.645	2.563
	04′ 43″	60	5.231	2.610	12.14	5.697	2.249	11.16	5.352	2.530	12.14
	07′ 21″	81	5.614	2.846	13.68	-----	-----	-----	5.614	2.846	13.68
linear	00′ 52″	40	0.711	0.214	1.256	0.669	0.214	1.129	0.690	0.215	1.256
	03′ 00″	60	0.469	0.185	0.896	0.536	0.183	1.022	0.487	0.187	1.022
	10′ 23″	81	0.409	0.185	0.991	-----	-----	-----	0.409	0.185	0.991

Table 7. Comparison between calibration algorithms: residual error (reduced set plus automatically selected parameters).

The iterative process is terminate when the inclusion of a new parameter improves the residual error less than 5%. Results obtained with this procedure are presented in Table 7.

Table 8 shows the parameters that have been added in the different trials; Δa_3 and $\delta\alpha_3$ seems more important because they have been selected more frequently by the algorithms.

Comparing Table 7 with Table 5 it is clear that the last version of the algorithm is slower but gives the best results producing lower residual errors. For example the '*amoeba*' algorithm the average residual on the complete set is reduced from 0.3 - 0.4 mm to about 0.2 mm. This means that the calibration procedure works better if the redundant parameters are removed.

		Added Parameters										
Algorithm	Poses	Δa_1	$\delta\alpha_1$	$\delta\alpha_2$	δy_2	Δd_3	Δa_3	$\delta\alpha_3$	Δa_4	$\delta\alpha_4$	Δa_5	$\delta\alpha_5$
amoeba	40	*x*					*x*	*x*				
	60	*x*		*x*	*x*		*x*	*x*	*x*			*x*
	81						*x*	*x*				
kalman	40					*x*						
	60								*x*			
	81					*x*						
linear	40											
	60					*x*		*x*				
	81	*x*				*x*	*x*	*x*	*x*	*x*	*x*	*x*

Table 8. Parameters automatically added to the reduce set Λ^*.

12. Conclusions

All the steps necessary to perform a kinematic calibration of serial manipulators have been discussed and applied to an actual manipulator.
This first part of the chapter has compared and extended two procedures for the identification of the geometrical parameters necessary to describe the inaccuracies in the kinematic structure of a *generic serial robot*. The discussion led to a new methodology called 'Modified Incremental' which holds the positive characteristics of the others. Each procedure generates a Minimum, Complete and Parametrically Continuous set of parameters.
First of all the formula for the determination of the total number of parameters has been discussed pointing out the distinction between *internal* and *external* parameters.
The D&H approach for the parameters identification has been extended to deal with all the special cases (adjacent parallel joint axes, prismatic joints and gripper frame). This approach has the good quality to include the joint offsets in the parameter set and shows that some parameters (l and φ) are *intrinsic* of the link while the others represent the joint motion and the assembly condition.
The Incremental approach is the most simple to be applied (only the type of the joints must be considered) and can be more easily automatized. The main drawback is that the joint offsets are not explicitly included in the parameters set. The Modified Incremental approach solves this problem and the proposed procedure for the elimination of the redundant parameters ensures the Minimality of the model.
The last part of this chapter fully describes a practical calibration of a measuring robot discussing different estimations algorithms and their performances.
All these concepts will be reviewed in the next chapter in order to be applied to a *generic parallel manipulator* .

13. Acknowledgment

This research work has been partially supported by MIUR grants cofin/prin (Ministero Università e Ricerca, Italia).

14. References

Cleary, W. (1997). Robot Calibration, Development and Testing of a Laser Measurement System, *Honours thesis*, The University of Western Australia, October 1997.

Denavit, J. & Hartenberg, R. S. (1955). Kinematic Modelling for Robot Calibration, *Trans. ASME Journal of Applyed Mechanics*, Vol. 22, June 1955, pp. 215 - 221.

Everett, L. J.; Driels, M. & Mooring B. W. (1987). *Proceedings of IEEE Int. Conf. on Robotics and Automation*, pp. 183-189, Vol. 1, Raleig NC, Mar. 1987.

Flannery B.P., Press W.H., Teukolsky S.A., Vetterling W.T., *Numerical Recipes in C, The Art of Scientific Computing*, Cambridge University Press, 1992.

Hayati, S. & Mirmirani, M. (1985). Improving the Absolute Positioning Accuracy of robot manipulators, *Journal of Robotic System*, Vol. 2, No. 4, page pp . 397-413.

International Standard ISO 9283 Manipulating industrial Robots_Performance criteria and related test methods.

Legnani, G. ; Casolo, F. ; Righettini, P. & Zappa B. (1996). A Homogeneous Matrix Approach to 3D Kinematics and Dynamics. Part 1: theory, Part 2: applications, *Mechanisms and Machine Theory*, Vol. 31, No. 5, pp. 573-605.

Legnani G., J. P. Trevelyan (1996), "Static Calibration of Industrial Manipulators: a Comparison Between Two Methologies", Robotics Towards 2000 27th Int. Symp on Industrial Robots, pp.111-116, 6-8 October 1996.

Khalil, W. ; Gautier, M. & Enguehard, C. (1991). Identifiable parameters and optimum configurations for robot calibration, *Robotica*, Vol.9, pp. 63-70.

Khalil, W. & Gautier, M. (1991). Calculation of the identifiable parameters for robot calibration, *Proceedings of IFAC 91*, pp. 888-892.

Meggiolaro, M. & Dubowsky, S. (2000). An Analytical Method to Eliminate the Redundant Parameters in Robot Calibration, *Proceedings of International Conference on Robotics and Automation (ICRA '2000)*, IEEE, pp. 3609-3615, San Francisco, CA, USA.

Mooring, B.W.; Roth, Z. S. & Driels, M. R. (1991). *Fundamentals of manipulator calibration*, John Wiley & Sons, Inc., USA.

Omodei, A. ; Legnani, G. & Adamini, R. (2001). Calibration of a Measuring Robot: Experimental Results on a 5 DOF Structure, *Journal of Robotic System* Vol.18 No.5, pp. 237-250.

Paul, R. (1981*). Robot Manipulators: Mathematics, Programming, and Control*, Mit Press, 1981.

Reinhar, G.; Zaeh, M. F. & Bongardt, T. (2004). Compensation of Thermal Errors at Industrial Robots, *Prod. Engineering* Vol. XI, No.1.

Wildenberg, F. (2000). Calibrations for Hexapode CMW, *Proceedings of* 2nd *Chemnitzer Parallel kinematik-Seminar: Working Accuracy of Parallel Kinematics*, pp. 101-112, Verlag Wissenschaftliche Scripten.

Wang, J. ; Masory, O. & Zhuang, H. (1993). On the Accuracy of a Stewart Platform-Part II: Kinematic Calibration and Compensation, *Proceedings of IEEE Int. Conf. on Robotics and Automation*, Atlanta May 1993.

Zhuang, H. & Roth, Z. S. (1992), Robot calibration using the CPC error model, *Robotics & Computer-Integrated Manufacturing*, Vol. 9, No. 3, pp. 227-237.

Zhuang, H. ; Roth, Z. S. & Hamano, F. (1992). A Complete and Parametrically Continuous Kinematic Model for Robot Manipulator, *IEEE Transactions on Robotics and Automation*, Vol. 8, No. 4, August 1992, pp. 451-463.

Calibration of Parallel Kinematic Machines: Theory and Applications

Giovanni Legnani[1], Diego Tosi[1], Riccardo Adamini[1], Irene Fassi[2]
[1]*Dip. Ing. Meccanica -- Università di Brescia Via Branze 38, 25123 Brescia -- Italy*
giovanni.legnani@ing.unibs.it
[2]*ITIA-CNR Institute of Industrial Technology and Automation,*
National Research Council V.le Lombardia 20/A 20131 Milano - Italy

1. Introduction

As already stated in the chapter addressing the calibration of serial manipulators, kinematic calibration is a procedure for the identification and the consequent compensation of the geometrical pose errors of a robot. This chapter extends the discussion to Parallel Manipulators (also called PKM Parallel Kinematic Machines). As described in the following (Section 2) this extension is not obvious but requires special care.
Although for serial manipulators some procedures for the calibration based on automatic generation of a MCPC (Minimum Complete Parametrically Continuos) model exist, for PKMs only methodologies for individual manipulators have been proposed but a general strategy has not been presented since now. A few examples of the numerous approaches for the calibration of individual PKMs are proposed in (Parenti-Castelli & Di Gregorio, 1995), (Jokiel et al., 2000) for direct calibration and (Neugebauer et al., 1999), (Smollett, 1996) for indirect or self calibration techniques.
This paper makes one significant step integrating available results with new ones and reordering them in simple rules that can be automatically applied to any PKM with general kinematic chains. In all the cases a MCPC kinematic model for geometrical calibration is automatically obtained.
In Section 2 the main features of PKMs calibration is pointed out and the total number of the necessary parameters is determined; this is an original contribution. In Sections 3 and 4 two novel approaches for the generation of a MCPC model are described. Sections 5 and 6 are dedicated to the analysis of the singular cases and to the procedure for the elimination of the redundant parameters respectively; actual cases are discussed. Section 7 presents several examples of application of the two proposed procedures to many existing PKMs. Section 8 eventually draws the conclusions.
Note*: in this chapter it is assumed that the reader has already familiarised himself with the notation and the basic concepts of calibration described in the chapter devoted to serial manipulators.*

2. PKMs Calibration

2.1 Differences with respect to serial robots

The identification of the parameter set to be utilized for parallel manipulators can be performed generalizing the methodology adopted for the serial ones. However some remarkable differences must be taken into account, namely:

- most of the joints are *'unsensed'*, that is the joint motion is not measured by a sensor;
- PKMs make use of multi-degree of freedom joints (cylindrical, spherical);
- some links have more than two joints;
- links form one or more closed kinematic loops;
- PKMs can be calibrated by 'internal calibration' which does not require an absolute external measuring system.

Internal calibration (also called *indirect* or *self* calibration) is performed using the measure of extra-sensors monitoring the motion of non actuated joints and comparing the sensor readings with the values predicted using the nominal manipulator kinematics (Parenti-Castelli & Di Gregorio, 1995), (Ziegert et al., 1999), (Weck et al, 1999). In these cases, some of the manipulator parameters cannot be identified and just a partial robot calibration can be performed. A typical full internal calibration is able to identify all the 'internal parameters' (relative position of the joints, joints offsets) but not the location of the arbitrary 'user' frames of the fixed and of the mobile bases (6 parameters for the base and 6 for the gripper).
External calibration (also called *direct* calibration) mainly consists in calibrating the pose of the frame attached to the mobile base with respect to that of the fixed base, and it is similar to the calibration of serial manipulators (Neugebauer et al., 1999), (Smollett, 1996). To perform it, it is necessary to measure the 6 absolute coordinates (3 translations and 3 rotations) of the mobile base. When the instrumentation measures less than 6 coordinates (e.g. when calibration is performed using a double ball bar - DBB), some of the robot parameters cannot be identified and a proper complete external calibration is not possible.

2.2. Identification of the number of the parameters

The number of the parameters necessary to describe a PKM can be very high (e.g. up to 138 for a 6 d.o.f. Stewart-Gough platform). However with standard geometrical dimension of the links, the effect of some of them is usually small and so the corresponding parameters can be neglected. A complete discussion of this 'reduction' is outside the scope of this paper which is devoted to identify all the parameters theoretically necessary. However some of the most common situations will be mentioned in Sections 5.2, 6.2 and 7.8.
The first result that should be achieved is the identification of the number of the parameters necessary to construct a MCPC model.

R revolute joint	**P** prismatic joint	**SS** number of singular links
S spherical joint	**C** cylindrical joint	**F** number of reference frames
L number of kin. loops	J_i i-th joint	**E** number of encoder (or position sensors)

Table 1 Symbols and abbreviations.

For convenience we report the formula that gives the total number N of the parameters for a generic serial robot

$$N = 4R + 2P + 6 \tag{1}$$

while the number of the internal ones is

$$N_i = N - 12 = 4R + 2P - 6 \tag{2}$$

In literature, the only available general result for PKMs is due to Vischer (Visher, 1996) who suggests to generalize Eq.s (1) and (2) as

$$N = 3R + P + SS + E + 6L + 6(F-1) \tag{3}$$

where SS is the number of the links composed simply by two spherical joints connected by a rod, E is the number of the encoders (or of the joint sensors), L is the number of the independent kinematic loops and F is the number of the arbitrary reference frames. However Vischer does not give a proof of Eq. (3) but simply states that *'this equation has been empirically tested on several examples and seems to be valid...'*. However some cases are not included and in the following we prove that the correct equation is

$$N = 3R + P + 2C + SI + E + 6L + 6(F-1) \tag{4}$$

where C is the number of *cylindrical* joints and SI is the number of 'singular' links which include the SS links with $SI = +1$ (Section 5.1) and the **SPS** legs with $SI = -1$ (Section 5.2). As proved in the following, spherical joints do not require any parameter and their number is not present in Eq. (4).
Eq. (4) which works both for serial and parallel manipulators can be explained generalizing the Eq. (1) and Eq. (2) given for serial robots. First of all, in the case of serial manipulators we get $SS = 0$, $E = R + P$, $L = 0$ and $F = 2$ for external calibration while $F = 0$ for internal calibration and so Eq. (3) reduces to Eq.s (1) and (2).
Moreover it is evident that, after choosing an absolute reference frame, each further 'user' frame needs 6 parameters (3 rotations and 3 translations) to calibrate its pose; this explains the term $6(F-1)$.
To explain the term $6L$ we initially consider a serial robot with two 'branches' (Fig. 1 left). Its calibration can be performed considering two kinematic chains $B1$ and $B2$ from the base frame to the two gripper frames (Khalil et a., 1991). Each joint requires the usual number of parameters, however since we have two grippers the total number of external parameters is incremented by 6 and Eq. (1) is transformed as $N = 6 + 6 + 4R + 2P$. If now we 'weld' link m to link d, the number of the parameters is clearly unchanged because the parameters describing the links geometry are still the same and the 6 parameters just added are used to describe the location of the 'welding' between the links $d-m$. We conclude that for each closed loop we add 6 parameters to Eq. (1).

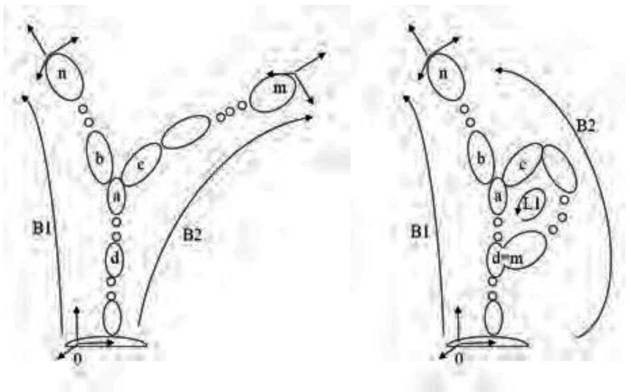


Fig. 1. A serial robot with two branches (left) and one with a closed loop (right).

As a further step we remember that for serial manipulators using the ED&H approach one parameter for each link corresponds to the joint variable. Now since in general in the PKMs

most of the joints are unsensed, we must remove one parameters for each of them; so the term $4R+2P$ of Eq. (1) transforms to $3R+P+E$ (if all the joints are sensed it is $E=R+P$).
A cylindrical joint can be considered as composed by a revolute joint plus a prismatic one so, apparently, it would require $3+1=4$ parameters. However the direction of the two joint axes coincides and so two parameters must be removed; this explains the term $2C$.
Universal joints are kinematically equivalent to a sequence of 2 revolute joints, and can be modelled accordingly.
Spherical joints (ball and socket) can be realised with high accuracy and so they can be often considered ideal and their modellisation requires only three coordinates to represent the location of their center. However spherical joints have 3 DOF and are always unsensed and so we must subtract three parameters. As a result we get *zero* and so S joints do not appear in Eq. (4).
The explication of term SI is more complex and will be analysed in Section 5. At the moment we just observe that for SS links it is evident that the rod length has to be counted as parameter. We also observe that PKMs with SS links hold internal degrees of freedom because the rods can freely rotate about their axes. Other 'singular links' will be analyzed in Section 5.

2.3 Comparison between different approaches

By extending the discussion given for serial manipulators in the first part of the work it follows that, since generally in any PKM there are unsensed joints, the Incremental approach will produce a model which is not minimum. It is so necessary to refer to the ED&H or to the Modified Incremental ones.
The adoption of ED&H approaches has the good point that the joint variables are explicitly present in the model and it is then easy to eliminate the joint offsets of the unsensed joints. However it is necessary to define new rules to fix the frames onto the links to take into account the presence of **S** and **C** joints. Some manipulators present singular cases that must be treated in a special way. They may result in the temporary insertion of redundant parameters to be removed in a second time.
On the other side, with the modified incremental approach it is nearly impossible to develop an algorithm for a general PKM that automatically generates a 'minimum' model. It is always necessary to generate a model that contains some redundancy and then eliminate them. This operation can be difficult because the jacobian to be analyzed could be quite large since it has a number of columns equal to the number of the parameters and a number of rows equal to the number of the scalar equations ($6(L+F-1)$). As already mentioned a Stewart-Gough PKM with **URPU** legs has 138 parameters and 5 loops producing a jacobian matrix with 36 rows and 138 columns. However this procedure can be automatized and it is sometime possible to work just on a part of the jacobian as explained for the serial robots in the relative chapter and extended to PKMs in Section 6 of this chapter.

3. An Extended D&H Approach for PKMs

3.1 Description of the procedure

The extension to PKMs of the ED&H methodology proposed for serial manipulators is based on the adoption of the multi frame approach (Section 9) which requires the definition on each link (fixed and mobile bases included) of a frame for each joint. One of them will be called '*intrinsic*' frame of the link and the other '*auxiliary*' frames of the joint. The univocal definition of the intrinsic frame, discussed further on, allows to verify once for all the minimality of the parameters used to describe the relative position of the joint auxiliary frames.

We indicate with H the matrices that describe the relative position of two frames embedded onto the same link (link transformation) while matrices G describe the relative location of the frames of two links connected by a joint, that is the assembly an the motion of the joint (Fig. 2 and Fig. 3). The matrices H depend on some geometrical parameters called '*link parameters*'.
The fixed base and the mobile base have an extra '*user*' frame. Their pose with respect to the intrinsic frame will be described by matrices E_F (fixed base) and E_M (mobile base).

A PKM with F user frames and L loops requires $L+F-1$ kinematics equations: one for each (closed) loop and one for each user (gripper) frame to represent its pose with respect to the fixed one ('branch' or 'open loop'). For each PKM different alternative choices are possible. For example for the PKM of Fig. 1-right three alternatives are possible: one equation for each of the two branches $B1$ and $B2$, or one equation for loop $L1$ and one equation for one of the two branches ($B1$ or $B2$).

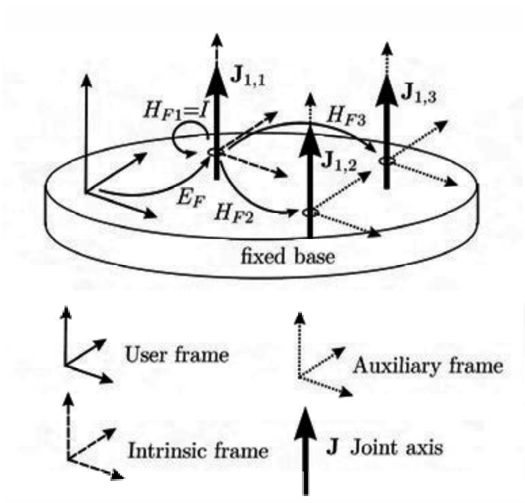


Fig. 2. Transformation matrices between user, intrinsic and joint auxiliary frames for the fixed base (example of a PKM with 3 legs). The auxiliary frame of the first leg coincides with the intrinsic frame and so $H_{F1} = I$. The matrices of the mobile base are similar.

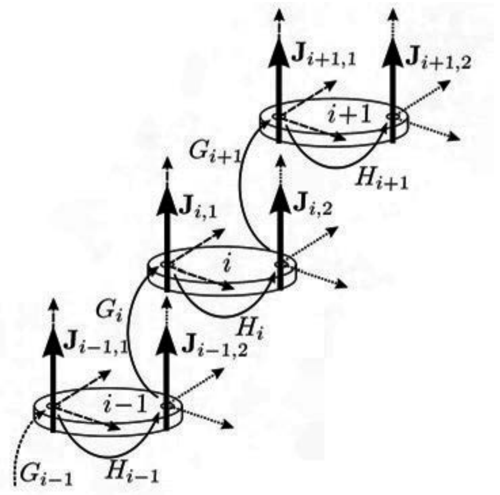


Fig. 3. Transformation matrices between the frames of three consecutive links with 2 joints. The pose of the joint frame $J_{i+1,2}$ with respect to $J_{i-1,1}$ is obtained as $M = H_{i-1}G_iH_iG_{i+1}H_{i+1}$.

The kinematic equations are obtained in matrix form by multiplying all the matrices that describe the relative position of the frames encountered by travelling along the considered branches or loops. For each branch we follow this path:

- Start from the absolute user frame and move to the intrinsic reference frame of the fixed base. This transformation is described by the matrix E_F
- Move to the joint frame of the first joint of the branch (matrix H_F).
- Move to the joint frame embedded on the next link (matrix G).
- Move to the last joint frame of the same link (matrix H).
- Repeat the two previous steps until the last link (mobile base).
- Move to the user frame of the mobile base (matrix E_M).

When a loop is considered only the joint frames must be considered:

- start from the last joint frame of an arbitrary link of the loop.
- move to the first joint frame of the next link (matrix G).
- move to the last joint frame of the same link (matrix H).
- repeat the two previous steps for all the joints of the kinematic loop.

The kinematics equations result as:

$$\begin{array}{ll} M_k = E_F H_{Fk} G_{1k} H_{1k} G_{2k} H_{2k} \ldots G_{nk} H_{Mk} E_M & \text{for the k-th branch } k=1,b \\ I = G_{1j} H_{1j} G_{2j} \ldots H_{mj} & \text{for the j-th loop } j = 1, \ell \\ & b + \ell = L + F - 1 \end{array} \quad (5)$$

where I denotes the identity matrix, M_k is the pose of the gripper at the end of the k -th branch, n is the number of the joints encountered on the branch and m is the number of the links composing the j -th loop. b and ℓ are the number of equations for branches and loops ($b \geq F-1$, $\ell \leq L$). Subscripts k and j will be often omitted for simplicity. Matrices E_F and E_M are identical for all the branches.

A complete set of rules to assign the location of the frames is given in the following Sections which also discuss the singular cases.

3.2 Definition of the intrinsic frames

As already mentioned, on each link it is necessary to fix an 'intrinsic frame'. The procedure is conceptually equivalent to that adopted for serial manipulators when frames are embedded on links with the D&H approach. The rules used for the positioning of the intrinsic frame avoid the use of redundant parameters. However, those rules must be generalized to consider also cylindrical and spherical joints. The necessary rules are summarized in Table 2 where the type of the link is identified by the group of its joints used to define the intrinsic frame. Table 2 also presents the intrinsic parameters of the links and the condition of singularity that may happen.

In links having several joints, different choices may sometimes be performed to define the intrinsic frame all resulting in a different but equivalent model. For example, in a link with two revolute joints (**R**) and a spherical one (**S**), it is possible to adopt the rules for **RR** -links or that for **RS** ones.

Some considerations must be done. **C** joints are not explicitly mentioned because they follow the rules given for the **R** ones. Moreover **PS** -joints (spherical joints that move on a straight line) also follows the rules of **R** joints; no parameters are defined in this case for the prismatic joint offset which is assumed to be null. Finally, if a link is composed by two **R** or **C** parallel joints only, the Hayati representation has to be adopted. When on one link more joints of the same type are present, they can be arbitrary numbered.

The geometry of some links does not allow a complete automatic definition of the intrinsic frame ('singular cases'). In these cases, some parameters that define the origin position and/or the attitude of the intrinsic frame can be partially freely assigned and they do not enter in the calibration model. All these singular cases are described in Table 3.

3.3 Definition of the auxiliary joint frames (H matrices)

For each joint of each link, an auxiliary joint frame must be defined following some rules dependent on the joint type:

- **R and C joints**: the axis z is chosen coincident to the joint axis. The axis x is chosen to lay on the common normal to the joint axis and one axis (generally z) of the intrinsic frame which is not parallel to it.
- **P joints**: the axis z is parallel to the joint axis. The axis x is defined as for revolute joints. Like the standard D&H links the origin of the frame can be freely translated so that in the assembly configuration it coincides with the origin of the frame associated to the first revolute or spherical joint which follows it in the kinematic chain.
- **S joints**: the origin of the frame coincides with the center of the joints and the frame axes are parallel to those of the intrinsic frame.

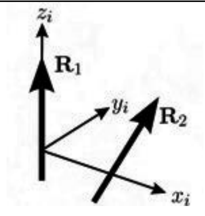
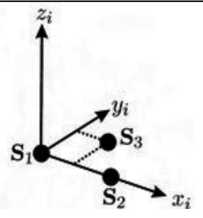
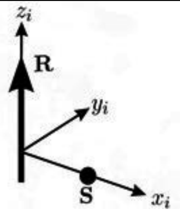
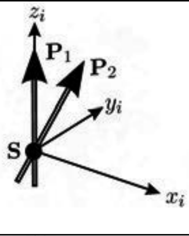
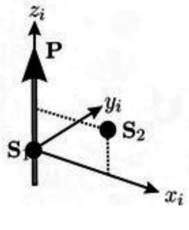
	**RR -links** **Rules:** the z_i axis coincident with the first joint axis, the x_i axis coincident with the common normal. **2 Intrinsic parameters:** the distance between the two joint axes and the angle among them. **Singular case:** when the axes of the two joints are parallel.
	**SSS -links** **Rules:** the origin of the frame in S_1, the axis x_i toward S_2, the axis y_i in such a way that the z coordinate of S_3 is null. **3 Intrinsic parameters:** the coordinate x of S_2 and the coordinates x and y of S_3. **Singular case:** when the three joints are aligned.
	**RS -links** **Rules:** The axis z_i coincident with the revolute joint axis, the axis x_i toward the spherical joint. **1 Intrinsic parameter:** The coordinate x of the spherical joint. **Singular case:** When the center of S is on the axis of R .
	**PPS -links** **Rules:** The origin of the frame in the spherical joint, the axis z_i parallel to P_1 joint axis, the axis x_i defined by the cross product of the two prismatic joint axes. **1 Intrinsic parameter:** The angle between P_1 and P_2. **Singular case:** When the two P axes are parallel to each other.
	**PSS -links** **Rules:** the origin of the frame in the spherical joint S_1, the axis z_i parallel to P axis, the axis x_i laying in the plane passing through z_i and S_2. **2 Intrinsic parameters:** the coordinates x and z of S_2. **Singular case:** when the axis of P is parallel to the segment connecting S_1 and S_2.

Table 2. Rules to position the intrinsic frame on a link.

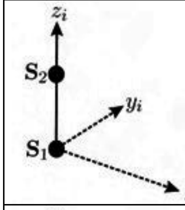
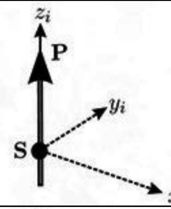
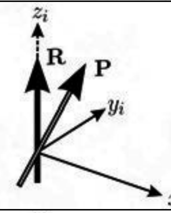
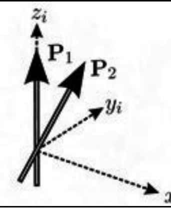
	**SS -links** **Rules:** the origin of the frame in S_1, the axis z toward S_2, the orientation of x and y axis arbitrary but orthogonal to the z axis. **1 Intrinsic parameter:** the coordinate z of S_2. **1 Free parameter:** the rotation around z axis.
	**PS -links** **Rules:** the origin of the frame in S, the axis z parallel to P , the orientation of x and y axis arbitrary but orthogonal to z axis. **None intrinsic parameter** **1 Free parameter:** the rotation around z axis.
	**RP -links** **Rules:** the axis z coincident to the R joint axis, the axis x in such a way that the P axis is parallel to the yz plane. **1 Intrinsic parameter:** the angle between the joint axes. **1 Free parameter:** the translation along z.
	**PP -links** **Rules:** the axis z parallel to the P_1 joint axis, the axis x in such a way that the P_2 axis is parallel to the yz plane ($x = P_2 \times P_1$). **1 Intrinsic parameter:** the angle between the joint axes. **3 Free parameters:** the translations along x , y and z .

Table 3. Rules to position the intrinsic frames on a link with free parameters (singular cases).

In all the mentioned cases a constant matrix H (the 'link matrix') can be built to represent the location of the auxiliary joint frame with respect to the intrinsic one. This matrix has generally the form of D&H-like standard matrices defined by two translations and two rotations. However for **S** joints three translations are required. As usual when a prismatic joint is present, its location can be arbitrary assigned and so two redundant parameters are inserted.
During the construction of the matrices H for the joints which have been involved in the definition of the intrinsic frame some parameters have necessarily a constant null value and so they are not inserted in the calibration parameter set Λ . For instance in a link with $n_s > 3$ spherical joints labelled $S_1, S_2, \cdots, S_{n_s}$ where the intrinsic frame is assigned with the rules of Table 2, the matrices describing the joint frames locations are:

$$H_1 = I \qquad H_2 = T(x, x_2) \qquad H_3 = T(x, x_3)T(y, y_3)$$

$$H_i = T(x, x_i)T(y, y_i)T(z, z_i) \qquad i = 4 \ldots n_s$$

where I is the identity matrix. So we need $3(n_s - 2)$ parameters to describe their relative position.
As a further example, in a link with 3 revolute joints $\mathbf{R_1}$, $\mathbf{R_2}$ and $\mathbf{R_3}$ and a spherical one (labelled 4), assuming that the intrinsic frame have been defined using the revolute joints $\mathbf{R_1}$ and $\mathbf{R_2}$, we get

$$H_1 = I \qquad H_2 = T(x,l_2)R(x,\varphi_2) \qquad H_3 = T(z,h_3)R(z,\theta_3)T(x,l_3)R(x,\varphi_3)$$
$$H_4 = T(x,x_4)T(y,y_4)T(z,z_4)$$

Note: for all the link types the frame of the first joint coincides with the intrinsic one and so $H_1 = I$.

3.4 Joint assembly and joint motions (G matrices)

If all the mentioned joint frames are defined with the given rules, **R, C** and **P** joints axes will coincide with the z axes of a frame (intrinsic or auxiliary). The relative assembly and motion of contiguous links can be then described by the 'joint matrices' G_i defined as

$$G_i = T(z,c_i)R(z,\gamma_i) \qquad \text{for } \mathbf{P}, \mathbf{R} \text{ and } \mathbf{C} \text{ joints}$$
$$G_i = R(x,y,z,q_i,q_{i+1},q_{i+2}) \qquad \text{for } \mathbf{S} \text{ joints}$$

where c_i and γ_i are joint coordinates or assembly condition in dependence of the joint type (Table. 10 in Section 9) while $R(x,y,z,q_i,q_{i+1},q_{i+2})$ is often represented by the product of three elementary rotations around orthogonal axes.

3.5 Parameters set for the ED&H approach

The complete set of parameters to be included in the MCPC calibration model is composed by increments of all the parameters describing the matrices E, G, H necessary to write the $L+F-1$ kinematics equations (Eq. (5)).
From the set of the parameters must be removed the offsets of the unsensed joints and the redundant parameters due to the **P** joints or the singular **SS** links. The elimination can be performed applying the methodology described for the serial robots in relative chapter and adapted to PKMs in Section 6.
When the external calibration is to be performed it is not necessary to define the intrinsic frame of the fixed and of the mobile base. If this alternative is chosen, a joint frame must be defined for each joint connected to the base and their position is assigned with respect to the user frame.

4. Modified Incremental Approach

When an external calibration is required, the identification of the parameters set to form the calibration model can be performed upgrading the modified incremental approach developed for the serial manipulators in the first part of this work.
It is possible to define on each link a number of frames equals to the number of its joints minus one ($n_j - 1$) and then representing the nominal kinematics of the PKM by a suitable number of matrix equations as already indicated by Eq. (5)

$$\begin{aligned} M_k &= A_{0k}A_{1k}A_{2k}\dots A_{nk} && \text{for each branch } k = 1,b \\ I &= A_{1j}A_{2j}\dots A_{mj} && \text{for each loop } j = 1,\ell \\ & && b+\ell = L+F-1 \end{aligned} \tag{6}$$

These matrix equations are equivalent to Eq. (5) with $A_{0k} = E_F H_{Fk}$, $A_{ik} = G_{ik}H_{ik}$ and $A_{nk} = G_{nk}H_{nk}E_M$. Eq.s (6) are then updated adding the matrices B_i describing errors in the joint location, matrices $E_{F'}$ and $E_{M'}$ describing the errors in the location of the frames of the fixed and of the mobile frames and the matrices D_i describing the offset in the joint coordinates. Assuming that the intrinsic frame is attached to the first joint of each link encountered travelling from the base to the gripper we get

$$\begin{aligned} &M = E_{F'} A_0 B_0 D_1 A_1 B_1 D_2 A_2 \ldots D_{n-1} A_{n-1} B_{n-1} D_n B_n A_n E_{M'} && \text{for each branch} \\ &I = D_1 A_1 B_1 D_2 A_2 B_2 \ldots D_m A_m B_m && \text{for each loop} \end{aligned} \tag{7}$$

where the subscripts k and j have been omitted for simplicity. If the intrinsic frame is attached to the second joint of the link or for *i*=*n* the product $A_i B_i$ commutes to $B_i A_i$. Matrices $E_{F'}$ and $E_{M'}$ are identical for all the branches.

Assuming that the frames are created in such a way that the axes of **R**, **P** and **C** joints coincide with z axes, the matrices B_i, E'_F and E'_M are as follows

$$\begin{aligned} &B_i = R(x,\Delta\alpha_i)R(y,\Delta\beta_i)T(x,\Delta a_i)T(y,\Delta b_i) && \text{if } J_{i+1}=\mathbf{R} \text{ or } \mathbf{C} \\ &B_i = R(x,\Delta a_i)R(y,\Delta b_i) && \text{if } J_{i+1}=\mathbf{P} \\ &B_i = T(x,\Delta a_i)T(y,\Delta b_i)T(z,\Delta c_i) && \text{if } J_{i+1}=\mathbf{S} \\ &E_{k'} = T(x,\Delta a_k)T(y,\Delta b_k)T(z,\Delta c_k)R(x,\Delta\alpha_k)R(y,\Delta\beta_k)R(z,\Delta\gamma_k) && \text{base frame } k=\text{F,M} \end{aligned}$$

and B_n has the same form but depends on J_n.

Matrices D_i assume the following form

$$\begin{aligned} &D_i = R(z,\Delta q_i) && \text{if } J_i=\text{R} \\ &D_i = T(z,\Delta q_i) && \text{if } J_i=\text{P} \\ &D_i = R(z,\Delta q_i)T(z,\Delta q_{i+1}) && \text{if } J_i=\text{C} \\ &D_i = R(x,\Delta q_i)R(y,\Delta q_{i+1})R(z,\Delta q_{i+2}) && \text{if } J_i=\text{S} \end{aligned}$$

Then the Eq.s (7) are analyzed in order to remove from matrices B_i the parameters which are redundant to the joint offsets of matrices D_i (Section 6). Finally all the parameters describing the joint offsets of the unsensed joints are also removed.

To reduce from the beginning the number of the redundant parameters to be removed, the matrices B_i can be defined taking in account some concepts developed in the $ED\&H$ methodology. On each link it is necessary to define the intrinsic and the joints frames, a matrix B_i is defined for each joint frame but some of its parameter can be immediately removed. The parameters removed are those whose value is implicitly fixed to zero by the definition of the intrinsic frame.

For instance in a link with $n_s > 3$ spherical joints $\mathbf{S}_i$, we get

$$B_1 = I \qquad B_2 = T(x,\Delta a_2) \qquad B_3 = T(x,\Delta a_3)T(y,\Delta b_3)$$
$$B_i = T(x,\Delta a_i)T(y,\Delta b_i)T(z,\Delta c_i) \qquad i = 4\ldots n_s$$

or in a link with 3 revolute joints (**R1**, **R2** and **R3**) and a spherical one (labelled 4), assuming that the intrinsic frame was defined using the revolute joints **R1** and **R2**, we get:

$$B_1{=}I \qquad B_2{=}T(x,\Delta a_2)R(x,\Delta\alpha_2) \qquad B_3{=}T(z,\Delta c_3)R(z,\Delta\gamma_3)T(x,\Delta a_3)R(x,\Delta\alpha_3)$$
$$B_4{=}T(x,\Delta a_4)T(y,\Delta b_4)T(z,\Delta c_4)$$

Note that for all the links we have $B_1 = I$ (see for analogies Section 3.3).

5. Singular Links

5.1 SS links

The Fig. 4 shows a **SS** link connecting link $i-1$ with link $i+1$. Adopting the modified Incremental approach, among the other parameters, it could be necessary to consider 3 joint offsets $\Delta q_{x,i-1}$, $\Delta q_{y,i-1}$ and $\Delta q_{z,i-1}$ in the joint $i-1$ and 3 translation parameters Δa_{i+1}, Δb_{i+1} and Δc_{i+1} in the joint $i+1$.

An infinitesimal analysis (Section 6) of the error propagation in the kinematic chain gives the following results. The error Δa_i is equivalent to $-\Delta q_{zi-1} l_i$ and Δc_i is equivalent to $\Delta q_{xi-1} l_i$; therefore, Δa_i and Δc_i must be ignored. The joint rotations $\Delta q_{x,i-1}$, $\Delta q_{y,i-1}$, and $\Delta q_{z,i-1}$, are to be discharged because the joint is unsensed. And so only Δb_i survives. The result is that **SS** -links require just one parameter (the link length); that is $SI = +1$.

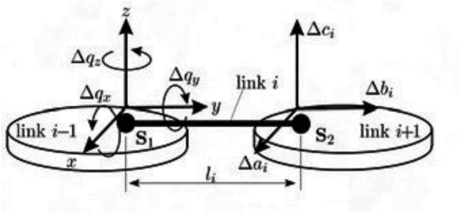


Fig. 4. Redundant parameters on a **SS** -link.

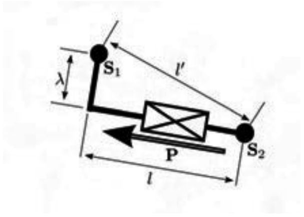


Fig. 5. An actuated **SPS** leg ($\lambda << l$).

5.2 SPS legs

Fig. 5 represents a simplified scheme of a **SPS** leg which is often employed in PKMs.

According to the general formula (Eq. (4)) such a structure would require two parameters to be described in fact the prismatic link is actuated ($P = 1$, $E = 1$). The two parameters are the link offset λ and the length l. The total link length (the distance between the two spheres) is $l' = \sqrt{\lambda^2 + l^2}$). However in general it is $\lambda << l$ and so $l' \cong l$. In these circumstances the parameters λ could be neglected, this is taken into account assigning the value $SI = -1$ to each **SPS** leg.

6. Elimination of the Redundant Parameters of PKMs Models

6.1 Analytical reduction

As explained in the chapter devoted to serial manipulators, the relation between the error ΔS of the gripper pose and the errors in the manipulator geometry Λ can be expressed by means of the jacobian J_Λ

$$\Delta S = [dx, dy, dz, d\alpha, d\beta, d\gamma]^T = J_\Lambda \, \Delta\Lambda \tag{8}$$

For PKM, the identification of the redundant parameters to be removed can be performed using the methodology developed for serial robots (see relative chapter). The only difference with the case of serial manipulators is that in general each parameter can be present in more than one of the $L + F - 1$ kinematic equations (Eq. (7)) and so the redundancy analysis must be performed analyzing an extended jacobian $\overline{J}_\Lambda$ formed by the $L+F-1$ jacobian $J_{\Lambda,k}$ ($k=1,L+F-1$) generated from each matrix equation (Eq. (5))

$$\overline{J}_\Lambda = \begin{bmatrix} \cdots \\ J_{\Lambda,k} \\ \cdots \end{bmatrix}$$

The matrix $\overline{J}_\Lambda$ has $6(L+F-1)$ rows. The i-th parameter is considered redundant to others if the i-th column of $\overline{J}_\Lambda$ can be expressed as a linear combination of those of the quoted parameters.

6.2 Numerical reduction of the parameters set

The algorithms described in the previous sections lead to a theoretically minimum and complete parametrically continuous set of parameters. For some PKMs, the number of parameters can be very high. However, as already mentioned in the paper, with particular dimensions of the links or when calibration is performed in a limited portion of the working area, some of the parameters may have a very limited effect on the manipulator motion and also considering the measuring errors their value cannot be reliably identified by calibration. We will call them 'numerically unobservable' parameters. In other cases two or more parameters may have nearly the same numerical effect on the collected data. In these cases it is not possible to separate their effects and estimate the correct value of all of them. In practice one (or more) of these parameters is considered 'numerically redundant'. A remarkable example is the link offset λ of **SPS** legs (Section 5.2).

In general, it is advisable to remove numerically unobservable and redundant parameters from the model to improve the calibration effectiveness because their presence degrades the identification of the numerical value of the parameters especially in presence of measuring noise or in case of not modelled sources of error.

A deep analysis of this topics is outside the scope of this paper but it is worth mentioning how this subject can be addressed. The basic idea is that, as already mentioned, one parameter is redundant if its column of the jacobian matrix can be expressed as linear combination of others. The methodology proposed in Section 6.1 based on an analytical analysis of the jacobian guarantees the elimination of all the redundant parameters. However, in the case of 'numerically unobservable' parameters, some columns are theoretically independent to each other but their numerical values is very close to this condition. In this case the elimination algorithm must be based on a numerical analysis of the jacobian and of Eq. (8) evaluated for all the measured poses. One possible approach has been presented in (Tosi & Legnani, 2003). The methodology is based on the analysis of the singular values of the jacobian which assume the meaning of amplification factors of the geometrical errors. A singular value equal to zero means that there is a certain linear combination of parameters which has no effect on the pose of the robot. A singular value whose numerical value is different from zero but lower than a prescribed threshold indicates that there is a parameter (or a group of parameters) that is 'nearly' redundant and whose effect is numerically indistinguishable from the others.

Indicating with S the position of the mobile base, with Q the joint positions and with Λ the geometrical parameters we can write

$$S = F(Q,\Lambda) \qquad \Delta S_i \cong \frac{\partial F}{\partial \Lambda}\Delta\Lambda = J_{Si}\Delta\Lambda$$

where F is the direct kinematics and J_{Si} is the jacobian evaluated in the *i*-th configuration. The different jacobians evaluated for all the considered m poses as well as the variations in the mobile base pose coordinates can be merged in one global equation

$$\Delta S_{tot}=\begin{bmatrix}\Delta S_1\\ \Delta S_2\\ \vdots\\ \Delta S_m\end{bmatrix}=\begin{bmatrix}J_{S1}\\ J_{S2}\\ \vdots\\ J_{Sm}\end{bmatrix}\Delta\Lambda=J_{Stot}\Delta\Lambda \tag{9}$$

The global jacobian can be splitted in three matrices by Singular Value Decomposition (Press et al. 1993) $J_{Stot}=US_sV$ converting Eq. (9) in

$$\Delta S^*_{tot}=S_S\Delta\Lambda^*$$

with

$$\Delta S^*_{tot} = U^T\Delta S_{tot} \qquad \Delta\Lambda^* = V^T\Delta\Lambda \qquad \Delta\Lambda^* = \begin{bmatrix}\Delta\lambda_1^*\Delta\lambda_2^*\ldots\Delta\lambda_N^*\end{bmatrix}^T$$

where $\Delta\Lambda^*$ represents the combined parameters, ΔS^*_{tot} represents the pose error in the considered configurations and the diagonal matrix S_s contains the singular values of J_{Stot} that can be interpreted as amplification coefficients. If one singular value is lower than a suitable threshold, the corresponding combined parameter can be neglected. The threshold $\overline{\sigma}_S$ depends on the sensors precision, the manufacturing tolerance and the required final precision. We can define two diagonal weight matrices P_S containing the maximum acceptable error for the i -th coordinate of the pose and P_Λ containing the maximum expected parameter error for the parameter λ_i.
These matrices can be used to normalize the vectors ΔS_{tot}, and $\Delta\Lambda$ as

$$\begin{aligned}\Delta\overline{S}_{tot} &= P_S^{-1}\Delta S_{tot}\\ \Delta\overline{\Lambda} &= P_\Lambda^{-1}\Delta\Lambda\end{aligned} \tag{10}$$

$\Delta\overline{\Lambda}$ is called 'scaled' parameters vector. Merging Eq.s (10) and Eq.(9) it can be found that the proposed procedure based on SVD must be applied to the normalized matrices

$$\begin{aligned}\overline{J}_{Stot} &= P_S^{-1}J_{Stot}P_\Lambda\\ \Delta\overline{S}_{tot} &= \overline{J}_{Stot}\Delta\overline{\Lambda}\end{aligned} \tag{11}$$

In this case, a proper threshold values $\overline{\sigma}_S$ for the singular values depends only on the dimension of vector $\Delta\overline{\Lambda}$. Indeed, the maximum expected parameter error $\Delta\overline{\Lambda}$ may cause an error on a coordinate s_i of the platform smaller than the acceptable one. We can write

$$\overline{\sigma}_S=\frac{\|\Delta\overline{S}_{tot}\|_\infty}{\|\Delta\overline{\Lambda}\|_2}$$

where, for a generic vector A, the norms are defined as $\|A\|_\infty=\max(|a_i|)$ and $\|A\|_2=\sqrt{\sum a_i^2}$.
From the definition of matrix P_Λ if follows that $|\Delta\overline{\lambda}_i|\le 1$ then using the euclidean norm

$$\| \Delta\overline{\Lambda} \|_2 \leq \sqrt{N}$$

where N is the number of the parameters. From the definition of matrix P_S we have

$$\| \Delta\overline{S}_{tot} \|_\infty = 1$$

then, a proper threshold value is

$$\overline{\sigma}_S = \frac{1}{\sqrt{N}}$$

	Reference	R	P	C	SI	E	L	F	N
		3	1	2	±1	1	6	6	
Stewart-Gough 6[**SPS**]	(Jokiel et al., 2000)	0	6	0	0	6	5	2	48 (42♣)
Stewart-Gough 6[**URPU**]	(Wang et al., 1993)	30	6	0	0	6	5	2	138
3[**CCR**]	(Callegari & Tarantini, 2003)	3	0	6	0	3	2	2	40
3[(**RP**)(**RP**) **R**]	Section 7.4	9	6	0	0	3	2	2	46
6[**PSS**]	(Ryu & Rauf, 2001)	0	6	0	6	6	5	2	54
Cheope 3[**P** (2 **S** /2 **S**)]	(Tosi & Legnani, 2003)	0	3	0	6	3	5	2	48
Hexa 6[**RSS**]	Fig. 10	6	0	0	6	6	5	2	66
Delta4 3[**R** (2 **S** /2 **S**)]	(Clavel, 1991)	3	0	0	6	3	5	2	54

Table 4. Number of parameters for external calibration (F=2) of some serial or parallel manipulators (Eq. (4)). For Internal calibration subtract 12 parameters (F=0). For extra sensors add the suitable number of parameters. ♣ generally one parameter for each of the 6 **SPS** leg can be numerically neglected.

7. Examples: Analysis of Some PKMs

This Section analysis some PKMs to clarify the presented methodology for the automatic identification of the parameter set. The discussion refers to Table 4. The manipulators are identified by a code containing the number of legs and the sequence of their joints.

7.1 External calibration of the Stewart-Gough platform with SPS legs

The parameters set of a standard Stewart-Gough platform with spherical joints (Fig. 6) can be identified as follows.

- *Fixed base:* an 'user frame' is created on the fixed base requiring 6 parameters. Then the intrinsic frame is defined using the rules given in Table 2 for **SSS** -links. The joint frames are created on the spherical joints requiring 0+1+2+3+3+3=12 parameters (Section 3.3). We get a total of 6+12=18 parameters.
- *Mobile base:* as for the fixed base we get 18 parameters.
- *Legs:* each leg is composed by two spherical joints connected by a prismatic joint. Each leg then requires $P+E=2$ parameters.
- *Total number of parameters:* N = 2*18+6*2 = 48. As a check using Eq. (4) we get $R=0$, $P=6$, $C=0$, $SI=0$, $E=6$, $L=5$, $F=2$ which confirms $N=48$.

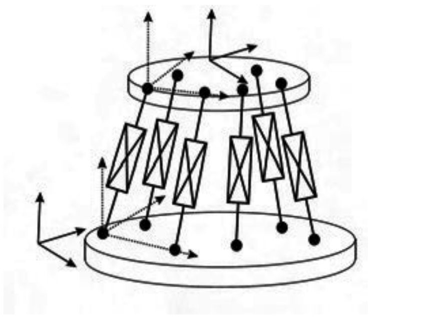
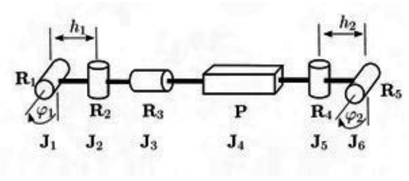


Fig. 6 Stewart-Gough platform. The 'intrinsic frames' defined on the joints are dashed, the 'user frames' are continuous.



h_1=h_2=0 (nominal values)

Fig. 7 A six-joints URPU leg.

As a control, in this simple case, the same results can be obtained observing that we need 3 coordinates for each spherical joints (their coordinates on the bases) plus the prismatic joint offsets and inclination errors (λ in Fig. 5).
In the common case in which $\lambda << l$, λ can be neglected reducing the number of the parameters by 6 ($SI = -6$). *Total number of parameters ignoring* λ *is then:* $N = 42$. This parameter set has been adopted during the external calibration of several PKMs (e.g., J kiel et al., 2000).

7.2 External calibration of Stewart-Gough platform with URPU legs

Stewart-Gough platforms are usually realized using universal joints in spite of spherical ones, due to their limited performances. Generally, the joints of one base are realized with universal joints plus rotational joints, while the others with bare universal joints (Fig. 7). We assume that also in this case we are interested in representing the mobile base motion with respect to the fixed one in the more general case and so the user base frames can be freely assigned on them.
On each of the 6 legs there are 5 not actuated revolute joints and one actuated prismatic joint. Using Eq. (4) it is $R = 30$, $P = 6$, $C = 0$, $SI = 0$, $E = 6$, $L = 5$, $F = 2$ and we get $N = 138$.
Calibrating a so large number of parameters can be a problem but some of them can be neglected because **URPU** legs are usually realized with intersecting and orthogonal joints axes; referring to Fig. 7, their nominal value is $h_1 = h_2 = 0$, (Wang & Masory, 1993), (Wang et al., 1993). In this case, little errors in some of the structural parameters do not affect significantly the mobile base pose (e.g. the relative orientation between joints $J_1 - J_2$ or $J_5 - J_6$) . As a further example, if the joint inclination changes just a little during PKM operation, errors on h_i are un-distinguishable from the offset error of the **P** joint coordinate. These parameters are sometimes neglected during external calibration and the **URPU** leg is modelled as a **SPS** leg (Patel & Ehmann, 1997). This is not possible when indirect calibration is performed using extra-sensors measuring the leg rotations because, for example, the value of the joint coordinate of J_2 is affected by its orientation with respect to J_1 .

7.3 Stewart-Gough platform calibrated by a double ball bar

Consider the **SPS** platform of Section 7 and assume that calibration is performed by a double ball bar (DBB) (Ryu & Rauf, 2001). No user frames are defined on the fixed and mobile bases. The number of the parameters is then decreased by 12. Since errors can be also present in the DBB or in its connection to the bases, 8 further parameters are needed (a DBB is cinematically equivalent to an actuated **SPS** leg). We get: *Total number of parameters:* N = 48-12+8=44.

If the parameter set is composed by the coordinates of the center of the spherical joints and by 2 parameters for each leg (joint offset and λ), this would give 56 parameters. To achieve minimality, the number of parameters can be reduced to 44 with a proper choice of the reference frames on the two bases: the origin is located on one sphere $\mathbf{S}_1$, being the x axis directed toward a second sphere $\mathbf{S}_2$, and y axis chosen in such a way that a third sphere $\mathbf{S}_3$ lies in the $x-y$ plane. As a results, the following coordinates error must be neglected for each base: Δx, Δy and Δz for $\mathbf{S}_1$, Δy and Δz for $\mathbf{S}_2$ and Δz for $\mathbf{S}_3$.

7.4 3[CCR] PKM

Fig. 8 represents a 3-dof PKM (Callegari & Tarantini, 2003) witch has three legs with two cylindrical and one revolute joint. If the axes of the cylindrical joints of each leg is orthogonal to the axes of the joints connecting the leg to the fixed and mobile bases, then the motion of the mobile platform is a pure translation. This PKM can be actuated by the prismatic joint located in the legs or those located in the base. We get a total of 40 parameters (Table 2).
The cylindrical pairs can be realized with true **C** pairs or by a combination of a **R** and a **P** joints; in this case it is not possible to assure that their joint axes are exactly parallel and so the number of parameters increases to 46 (see manipulator 3[(**RP**)(**RP**) **R**] in Table 4).

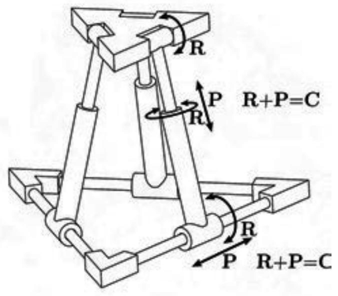


Fig. 8 A 3 DOF PKM with **CCR.**

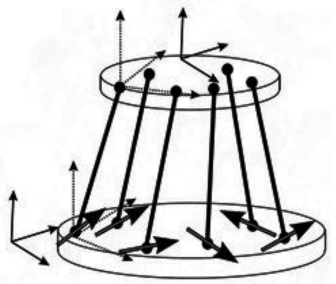


Fig. 9. A PKM with **PSS** legs.

7.5 6-dof PKM with PSS legs

The analyzed PKM is described in (Ryu & Rauf, 2001). It is similar to the Stewart-Gough platform but the legs have constant length (Figure 9). The mobile base is actuated by moving the ends of the legs on prismatic joints of the fixed base.
For the external calibration 5 parameters are necessary to define the position of the moving end of each leg (4 to define the joint axis, 1 to define the position of the spherical joint on the line), 1 parameter is necessary for each leg (its length), 3 parameters are necessary for each spherical joint on the mobile base. *Total number of parameters:* N =6*(5+1+3)=54. This number is confirmed by eq. Eq. (4) and Table 4.
If indirect calibration is performed by a DBB, in analogy with what discussed for the Stewart-Gough platform, we get: *Total number of parameters:* N =54-12+8=50.
If we have only 3 slides each one moving 2 spherical joints we get the Cheope manipulator (Tosi & Legnani, 2003) which has 48 parameters (Section 7.8).

7.6 PKM with RSS legs (Hexa manipulator)

Equations: Let us consider a generic 6[**RSS**] PKM with rotational actuated joints. The i -th leg is shown in Fig. 10-left. According to Eq. (4) and to Table 4, the model requires 66 parameters. Adopting an ED&H model, the complete transformation matrix M , describing the position and the orientation of the gripper with respect to the reference frame, comprehensive of all possible geometrical and joint offset error parameters of the i -th leg, is (Eq. (5), Fig. 10-right):

$$M = E_F \cdot H_{Fi} \cdot G_{1i} H_{1i} \cdot G_{2i} H_{2i} \cdot G_{3i} H_{Mi} \cdot E_M$$

while for the modified incremental approach we need for each leg

$$M = E_F \cdot A_{0i} B_{0i} \cdot D_{1i} A_{1i} B_{1i} \cdot D_{2i} A_{2i} B_{2i} \cdot B_{3i} A_{3i} \cdot E_M$$

If internal calibration is performed, some error parameters, modelled by matrices B and D , can be neglected, and $E_F = E_M = I$. For a comparison between the available alternatives see Table 5, Table 6, Table 7 and Table 8.

Internal calibration: An intrinsic frame is defined on the fixed base considering the revolute joints of leg 1 and leg 2 and on the mobile base considering the spherical joints of legs 1, 2 and 3.

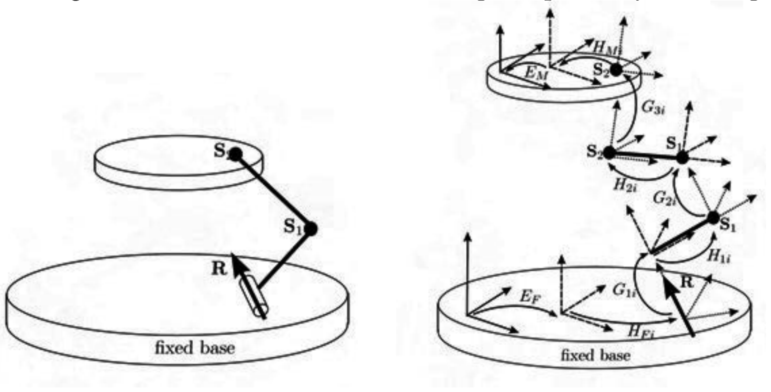


Fig. 10. Reference frames and transformation matrices for nominal kinematics on i -th leg of a **RSS** PKM. The actuator is on the R joint.

Internal Parameters					
Leg	**Fixed Base**	**R** Joint	**RS** Link	**RS** Link	**SS**
i	H_{Fi}	G_{1i}	H_{1i}	H_{2i}	H_{Mi}
1	-	$\Delta c, \Delta q$	Δa	Δa	-
2	$\Delta l, \Delta\varphi$	$\Delta c, \Delta q$	Δa	Δa	Δa
3	$\Delta h, \Delta\theta, \Delta l, \Delta\varphi$	$\Delta c, \Delta q$	Δa	Δa	$\Delta a, \Delta b$
4	$\Delta h, \Delta\theta, \Delta l, \Delta\varphi$	$\Delta c, \Delta q$	Δa	Δa	$\Delta a, \Delta b, \Delta c$
5	$\Delta h, \Delta\theta, \Delta l, \Delta\varphi$	$\Delta c, \Delta q$	Δa	Δa	$\Delta a, \Delta b, \Delta c$
6	$\Delta h, \Delta\theta, \Delta l, \Delta\varphi$	$\Delta c, \Delta q$	Δa	Δa	$\Delta a, \Delta b, \Delta c$
External Parameters					
Fixed Base		E_F	$\Delta a, \Delta b, \Delta c, \Delta\alpha, \Delta\beta, \Delta\gamma$		
Mobile Base		E_M	$\Delta a, \Delta b, \Delta c, \Delta\alpha, \Delta\beta, \Delta\gamma$		

Table 5. The 66 parameters (54 internal, 12 external) of the **RSS** PKM (ED&H Model).

Internal and External Parameters					
Leg	Fixed Base	**R** Joint	**RS** Link	**RS** Link	**SS**
i	H_{Fi}	G_{1i}	H_{1i}	H_{2i}	H_{Mi}
1	$\Delta h, \Delta\theta, \Delta l, \Delta\varphi$	$\Delta c, \Delta q$	Δa	Δa	$\Delta a, \Delta b, \Delta c$
2	$\Delta h, \Delta\theta, \Delta l, \Delta\varphi$	$\Delta c, \Delta q$	Δa	Δa	$\Delta a, \Delta b, \Delta c$
3	$\Delta h, \Delta\theta, \Delta l, \Delta\varphi$	$\Delta c, \Delta q$	Δa	Δa	$\Delta a, \Delta b, \Delta c$
4	$\Delta h, \Delta\theta, \Delta l, \Delta\varphi$	$\Delta c, \Delta q$	Δa	Δa	$\Delta a, \Delta b, \Delta c$
5	$\Delta h, \Delta\theta, \Delta l, \Delta\varphi$	$\Delta c, \Delta q$	Δa	Δa	$\Delta a, \Delta b, \Delta c$
6	$\Delta h, \Delta\theta, \Delta l, \Delta\varphi$	$\Delta c, \Delta q$	Δa	Δa	$\Delta a, \Delta b, \Delta c$

Table 6. The 66 parameters of the **RSS** PKM (ED&H Model, no intrinsic frames on the fixed and mobile bases).

Internal Parameters					
Leg	**Fixed Base**	**R** Joint	**RS** Link	**RS** Link	**SS**
i	B_{0i}	D_{1i}	B_{1i}	B_{2i}	B_{3i}
1	-	Δq	$\Delta c, \Delta a$	Δa	-
2	$\Delta a, \Delta\alpha$	Δq	$\Delta c, \Delta a$	Δa	Δa
3	$\Delta a, \Delta b, \Delta\alpha, \Delta\beta$	Δq	$\Delta c, \Delta a$	Δa	$\Delta a, \Delta b$
4	$\Delta a, \Delta b, \Delta\alpha, \Delta\beta$	Δq	$\Delta c, \Delta a$	Δa	$\Delta a, \Delta b, \Delta c$
5	$\Delta a, \Delta b, \Delta\alpha, \Delta\beta$	Δq	$\Delta c, \Delta a$	Δa	$\Delta a, \Delta b, \Delta c$
6	$\Delta a, \Delta b, \Delta\alpha, \Delta\beta$	Δq	$\Delta c, \Delta a$	Δa	$\Delta a, \Delta b, \Delta c$
External Parameters					
Fixed Base		E_F	$\Delta a, \Delta b, \Delta c, \Delta\alpha, \Delta\beta, \Delta\gamma$		
Mobile Base		E_M	$\Delta a, \Delta b, \Delta c, \Delta\alpha, \Delta\beta, \Delta\gamma$		

Table 7. The 54 internal and the 12 external parameters of the **RSS** PKM (Modified Incremental Model).

Internal and External Parameters					
Leg	**Fixed Base**	**R** Joint	**RS** Link	**RS** Link	**SS**
i	B_{0i}	D_{1i}	B_{1i}	B_{2i}	B_{3i}
1	$\Delta a, \Delta b, \Delta\alpha, \Delta\beta$	Δq	$\Delta c, \Delta a$	Δa	$\Delta a, \Delta b, \Delta c$
2	$\Delta a, \Delta b, \Delta\alpha, \Delta\beta$	Δq	$\Delta c, \Delta a$	Δa	$\Delta a, \Delta b, \Delta c$
3	$\Delta a, \Delta b, \Delta\alpha, \Delta\beta$	Δq	$\Delta c, \Delta a$	Δa	$\Delta a, \Delta b, \Delta c$
4	$\Delta a, \Delta b, \Delta\alpha, \Delta\beta$	Δq	$\Delta c, \Delta a$	Δa	$\Delta a, \Delta b, \Delta c$
5	$\Delta a, \Delta b, \Delta\alpha, \Delta\beta$	Δq	$\Delta c, \Delta a$	Δa	$\Delta a, \Delta b, \Delta c$
6	$\Delta a, \Delta b, \Delta\alpha, \Delta\beta$	Δq	$\Delta c, \Delta a$	Δa	$\Delta a, \Delta b, \Delta c$

Table 8. The 66 parameters of the **RSS** PKM (ED&H Model, no intrinsic frames on the fixed and mobile bases).

On the **RS** links an intrinsic frame is constructed with the rules given in Section 3.2. The parameter error Δa is the link length. Each revolute joint requires the parameter Δc (translation along the joint axis) and the joint coordinate offset Δq . On the **SS** link only the link length error Δa is to be considered because the spherical joints are not actuated and they do not require any parameter.

External calibration: For external calibration, 12 more parameters are added to describe matrices E_F and E_M containing the errors in the pose of the user frames with respect to the intrinsic ones (compare Table 5 and Table 7).
As a second alternative, we can omit to define the intrinsic frames on the two bases and errors of the joints can be defined with respect to the user frame ($E_F = E_M = I$, Table 6 and Table 8).

7.7 Delta 4 PKM

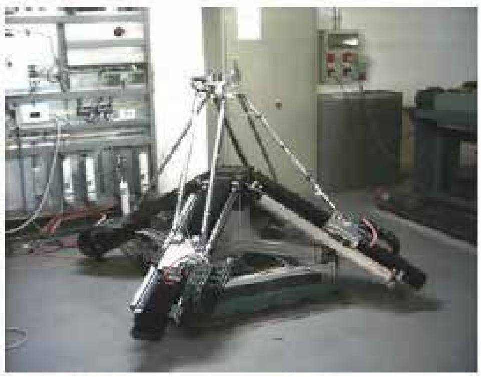
The delta robot (Clavel, 1991) can be obtained by merging two-by-two the contiguous **RS** links of the 6[**RSS**] PKM described in Section 7.6. In this case we get $R = 3$, $P = 0$, $SI = 6$, $L = 5$, $E = 3$, $C = 0$, $SI = -6$ giving a total number of parameters $N = 3*3+0+0+6+3+5\cdot 5+6\cdot(2-1) = 54$ for external calibration (Table 9).

Internal and External Parameters					
Leg	**Fixed Base**	**R** Joint	**RSS** Link	**SS** Link	**Mobile Base**
i	H_{Fi}	G_{1i}	H_{1i}	H_{2i}	H_{Mi}
1	$\Delta h, \Delta\theta, \Delta l, \Delta\varphi$	$\Delta c, \Delta q$	Δa	Δa	$\Delta a, \Delta b, \Delta c$
2			$\Delta a, \Delta b, \Delta c$	Δa	$\Delta a, \Delta b, \Delta c$
3	$\Delta h, \Delta\theta, \Delta l, \Delta\varphi$	$\Delta c, \Delta q$	Δa	Δa	$\Delta a, \Delta b, \Delta c$
4			$\Delta a, \Delta b, \Delta c$	Δa	$\Delta a, \Delta b, \Delta c$
5	$\Delta h, \Delta\theta, \Delta l, \Delta\varphi$	$\Delta c, \Delta q$	Δa	Δa	$\Delta a, \Delta b, \Delta c$
6			$\Delta a, \Delta b, \Delta c$	Δa	$\Delta a, \Delta b, \Delta c$

Table 9. The 54 parameters of the Delta4 PKM (ED&H Model, no intrinsic frames on the fixed and on the mobile bases).

7.8 Calibration of the Cheope manipulator

The Cheope manipulator (Fig. 11) has three slides. Each one moves two spherical joints on a linear track. Six spherical joints are located on the mobile platform and 6 rods connect the spherical joints of the platform with those of the slides. The PKM has three translational DOF. Other configurations, not described in this paper, may use the forth slide and an optional seventh rod (Tosi & Legnani, 2003).
As already mentioned in Table 4 Cheope requires 48 parameters for external calibration and 36 for internal. In practice, as mentioned in Section 6.2, many of these parameters may have a limited influence on the manipulator motion and so they can be eliminated from the calibration process. This is quite useful especially in presence of noise in the measures.
It is important to note that in many circumstances it is more important to consider the combined effects of some parameters rather than their individual effects. For example in Cheope, the average (and the difference) of the length error of each pair of rods are more significant than the individual length errors. The calibration will then be performed using suitable linear combinations of the parameters rather that using the individual parameters.
The extraction of the reduced set of parameters to be considered as well as the generation of the combined parameters can be automatically performed on the bases of the procedure explained in Section 6.2.



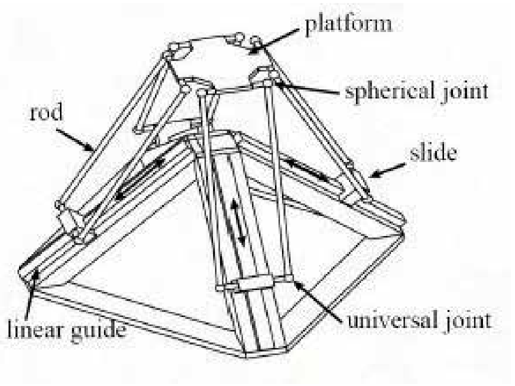
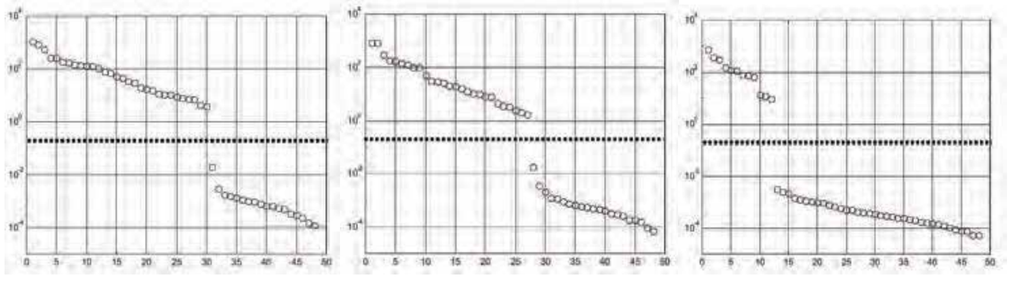


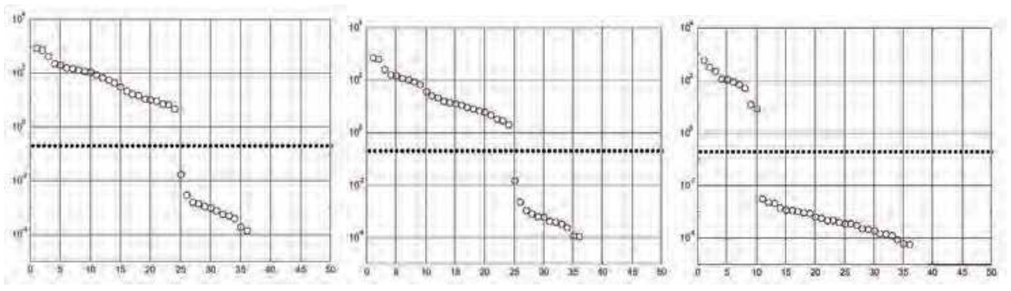
Fig. 11. Photo of the hybrid robot 'CHEOPE' and scheme of the parallel kinematic part.

Figure 12 and Figure 13 show the singular values of Cheope respectively for external and internal calibration. For each case three different situations are investigated: a) when both position and rotational errors of the mobile base are important, b) when only position errors are of interest, c) when only rotational errors are considered.



a) $S = [x_p \ y_p \ z_p \ \theta \ \varphi \ \psi]^T$ b) $S = [x_p \ y_p \ z_p]^T$ c) $S = [\theta \ \varphi \ \psi]^T$

Fig. 12. Singular values of matrix $\overline{J}_{Stot}$ in the case of external calibration: (a) complete pose, (b) position only, (c) rotation only; $\overline{\sigma}_S = 1/\sqrt{48} = 0.14$



a) $S = [x_p \ y_p \ z_p \ \theta \ \varphi \ \psi]^T$ b) $S = [x_p \ y_p \ z_p]^T$ c) $S = [\theta \ \varphi \ \psi]^T$

Fig. 13. Singular values of matrix $\overline{J}_{Stot}$ in the case of internal-calibration: (a) complete pose, (b) position only, (c) rotation only; $\overline{\sigma}_S = 1/\sqrt{36} = 0.17$.

The dashed line represent the threshold $\overline{\sigma}_S$, only the parameters associated to the higher singular values must be considered. For example, when both linear and angular errors are of

concern, the external calibration requires 30 combined parameters (Figure 12-a) while the internal one can be performed with 24 combined parameters only (Figure 13-a).
The thresholds have been evaluated considering manufacturing errors and joint sensor offsets with a maximum magnitude of 10^{-5} m, required PKM accuracy of 10^{-4} m, measuring error of 10^{-5} m for linear sensors and 10^{-5} rad for the rotational ones.

8. Conclusions

The paper has presented a complete methodology for the identification of the geometrical parameter set necessary to describe inaccuracy in the kinematic structure of a generic parallel manipulator.
The methodology, that can be applied to any existing PKM, supplies a minimum, complete and parametrically continuous model of the manipulators. It can be applied both to intrinsic (internal) and extrinsic (external) calibration.
The methodology identifies the parameters describing geometric errors of the manipulator, joint offset errors and errors in the external devices used for calibration (e.g. double ball bar). Furthermore, a general formula to determine the total number of necessary parameters has been presented and discussed.
The paper shows how for some manipulators the number of parameters that are theoretically necessary is quite large and a numerical methodology to remove the less significant is mentioned.
The final methodology is general and it is an *algorithm*, this means that it can be automatically applied to any given PKM.
Numerous practical explicative examples are also given.

9. Appendix. D&H Notation: the Multiple Frames Extension

In the first part of this work it has been mentioned how the four D&H parameters of one link represent its geometry, the assembly condition with the previous link and the relative motion with respect to it (joint coordinate). In some situations it is advisable to separate these factors (Sheth & Uicker, 1971) defining a notation with a constant part representing the rigid link and its assembly and a distinct variable part describing (the joint) motion. This is particularly necessary in the calibration of PKMs where there are links with more than two joints. Standard D&H notation for a link with n_j joints requires the definition of $n_j - 1$ frames attached to it and the so $n_j - 1$ sets of parameters are generated. As an example the case of a ternary link is reported in Fig. 14.
In the quoted example, the frame i is embedded on link A, while frames j and k are placed on link B. The link assembly and the joint motion between the links A - B are represented by the link offsets h_j and h_k plus the rotations θ_j and θ_k. The shape of the link is described by the two lengths l_j, and l_k plus the two twists φ_j and φ_k and by the difference between the two rotations $\theta_B = \theta_k - \theta_j$ and of the two link offsets $h_B = h_k - h_j$. To separate the constant and the variable parts Sheth and Uicker propose the adoption of a further frame to be attached on the link B in a free location with the only constrain to have its z axis coincident with the joint between the links A - B. In this paper we give rules to assign the location of this frame in a convenient way for calibration purposes (Section 3.2).

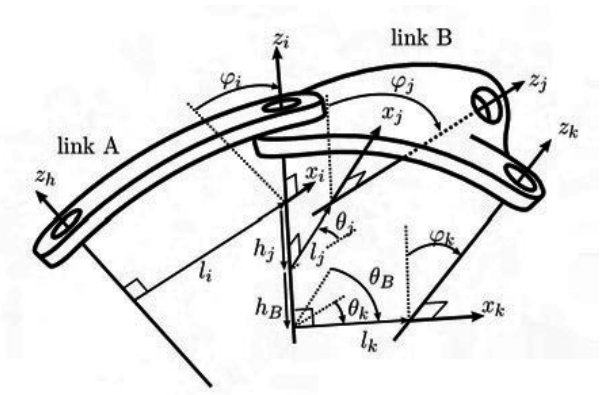


Fig. 14 The Denavit and Hartenberg parameters for a ternary link.

In the case of **R** , **P** and **C** joints, the transformation between the frames i - j and i - k can then be represented as

$$\begin{cases} M_{ij} = G_j H_{Bj} = \left(R(z,\theta_j)T(z,h_j)\right)T(x,l_j)R(x,\varphi_j) \\ M_{ik} = G_j H_{Bk} = \left(R(z,\theta_j)T(z,h_j)\right)R(z,\theta_B)T(z,h_B)T(x,l_k)R(x,\varphi_k) \end{cases} \tag{12}$$

The term $G_j = R(z,\theta_j)T(z,h_j)$ is common to both equations and is called 'joint transformation'. The product of the other terms is called 'link transformation' and will be indicated by H . Each joint has one joint transformation and each link has $n_j - 1$ link transformations. If we define on the link B a frame labelled (B) having its axis z coincident with the joint between the links A - B and its x axis coincident with axis x_j , we can interpret the common term G of Eq. (12) as the transformation between frames i - B and the others (H) as description of the link shape. For simplicity, the frame B is not shown in the figure. The exact interpretation of θ_j and h_j depends on the joint type as shown in Table. 10.

J Type	θ_j	h_j
R	Joint Variable	Assembly
P	Assembly	Joint Variable
C	Joint Variable	Joint Variable

Table. 10 Meaning of coordinates θ_j and h_j in the joint matrix G_j of Eq. (12).

In this paper, the frame B is called *intrinsic* frame of the link while j and k are (auxiliary) *joint* frames. Depending on the number and on the type of the joints of the links, the rules to assign the intrinsic frames must be adapted for calibration purposes in order to avoid singularities and redundancy (Sections 3.2 and 3.3).

10. Acknowledgment

This research work has been partially supported by MIUR grants cofin/prin (Ministero Università e Ricerca, Italia).

11. References

Besnard, S. & Khalil, W. (2001). Identifiable parameters for Parallel Robots kinematic calibration, *ICRA 2001*, Seoul.

Callegari, M. & Tarantini, M. (2003). Kinematic Analysis of a Novel Translation Platform, *Journal of Mechanical Design, Trans. of ASME,* Vol. 125, pp. 308-315, June 2003.

Clavel, R. (1991) Conception d'un robot parallè le rapide à 4 degrés de liberté, *Ph.D. Thesis*, EPFL, Lausanne, Switzerland, 1991.

Denavit, J. & Hartenberg, R. S. (1955). *Transactions of ASME Journal of Applyed Mechanics*, Vol. 22, pp. 215 - 221, June 1955.

Everett, L. J.; Driels, M. & Mooring, B.W. (1987). Kinematic Modelling for Robot Calibration, *Procedeengs of IEEE International Conferenc on Robotics and Automation*, Raleig, NC, Vol. 1, pp. 183-189.

Hayati, S. & Mirmirani, M. (1985). Improving the Absolute Positioning Accuracy of robot manipulators, *Journal of Robotic Systems*, Vol. 2, No. 4, pp. 397-413.

Jokiel, B.; Bieg, L. & Ziegert, J. (2000). Stewart Platform Calibration Using Sequential Determination of Kinematic Parameters, *Proceedings of 2000-PKM-IC*, pp. 50-56, Ann Arbor, MI, USA.

Khalil, W.; Gautier, M. & Enguehard, C. (1991). Identifiable parameters and optimum configurations for robot calibration, *Robotica*, Vol. 9, pp. 63-70.

Khalil, W. & Gautier, M. (1991). Calculation of the identifiable parameters for robot calibration, *IFAC 91*, pp. 888-892.

Legnani, G.; Casolo, F.; Righettini P. & Zappa, B. (1996). A Homogeneous Matrix Approach to 3D Kinematics and Dynamics. Part 1: theory, Part 2: applications, *Mechanisms and Machine Theory*, Vol. 31, No. 5, pp. 573-605.

Meggiolaro, M. & Dubowsky, S. (2000). An Analytical Method to Eliminate the Redundant Parameters in Robot Calibration, *Procedeengs of the International Conference on Robotics and Automation (ICRA '2000),* IEEE, San Francisco, CA, USA, pp. 3609-3615.

Mooring, B. W.; Roth, Z. S. & Driels M. R. (1991). *Fundamentals of manipulator calibration*, John Wiley & Sons, Inc., USA.

Neugebauer, R.; Wieland, F.; Schwaar, M. & Hochmuth, C. (1999). Experiences with a Hexapod-Based Machine Tool, In: *Parallel Kinematic Machines: Theoretical Aspects and Industrial Requirements*, pp. 313-326, Springer-Verlag, C.R. Bor, L. Molinari-Tosatti, K. S. Smith (Eds).

Omodei, A.; Legnani, G. & Adamini, R, (2001). Calibration of a Measuring Robot: Experimental Results on a 5 DOF Structure, *J. Robotic Systems* Vol. 18, No. 5, pp. 237-250.

Parenti-Castelli, V. & Di Gregorio, R. (1995). Determination of the actual configuration of the general Stewart platform using only one additional displacement sensor. In: *Proc. of ASME Int. Mechanical Engineering Congress & Exposition*, Nov. 12-17 1995, San Francisco, CA, USA.

Patel, A. & Ehmann, K. F. (1997). Volumetric Error Analysis of a Stewart Platform Based Machine Tool, *Annals of CIRP*, Vol. 46, No. 1, pp. 287-290.

Press, W.H.; Flannery, B.P.; Teukolsky, S.A.; Vetterling, W.T. (1993). Numerical Recipes in C/Fortran/Pascal, Cambridge University Press.

Reinhar, G.; Zaeh, M. F. & Bongardt, T. (2004). Compensation of Thermal Errors at Industrial Robots, *Production Engineering,* Vol. XI, No. 1.

Ryu, J. & Rauf, A. (2001). Efficient Kinematic Calibration of Parallel Manipulators using a Double Ball Bar System, *Proc. 32nd Int. Symp. on Robotics*, April 2001, Seoul, Korea.

Sheth, P. N. & Uicker, J. J. Jr (1971). A Generalized Symbolic Notation for Mechanisms, *Journal of Engineering for Industry, Transactions of ASME*, pp.102-112, Feb. 1971.

Smollett, R. (1996). A method for Measuring the Position and Orientation of a Stewart Platform, *Proc. of the 1996 Florida Conf. on Recent Advances in Robotics*, pp. 257-260.

Tosi, D. & Legnani, G. (2003). Calibration of a Parallel-Serial Hybrid Redundant Manipulator, *International Robots & Vision Show/34th International Symposium on Robotics*, Rosemont (Chicago), June 3-5, 2003.

Visher, P. (1996) Improving the Accuracy of Parallel Robots, *Phd thesis*, E.P.F de Lausanne.

Wang, J. & Masory, O. (1993). On the accuracy of a Stewart platform-Part I: The effect of manufacturing tolerances, *Proceedings of IEEE International Conference on Robotics and Automation*, pp. 114-120, Atlanta, 2-6 May 1993.

Wang, J.; Masory, O. & Zhuang, H. (1993). On the Accuracy of a Stewart Platform-Part II: Kinematic Calibration and Compensation, *Proceedings of IEEE International Conference on Robotics and Automation*, Atlanta, 2-6 May 1993.

Weck, M.; Giesler, M.; Meylahn, A. & Staimer, D. (1999). Parallel Kinematics: the Importance of Enabling Technologies, In: *Parallel Kinematic Machines: Theoretical Aspects and Industrial Requirements*, pp. 283-294, Springer Verlag, C.R. Boer, L. Molinari-Tosatti, K. S. Smith (Eds.).

Wildenberg, F. (2000). Calibrations for Hexapode CMW, *proceedings of 3rd Chemnitzer Parallel kinematik-Seminar: Working Accuracy of Parallel Kinematics*, pp. 101-112, Verlag Wissenschaftliche Scripten.

Ziegert, J.C.; Jokiel, B. & Huang, C.C. (1999). Calibration and Self-Calibration of Hexapod Machine Tools, In: *Parallel Kinematic Machines: Theoretical Aspects and Industrial Requirements*, pp. 205-216, Springer Verlag, C.R. Boer, L. Molinari-Tosatti, K. S. Smith (Eds.).

Zhuang, H. & Roth, Z. S. (1992). Robot calibration using the CPC error model, *Robotics & Computer-Integrated Manufacturing*, Vol. 9, No. 3, pp. 227-237.

Zhuang, H.; Roth Z. S. & Hamano, F. (1992). A Complete and Parametrically Continuous Kinematic Model for Robot Manipulators, *IEEE Transactions on Robotics and Automation*, Vol. 8, No. 4, August 1992, pp. 451-463.

Interfacing the Gryphon Robot with the World Wide Web

Robert T. F. Ah King, Saravanen Mootien, Harry C. S. Rughooputh
University of Mauritius
Mauritius

1. Introduction

The field of robotics encloses mechanics, electronics, computer intelligent control and network communication technology. It has application in industry, exploration, public service and medicine. Through supervisory control modes, robotic system can be controlled from remote places and thus preventing the user from direct contact with hazardous and inaccessible environment. Moreover, with the development of computer network and the Internet, connectivity can be achieved to remote devices virtually anywhere in the world. Besides, distance education is becoming a widely used common way of teaching different disciplines (Khan, 1997). Along this line, in addition to introducing the field of robotics through the World Wide Web, this medium can be used as a link to program and control the Gryphon robot remotely. The motivation behind the development of this system was to develop a low cost implementation (You, 2001) to enable students to learn robotics, program and run the robot remotely with existing facilities at the Faculty of Engineering of the University of Mauritius. The proposed system (Mootien, 2003) is based on the client/server model, with a suitable front-end for viewing and interactive sessions with web users and a back-end relational database for storing and retrieving relevant information at the server side. A Web-based interface is also developed to enable online training of students on the programming and control of the robot. This chapter presents a detailed description of the work that has been carried out from the design to the implementation and application (use) of the system. The hardware and software technologies that have been used for implementing the system are Microsoft Internet Explorer, Microsoft Internet Information Services (IIS), VBScript, JavaScript and Hyper Text Markup Language (HTML), Microsoft Active Server Pages (ASP), Microsoft Access, Virtual Network Computing (VNC), Microsoft NetMeeting, WALLI3 for Windows and Creative Web Cam.

2. Gryphon Ec Precision Robot

The Gryphon Ec Robot, depicted in Fig. 1, forms part of a family of robots designated by the Walli group of robots, which are the products of the Italtec Company, Italy. Gryphon is a conventional robot emulating the movement of the human arm and hand and having axes for rotation about the shoulder, elbow and wrist in the forward plane. The wrist has two effective axes, for rotation and elevation. There are five axes in all and a gripper. Either a two fingered or a vacuum gripper may be fitted and these may be readily interchanged. The one actually available is the vacuum gripper. The gripper requires an air supply with a pressure of about 5 to 8 bar. The gripper entry is either 0 or 1, indicating open or closed for the Pneumatic actuated gripper. A control box is supplied together with the robot. Any other compatible devices may be connected to form a workcell via the control box. The articulated arm is under the control of 4 microprocessors and, if properly programmed, will accurately work with components in a workcell. Each axis is powered by a stepper motor with optical encoder feedback to provide closed-loop control. In the controller there is one microprocessor to monitor the positions of the axes, two more to control the motors and another one to supervise the first three and to communicate with the host computer. Programming may be accomplished in a variety of ways. Data for each axis may be directly entered on-screen. Alternatively, the Gryphon EC Robot may be moved by the teach pendant or by hand (lead-by-the-nose).

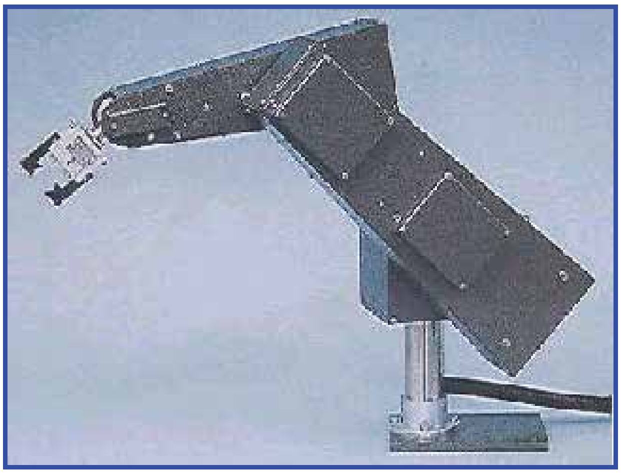


Fig. 1. Gryphon Ec Robot.

2.1 Modes of Robot Programming

The following techniques can be used to program and control the robot to complete an assigned task:

Keyboard	~ Direct entry of axis data from keyboard (using WALLI3 Software).
Lead By Nose	~ Axis data taken from Robot power off.
Pendant	~ Axis data taken by moving the robot under Push button control.
Simulator	~ Axis data taken from simulator positions.

Keyboard Control is described in the next section as this mode is used for interfacing the robot with the World Wide Web. Lead-By-Nose describes the means of programming a robot by physically moving its end effector, referred to as the nose, through the trajectories which the robot will be required to reproduce automatically. To be totally effective, it is necessary for the forces, that are required to move the robot manually, to be very light. Pendant Control is an alternative physical mode of robot programming comparable to Simulator Control, but on one axis at a time. It is provided for Gryphon since it has digital positional feedback through encoders. The robot may be moved using push buttons on the Teach Pendant, without the host computer supporting the WALLI3 software being connected. Continuous programming is not possible by this means and it is only a suitable alternative for point-to-point Simulator control. The Gryphon Robot has an additional accessory, called a simulator, which mechanically mimics the robot in a miniature and simplified form. The movement of each of the simulator axis is monitored through potentiometer sensor devices, so that manual movement of the simulator may be reflected through the control system, to move the robot. When a robot is being moved in this remote manner, it is often referred to be in Teledictor control mode.

2.2 Keyboard Control: Using WALLI3

WALLI3 (Workcell Amalgamated Logical Linguistic Instructions, version 3), shown in Fig. 2, is an updated and extended version of WALLI software written for the Windows environment. As the expanded name suggests, the application is concerned with coordinating the activity within an automated or robotic workcell. It as capable of supporting stand-alone robotic devices or a combination of devices and components within a workcell built to demonstrate the organization of automated production. ***WALLI3*** or ***Walli for Windows*** is the software (Italtec, 1997) that has been written to provide the overall supervisory control for the WALLI range of robots, CNC machines and peripheral or sub-devices, and to integrate them into an automated workcell. The contents of the workcell is entirely flexible and may include any combination and number of the devices that the user wishes to specify and may be changed at will to accommodate different teaching requirements. Because it is written for the Windows environment or Operating System, the software must be installed into a computer with an INTEL 386 processor as the minimum requirement. The software will run simply as an installed software package, simulating all the machine's actions. But to control any of the robotic hardware, an interface card must be installed into the PC. The interface card is connected into the internal ISA bus of the computer and the card may be driven up to eight hardware devices. Furthermore, a second interface card may co-exist if the required installation becomes extensive.

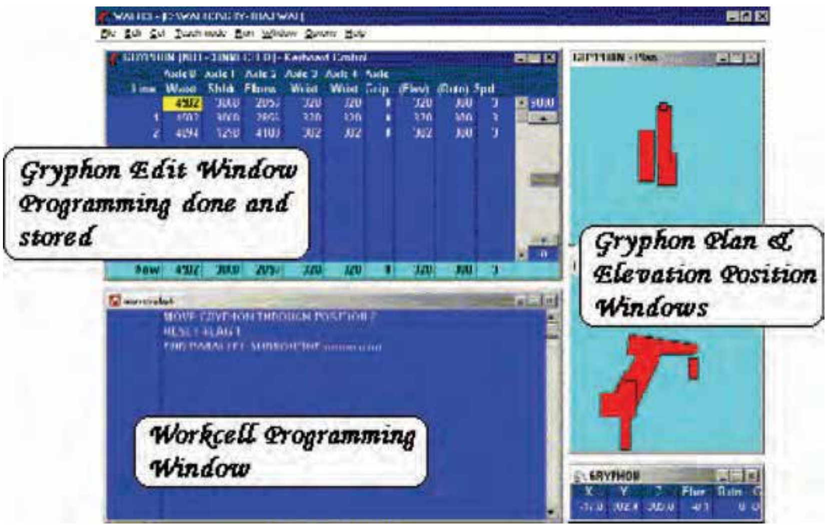

Gryphon Edit Window Programming done and stored
Gryphon Plan & Elevation Position Windows
Workcell Programming Window

Fig. 2. WALLI3 Software's Screen..

2.3 Working with the Gryphon Robot

Equipped as it is with a gripper as an end effector, the Gryphon is essentially a "pick and place" robot, for picking things up and placing them somewhere else. This is exactly its employment within the workcell, taking model component from the end of a conveyor and placing them into bins, onto another conveyor or Index table. Any use of the robot should be likely some variation of this basic function. The opening and closure is an on/off function, it is not progressive, and jaws travel their full extent. The opening width may be adjusted, but they will always close fully unless they grip on an object. In the Gryphon Robot, the same stepper motors and gearboxes are used on axes 0 (Waist), 1 (Shoulder), 2 (Elbow) and smaller motors and gear boxes for axes 3 (Rotation) and 4 (Elevation). Axes 0, 1 and 2 move at approximately the same rate over large excursions, depending on the speed set, while the combined effect of axes 3 and 4 are more rapid.

2.4 Working Environment

For the Gryphon Robot the theoretical outer limit may be readily demonstrated by measuring the full extent of the arm and gripper, in both a horizontal and vertical position, at its limits of travel and from measurement of the shoulder axis height above the workcell surface. Fig. 3 shows the robot outer limit of reach. Referring to the figure, Gryphon has a net offset to the right of the shoulder axis from the plane of the robot arm, and a forward offset. Therefore the Sphere of Influence has a flat pole.

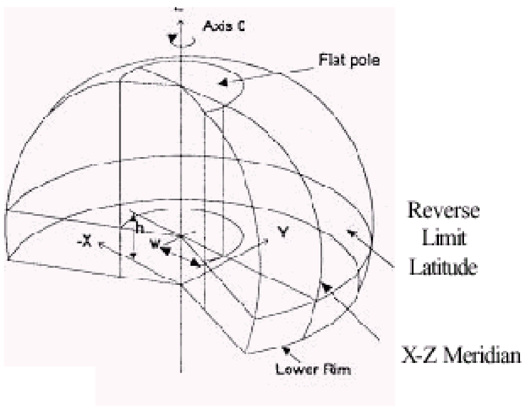


Fig. 3. Sphere of Influence of Gryphon Ec Robot.

3. Gryphon Digital Control System

Gryphon has a digital control system using commercially available incremental encoders to feed back position information for each axis. The control circuit uses a master processor and three slave processors and the axes are driven by stepper motors. Two slave processors are used for the five axes motors on Gryphon, one for the waist and wrist control, the other for the shoulder and elbow axes. A third slave processor is dedicated to servicing all five encoders. The Gryphon stepper motors are driven through a matched combination of integrated circuits, the bridge driver and stepper motor controller- one pair for each motor. Instead of pulse modulation, the bridge driver accepts a pulse and direction signal from an up down counter in the motor controller driven by one of the slave processors. The basic scheme is repeated for all axes and the overall schematic is summarized in Fig. 4.

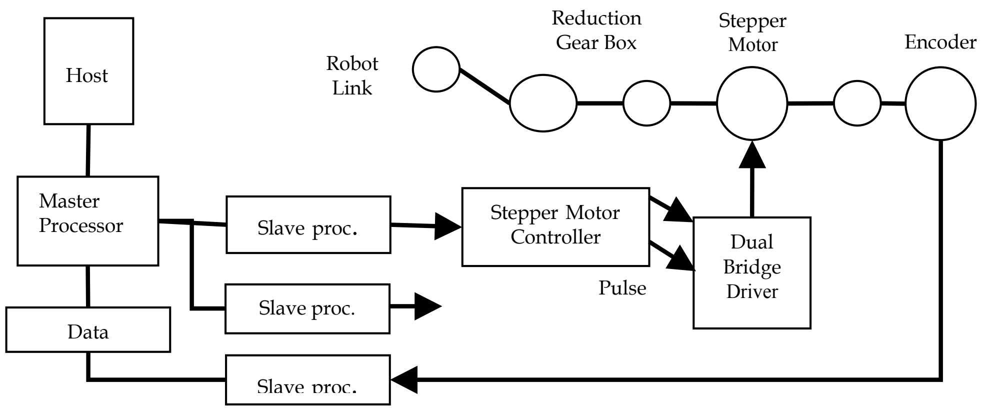


Fig. 4. Gryphon Axis Control System.

The Gryphon Robot is powered by variable reluctance stepper motors coupled through worm and wheel gearboxes to each of the five axes. The stepper motor resolution and gearbox ratios allow a high degree of mechanical resolution in moving the axes, while the use of geared optical encoders for position feedback, giving resolution of up to 16000 steps per revolution, allows the control system to take full advantage of the drive resolution to ensure precision in positioning the robot axes.

3.1 Digital Control Boards

The majority of WALLI devices have digital control systems built from a set of common, notably a master and a slave processor printed circuit board, an eight channel 12 bit ADC to the slave format, and a stepper motor driver. Both the master or CNC processor and the slave processor use the INTEL 80C32 micro-controller, with external RAM and ROM provided for the master that also uses the enhanced 16MHz version of the micro-controller. The CNC processor is in communication with the host processor via a serial link formed from four twisted pairs, to an RJ45 Interface Board in the host computer. This board is the preferred means of communication between the host PC and the control box and is used for all digital control systems. Gryphon uses three slave processors, one to service the encoders used for position feedback on all five axes, the two other to control the stepper motor drives to axes 1 and 2, and to axes 0, 3 and 4, respectively. It also uses individual control boards for each of the stepper motors with load current regulation control to ensure good speed-torque characteristics. For Gryphon, this amounts to a total of five 2A stepper drivers.

3.2 12-Bit ADC

The 12-bit ADC circuit board provides the means of connecting the simulator programming devices into the CNC processor used in the Gryphon Robot control box. Together with the Simulator, other hand-manipulated devices have potentiometers mounted at each of the joints, which are representative of the robot axes, and there is a consequent need to transfer the analog signals to digital form. The board is built to the same physical format as the slave processor so that it may be stacked with them in the control cabinet and connected into the 34 way parallel interface between the master CNC processor and the slave processors.

3.3 Stepper Driver
The 2A stepper driver board provides an individual circuit to drive the stepper motors for the axes of the robot and the connection is taken from the slave processors output ports. The stepper motor controller provides all the necessary signals to drive the stepper motor in the correct sequence to advance the motor shaft in the required direction by a step.

3.4 RJ45 Serial Interface Board
The RJ45 interface is now the preferred means of communication between the compatible host computer and the WALLI robots family. But for the Gryphon Ec Robot it is the only means of communication. The RJ45 interface uses serial transmission over four signal lines, arranged as four twisted pairs to minimize cross-talk and is mapped into the I/O space of the PC. The RJ45 interface is a proprietary design for serial transmission and uses a pre-formed cable with a standard RJ45 plug at each end. The transmission is synchronous with the clock signal carried on one twisted pair and with 8 bits serial information, designated address, control and data, carried on the other three. A clock rate of 2 MHz allows the 8 bit address, control and data information to be transmitted at a rate of 250 Kbytes per second on each of the three lines concurrently, giving an overall maximum transfer rate of 750 Kbytes per second. The clock signal from the PC interface is carried directly to the address, control and data registers in the device interface, there is no requirement for additional protocol and the device address and functionality is derived from the information received.

4. System Objectives and Software Development

This section describes the main objectives for setting up the system and the framework considered in developing the software for interfacing the Gryphon robot with the World Wide Web.

4.1 System Objectives
The main objectives of the project (Mootien, 2003) are as follows:

- To design a web site to act as an introduction to Robotics.
- To enable the web site to behave as a platform to the control of Gryphon Robot virtually anywhere on the Internet/Intranet.
- To introduce Gryphon Ec Robot, its compatible WALLI3 software and other related technologies in the Web Site.
- To enable any user to send his/her comments about the Web Site.
- To permit restricted number of users to be in command of the robot.
- To obtain a video feedback of the robot and its environment.

4.2 Software Process Model
A software process can be defined as a framework for the tasks that are required to build a software system. In order to develop the actual project, solve the related problems and reach the set objectives, a development strategy was adopted. The linear sequential model for software engineering was selected. As the name suggest, the project was started through chronological approach to the software development, that started at the system level and progressed through analysis, design, coding and finally testing.

The following activities were enclosed within the linear sequential model shown in Fig. 5:

- System/Information Engineering:
  - The requirements for the project were established.
  - Information research and background study for different aspects and subsets of the system/project were dealt with accordingly.
  - Interface with other elements such as hardware, people and databases were considered.
  - Initial designs were proposed.
- Analysis:
  - Research was more specifically focused on software and technologies to be used.
  - The software constraints and performance were established and evaluated.
  - Choice of software and other requirements were specified.
  - A reliable proposed solution was maintained.
- Design:
  - The steps towards the implementation were explained explicitly.
  - The configuration of the web site and subsequent pages were set.
  - Flowcharts and state diagrams were used to initialize the coding part.
- Code Generation:
  - The design was coded with selected programming languages.
  - The code generated was mingled with the appropriate information for particular web pages.
  - Interfacing between different programming languages and tools were dealt with accordingly.
- Testing:
  - All functions were tested and proper corrections were made.
  - Interactions between hardware, software and human operator were considered and necessary changes were brought to the system.

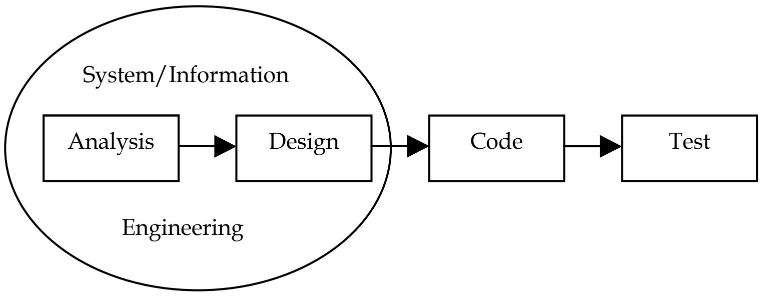


Fig. 5. The Linear Sequential Process Model.

5. Proposed System

Since the RJ45 Interface Board is a proprietary design for the Gryphon Ec Robot and the WALLI technology, the proposed system should inevitably incorporate the Virtual Network Computing (VNC) technology. VNC is, in essence, a remote display system which allows one to view a computing 'desktop' environment not only on the machine where it is running, but from anywhere on the Internet and from a wide variety of machine architectures. In addition to this tool, the proposed system must have a video feedback for the robot and its environment. For this to be implemented, the Netmeeting technology developed by Microsoft should be used. Netmeeting delivers an open, extensible platform for real-time communications, offering audio and video. Besides the use of these technologies, it is required to build a web site that would act as the link between these tools and the Gryphon system. Moreover, it is mandatory for the web site to have an introduction to the field of robotics and associated topics. Fig. 6 describes the overall proposed solution.

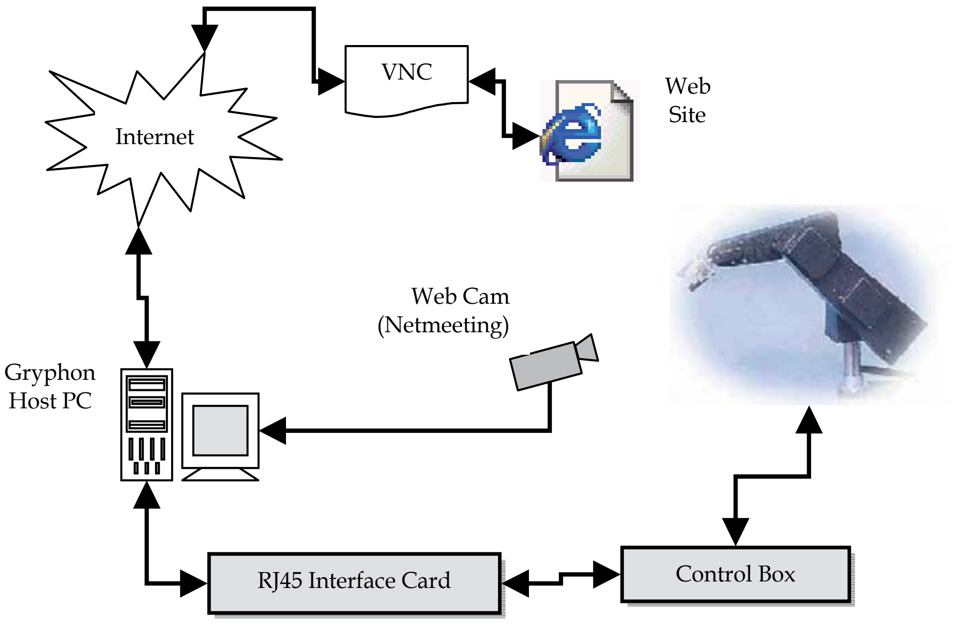


Fig. 6. Proposed System.

The proposed system is developed using the following:

- Microsoft Internet Explorer 5.0 as Front-end browser.
- IIS 4.0 as web server.
- VBScript for server-side scripting and client-side validation.
- JavaScript and HTML for client-side programming.
- Microsoft Active Server Pages (ASP) 3.0 as programming technology.
- Microsoft Access as Back-end for data storage.
- VNC as the connection to the robot
- Microsoft NetMeeting as the software for video transmission

5.1 Microsoft Internet Explorer 5.0 as Front-End Browser

Internet technology such as HTML, DHTML and ASP are used for developing the web site and this system has its own particularities, which makes it a must for considering alternative browsers capabilities. Popular browsers for viewing hypertext documents are Microsoft Internet Explorer and Netscape Communicator. Microsoft Internet Explorer is a powerful browser, which runs on all Windows Operating Systems family. It supports multimedia, graphics images and ActiveX components also, and suits the requirements of

ASP. Netscape runs on Windows based platforms, and supports multimedia and graphics images. However, Netscape lacks certain requirements of ASP, as Netscape Communicator does not recognize all syntax of VBScript but instead recognizes JavaScript. The solution is to opt for Internet Explorer 5.0 Browser because it supports efficiently ASP technology and VBScript. Moreover, since the browser is widely used and actually the most common browser on the Internet, the logical solution is Internet Explorer.

5.2 Microsoft Internet Information Services (IIS) 4.0 as Web Server

The pivoting role of processing HTTP requests and sending out static Web pages to the client browser in response is the Web Server. Apart from this, the Web Server can also execute applications that significantly enhance the content of a Web site. Issues governing the choice of a Web server are speed at which the server executes Web-applications; security; whether it supports virtual directories; restricting access with IP addresses. Microsoft Internet Information Services or IIS is a secure server with its security system linked to Windows NT. Together, IIS and Windows NT provide a number of security layers through which a user must pass before gaining access to a site. In brief, it has the depth and breadth of security features required by the most exigent Web site. Personal Web Server (PWS) is a server designed for use with Microsoft Windows 95/98. Whilst PWS supports many but not all the features of IIS, it is more tailored for developing applications off line rather than for functioning as a robust, flexible and secure Web Server. PWS allows for virtual directories and execution can be done for a limited number of clients. Moreover, it lacks hold of security. PWS cannot and has not been intended to compete with IIS. Therefore, the obvious choice for a robust, flexible, secure Web Server is IIS.

5.3 VBScript for Server-Side Scripting and Client-Side Validation

VBScript is a scripting language that allows functions to be written be embedded in HTML documents. With VBScript, creation of rich, dynamic, interactive Web content is possible. VBScript has a rich feature set that provides a very good environment for developing client and server applications. Many tasks that once required processing on the server can be done on the client. This reduces both the number of requests to the server and the workload placed on the server. However, because VBScript is interpreted, the host environment must have the correct software to execute the code. In the case of client-side script, the Web browser must be aware of VBScript. Microsoft Internet Explorer has this ability built-in, but many other browsers require a plug-in to interpret VBScript.

5.4 JavaScript and HTML for Client-Side Programming

JavaScript is an object-based language developed by Netscape. Syntactically, it is very similar to Java but is not a subset of it. JavaScript is well suited for quick development and easy maintenance of relatively small programs. JavaScript is a scripting language that allows programs to be written that resides inside HTML documents. It is an interpreted language, so the host environment or Web browser, must have a JavaScript engine to execute the code. Both Microsoft's Internet Explorer and Netscape's Navigator support JavaScript.

5.5 Microsoft Active Server Pages (ASP) 3.0 as Programming Technology

Up to now, with the help of Client-side programming, pages are published with some degree of interactivity due to client-side scripting. However, some applications may need to access databases to execute queries. ASP can be described as a server-side scripting environment that can be used to create and run dynamic, interactive, high-performance Web Server applications (Jones, 2000). ASP files combine HTML, scripts, and ASP code to enable a much higher degree of interactivity than is possible with plain HTML. ASP can be used to include executable scripts directly into HTML files. Microsoft Active Server Pages is a server-side scripting environment that allows programmers to create and run dynamic high-performance Web server applications. ASP scripts run on the server, instead of the client, and the Web server sends the output as an HTML page to the client-browsers. There is no need to worry whether the browser is compatible with the scripting language. The browser only view the end-result of the server-side processed HTML page. ASP encourages browser independence. Other benefits of ASP applications are as follows:

- Completely integrated with HTML files.
- Easy to create, with no manual compiling or linking of programs required.
- ASP enables programmers to use any scripting language provided ASP support the language or script. ASP supplies scripting engines for Microsoft Visual Basic Scripting Edition (VBScript) and Jscript. For this project, VBScript is the language used.
- Flexible and secure: With ASP, developers can choose whether to run their scripts at the client or server to meet their specific needs. Furthermore, since scripts and server components are executed at the server, they remain hidden to the user.

5.6 Microsoft Access as Back-End for Data Storage

The performance of the system highly relies upon the back-end to be included in the system. The primary goal of SQL Server is to allow data to exist in multiple formats and be accessed using many different methods. With SQL Server, Microsoft wanted to provide not only a more powerful relational database management system, but also a mechanism for gathering contrasting information stores and for presenting data in a consistent and useful manner. It is as well used when databases are involved in replication. The SQL Server replication technology copies data, moves these copies to different locations, and synchronizes the data so that all copies have the same data values. Access can make a good back-end package, compared to other desktop database packages. Access has one huge advantage in that it is most likely that users are running Windows as operating system and using Microsoft Office as their application base. Access integrates well with these packages and data transfer between Access and the other office components is relatively easy. In addition to this, Access is very easy to use, almost for anyone of any level. Active Server Pages supports Microsoft Access as a valid data source. A great benefit is certainly that Access offers portability, meaning that it can be moved from server systems to another without altering its functionality and configuration. Microsoft Access is the appropriate solution because all the characteristics that are required for the system are observed in it. Moreover, Access is used since the proposed design does not require large and complex database utilities.

5.7 VNC as the Connection to the Robot

The Virtual Network Computing (VNC) technology (Richardson et al., 1998; AT&T, 1999) developed for ATM Network Computers by AT&T is a remote display system which allows one to view a computing 'desktop' environment not only on the machine where it is running, but from anywhere on the Internet and from a wide variety of machine architectures. VNC consists of two types of component: a Server which generates a display and a Viewer which actually draws the display on the remote screen. The server and the viewer may be on different machines and on different architectures. The protocol which connects the server and viewer is simple, open, and platform-independent. The current VNC software requires a TCP/IP connection between the server and the viewer. Thus, VNC is ideal for calling WALLI3 software for programming and operating the robot remotely.

5.8 Microsoft NetMeeting as the Software for Video Transmission

When the Gryphon Robot is under operation, feedback is essential to know whether or not the work assigned has been fulfilled. With WALLI3, only the robot's position can be seen, hence to view all the workstation visual information is mandatory. The system must have a video feedback for the robot and its environment. Since Microsoft NetMeeting delivers an open, extensible platform for real-time communications, offering audio and video (Microsoft, 2000), it is used for video capture and feedback to view the robot action as the program is executed.

6. System Development

The proposed system will mainly consist of client and server web pages linked together using ASP, HTML and communicating with a back-end database server. The web pages should give an introduction to the robotic technology and other fields related to this project. As stated in the objectives, the web pages should also give a detailed description of the mechanical device involved and the means used for communicating with it, i.e. VNC and Netmeeting.

6.1 Graphical Layout

The web site should use two frames to efficiently ease travel within the site. The first frame should host only the contents page while the other one should host the remaining pages upon request from the contents page. The different levels of menu and links to be implemented are shown in Fig. 7. The web pages are described in the next section.

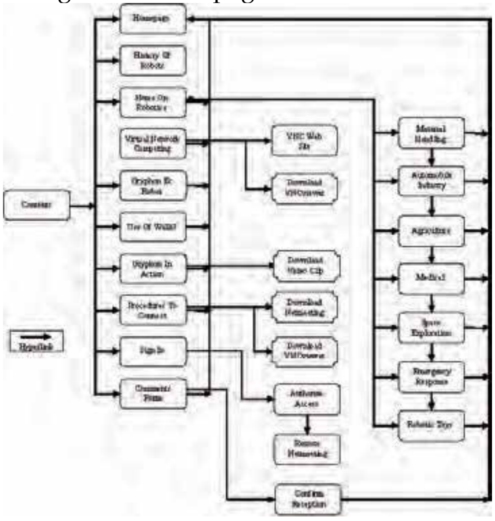
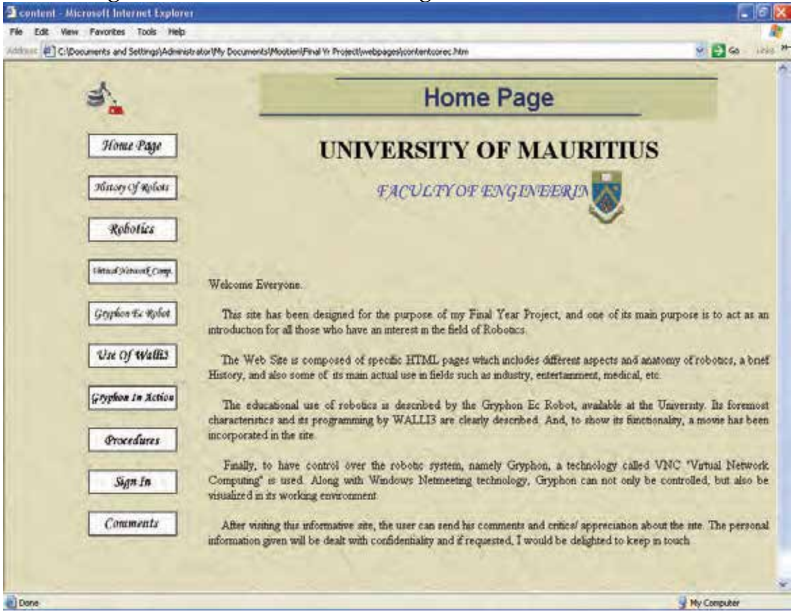


Fig. 7. Graphical Layout of Proposed System .

6.2 Web Pages

The Homepage introduces the project and provides an overview of the content of the entire site. All the other web pages are redirected to the Homepage. The Homepage is illustrated in Fig. 8 and the image has been embedded using the HTML codes.


content - Microsoft Internet Explorer
File Edit View Favorites Tools Help
C:\Documents and Settings\Administrator\My Documents\Moobien\Final Yr Project\webpages\contentcorec.htm
Home Page
History Of Robots
Robotics
Virtual Network Comp.
Gryphon Ec Robot
Use Of Wallis
Gryphon In Action
Procedures
Sign In
Comments
Done
My Computer

Home Page

UNIVERSITY OF MAURITIUS

FACULTY OF ENGINEERIN

Welcome Everyone.

This site has been designed for the purpose of my Final Year Project, and one of its main purpose is to act as an introduction for all those who have an interest in the field of Robotics.

The Web Site is composed of specific HTML pages which includes different aspects and anatomy of robotics, a brief History, and also some of its main actual use in fields such as industry, entertainment, medical, etc.

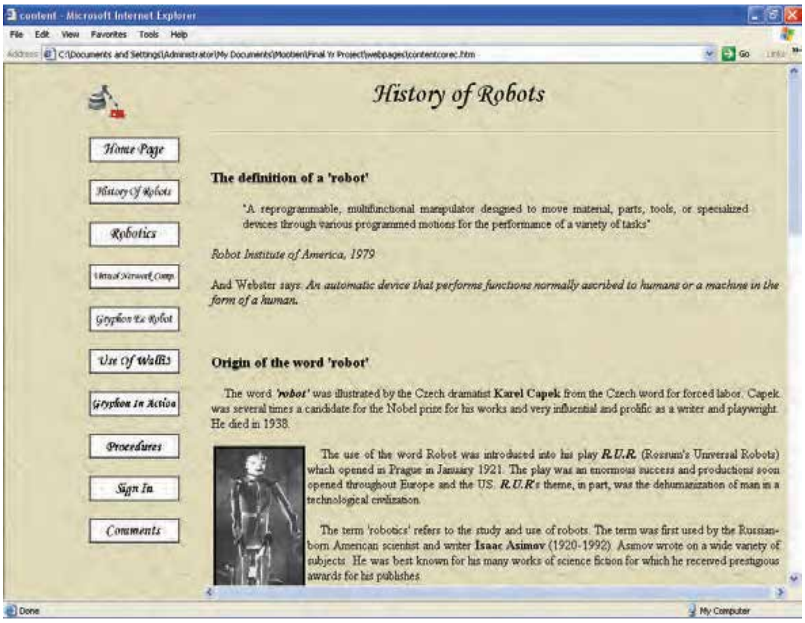
The educational use of robotics is described by the Gryphon Ec Robot, available at the University. Its foremost characteristics and its programming by WALLI3 are clearly described. And, to show its functionality, a movie has been incorporated in the site.

Finally, to have control over the robotic system, namely Gryphon, a technology called VNC "Virtual Network Computing" is used. Along with Windows Netmeeting technology, Gryphon can not only be controlled, but also be visualized in its working environment.

After visiting this informative site, the user can send his comments and critical appreciation about the site. The personal information given will be dealt with confidentiality and if requested, I would be delighted to keep in touch.

Fig. 8. Homepage

The web page on the History of Robots shown in Fig. 9 illustrates the existence of robots and milestones in the robotic industry.


content - Microsoft Internet Explorer
File Edit View Favorites Tools Help
C:\Documents and Settings\Administrator\My Documents\Moobien\Final Yr Project\webpages\contentcorec.htm
Home Page
History Of Robots
Robotics
Virtual Network Comp.
Gryphon Ec Robot
Use Of Wallis
Gryphon In Action
Procedures
Sign In
Comments
Done
My Computer

History of Robots

The definition of a 'robot'

"A reprogrammable, multifunctional manipulator designed to move material, parts, tools, or specialized devices through various programmed motions for the performance of a variety of tasks"

Robot Institute of America, 1979

And Webster says: *An automatic device that performs functions normally ascribed to humans or a machine in the form of a human.*

Origin of the word 'robot'

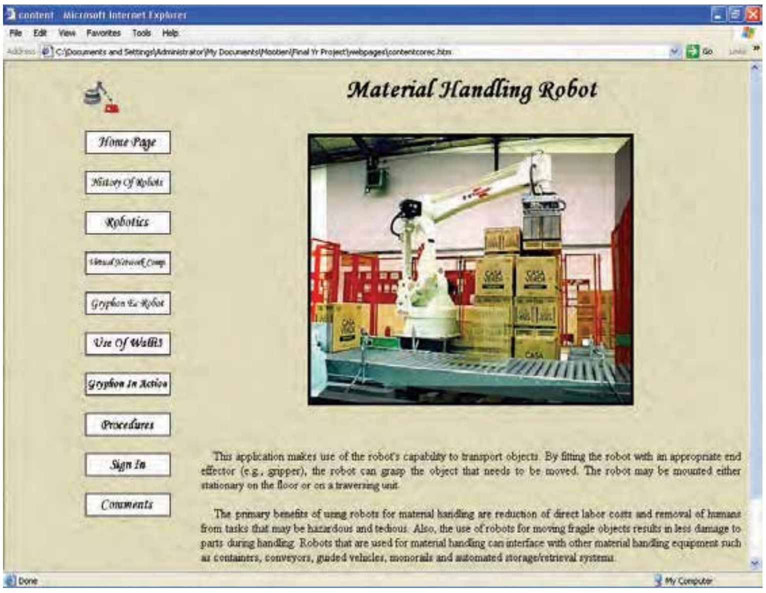
The word ***'robot'*** was illustrated by the Czech dramatist **Karel Capek** from the Czech word for forced labor. Capek was several times a candidate for the Nobel prize for his works and very influential and prolific as a writer and playwright. He died in 1938.

The use of the word Robot was introduced into his play ***R.U.R.*** (Rossum's Universal Robots) which opened in Prague in January 1921. The play was an enormous success and productions soon opened throughout Europe and the US. ***R.U.R's*** theme, in part, was the dehumanization of man in a technological civilization.

The term 'robotics' refers to the study and use of robots. The term was first used by the Russian-born American scientist and writer **Isaac Asimov** (1920-1992). Asimov wrote on a wide variety of subjects. He was best known for his many works of science fiction for which he received prestigious awards for his publishes.

Fig. 9. History of Robots Screen Shot.

Applications and uses of robots in different sectors: Material Handling (Fig. 10), Automobile Industry, Agriculture, Medical, Space Exploration, Emergency Response, Robotic Toys are emphasized.


content Microsoft Internet Explorer
File Edit View Favorites Tools Help
Address C:\Documents and Settings\Administrator\My Documents\Mootien\Final Yr Project\webpages\contentcorec.htm
Home Page
History Of Robots
Robotics
Virtual Network Comp
Gryphon Ex Robot
Use Of WalliI3
Gryphon In Action
Procedures
Sign In
Comments
Done
My Computer

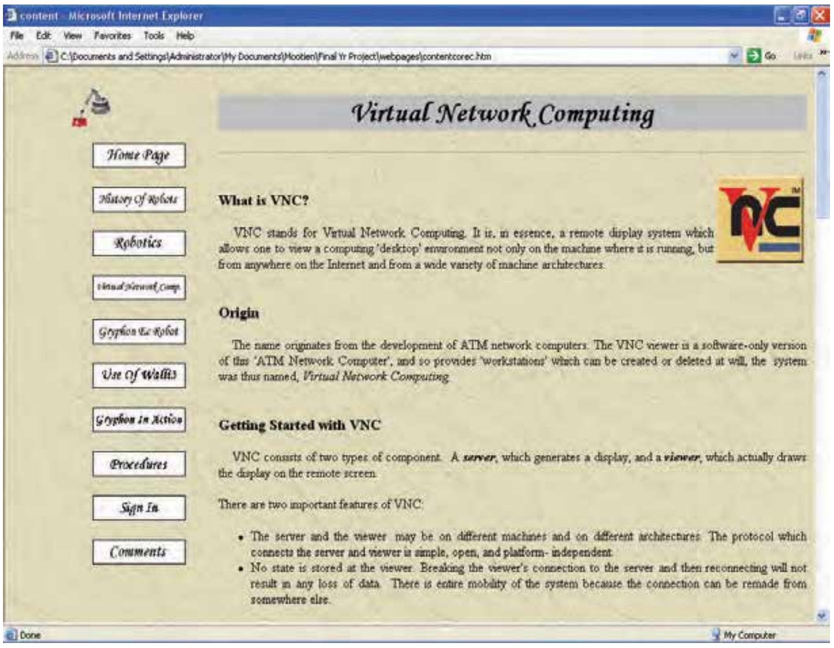
Material Handling Robot

This application makes use of the robot's capability to transport objects. By fitting the robot with an appropriate end effector (e.g., gripper), the robot can grasp the object that needs to be moved. The robot may be mounted either stationary on the floor or on a traversing unit.

The primary benefits of using robots for material handling are reduction of direct labor costs and removal of humans from tasks that may be hazardous and tedious. Also, the use of robots for moving fragile objects results in less damage to parts during handling. Robots that are used for material handling can interface with other material handling equipment such as containers, conveyors, guided vehicles, monorails and automated storage/retrieval systems.

Fig. 10. Notes on Robotics/Material Handling Screen Shot.

The web page on Virtual Network Computing depicted in Fig. 11 includes its origin, use and other features in relation to the project. This web page provides a link to the official VNC web site and additionally a proper downloadable format of this technology is freely supplied.


content - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Address C:\Documents and Settings\Administrator\My Documents\Mootien\Final Yr Project\webpages\contentcorec.htm
Home Page
History Of Robots
Robotics
Virtual Network Comp
Gryphon Ex Robot
Use Of WalliI3
Gryphon In Action
Procedures
Sign In
Comments
Done
My Computer

Virtual Network Computing

What is VNC?

VNC stands for Virtual Network Computing. It is, in essence, a remote display system which allows one to view a computing 'desktop' environment not only on the machine where it is running, but from anywhere on the Internet and from a wide variety of machine architectures.

Origin

The name originates from the development of ATM network computers. The VNC viewer is a software-only version of this 'ATM Network Computer', and so provides 'workstations' which can be created or deleted at will, the system was thus named, *Virtual Network Computing*.

Getting Started with VNC

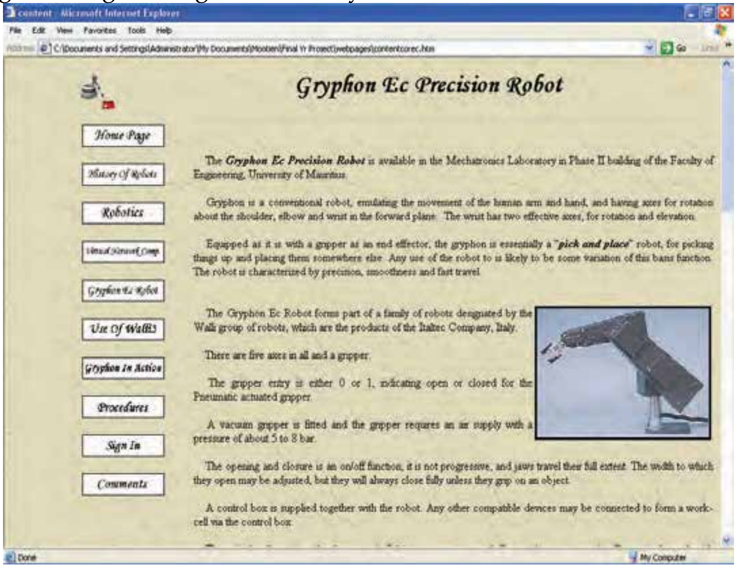
VNC consists of two types of component. A ***server***, which generates a display, and a ***viewer***, which actually draws the display on the remote screen.

There are two important features of VNC:

- The server and the viewer may be on different machines and on different architectures. The protocol which connects the server and viewer is simple, open, and platform- independent.
- No state is stored at the viewer. Breaking the viewer's connection to the server and then reconnecting will not result in any loss of data. There is entire mobility of the system because the connection can be remade from somewhere else.

Fig. 11. Virtual Network Computing.

The web page shown in Fig. 12 dedicated to the Gryphon robot describes its specification, modes of programming and digital control system.


content - Microsoft Internet Explorer
File Edit View Favorites Tools Help
C:\Documents and Settings\Administrator\My Documents\Mootien\Final Yr Project\webpages\contentcorec.htm
Home Page
History Of Robots
Robotics
Virtual Network Comp
Gryphon Ec Robot
Use Of Walli3
Gryphon In Action
Procedures
Sign In
Comments
Done
My Computer

Gryphon Ec Precision Robot

The ***Gryphon Ec Precision Robot*** is available in the Mechatronics Laboratory in Phase II building of the Faculty of Engineering, University of Mauritius.

Gryphon is a conventional robot, emulating the movement of the human arm and hand, and having axes for rotation about the shoulder, elbow and wrist in the forward plane. The wrist has two effective axes, for rotation and elevation.

Equipped as it is with a gripper as an end effector, the gryphon is essentially a "***pick and place***" robot, for picking things up and placing them somewhere else. Any use of the robot to is likely to be some variation of this basic function. The robot is characterized by precision, smoothness and fast travel.

The Gryphon Ec Robot forms part of a family of robots designated by the Walli group of robots, which are the products of the Italtec Company, Italy.

There are five axes in all and a gripper.

The gripper entry is either 0 or 1, indicating open or closed for the Pneumatic actuated gripper.

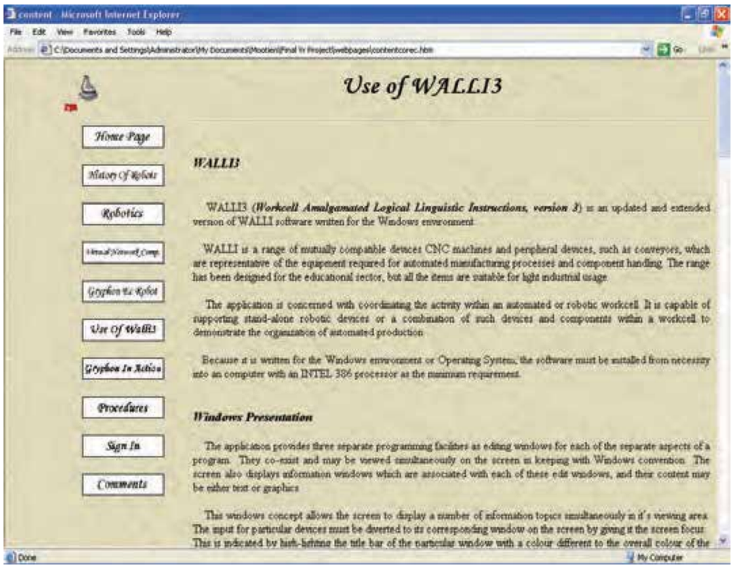
A vacuum gripper is fitted and the gripper requires an air supply with a pressure of about 5 to 8 bar.

The opening and closure is an on/off function, it is not progressive, and jaws travel their full extent. The width to which they open may be adjusted, but they will always close fully unless they grip on an object.

A control box is supplied together with the robot. Any other compatible devices may be connected to form a work-cell via the control box.

Fig. 12. About Gryphon.

Since the mode of programming the robot adopted is keyboard control, WALLI3 software is described in the web page shown in Fig. 13 where the main keys to successful programming are also presented.


content - Microsoft Internet Explorer
File Edit View Favorites Tools Help
C:\Documents and Settings\Administrator\My Documents\Mootien\Final Yr Project\webpages\contentcorec.htm
Home Page
History Of Robots
Robotics
Virtual Network Comp
Gryphon Ec Robot
Use Of Walli3
Gryphon In Action
Procedures
Sign In
Comments
Done
My Computer

Use of WALLI3

WALLI3

WALLI3 (***Workcell Amalgamated Logical Linguistic Instructions, version 3***) is an updated and extended version of WALLI software written for the Windows environment.

WALLI is a range of mutually compatible devices CNC machines and peripheral devices, such as conveyors, which are representative of the equipment required for automated manufacturing processes and component handling. The range has been designed for the educational sector, but all the items are suitable for light industrial usage.

The application is concerned with coordinating the activity within an automated or robotic workcell. It is capable of supporting stand-alone robotic devices or a combination of such devices and components within a workcell to demonstrate the organization of automated production.

Because it is written for the Windows environment or Operating System, the software must be installed from necessity into an computer with an INTEL 386 processor as the minimum requirement.

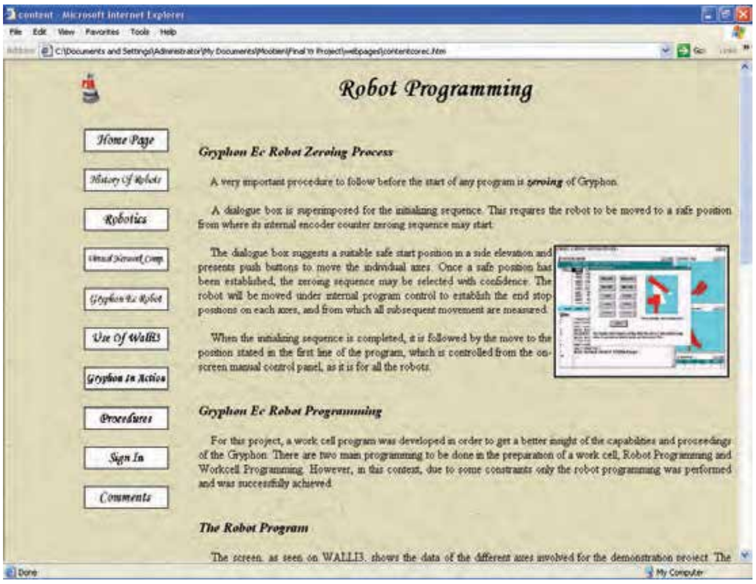
Windows Presentation

The application provides three separate programming facilities as editing windows for each of the separate aspects of a program. They co-exist and may be viewed simultaneously on the screen in keeping with Windows convention. The screen also displays information windows which are associated with each of these edit windows, and their content may be either text or graphics.

This windows concept allows the screen to display a number of information topics simultaneously in it's viewing area. The input for particular devices must be diverted to its corresponding window on the screen by giving it the screen focus. This is indicated by high-lighting the title bar of the particular window with a colour different to the overall colour of the

Fig. 13. Use of WALLI3.

Now that the robot and its supporting control system have been explained to the user, there is a need to demonstrate the capabilities of the robot. A simple robot program has been implemented and this program is explained to the user on the web page presented in Fig. 14. The user is instructed to follow the steps to program and run the robot by calling WALLI3 through the remote interface. Details regarding the connection procedures are described in the next paragraph. The performance of the robot is demonstrated using an embedded video clip in the web page and this clip is downloadable. The user can compare the robot action with the clip to check if he has correctly programmed the robot. Moreover, a more elaborate program is also available and this involves the robot in picking and placing objects on a conveyor.


content - Microsoft Internet Explorer
File Edit View Favorites Tools Help
C:\Documents and Settings\Administrator\My Documents\Mootien\Final Yr Project\webpages\contentcorec.htm
Home Page
History Of Robots
Robotics
Virtual Network Comp
Gryphon Ec Robot
Use Of Walli3
Gryphon In Action
Procedures
Sign In
Comments
Done
My Computer

Robot Programming

Gryphon Ec Robot Zeroing Process

A very important procedure to follow before the start of any program is ***zeroing*** of Gryphon.

A dialogue box is superimposed for the initialising sequence. This requires the robot to be moved to a safe position from where its internal encoder counter zeroing sequence may start.

The dialogue box suggests a suitable safe start position in a side elevation and presents push buttons to move the individual axes. Once a safe position has been established, the zeroing sequence may be selected with confidence. The robot will be moved under internal program control to establish the end stop positions on each axes, and from which all subsequent movement are measured.

When the initialising sequence is completed, it is followed by the move to the position stated in the first line of the program, which is controlled from the on-screen manual control panel, as it is for all the robots.

Gryphon Ec Robot Programming

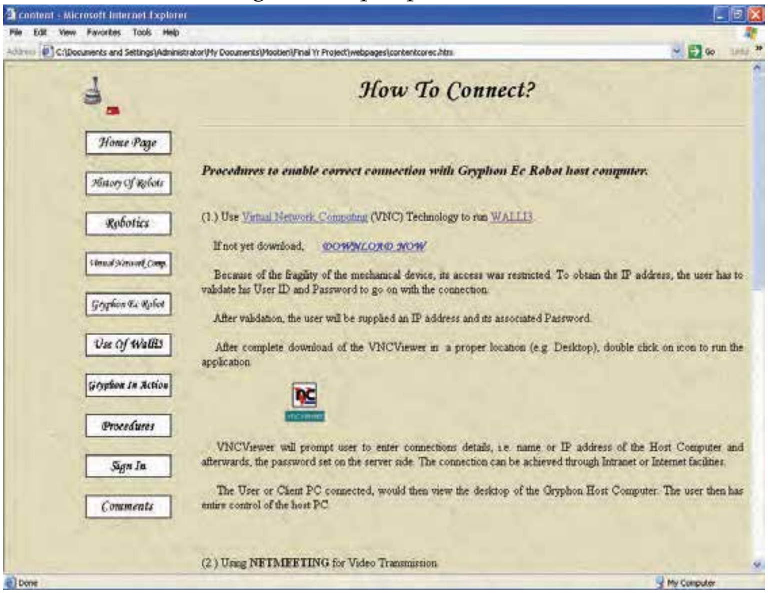
For this project, a work cell program was developed in order to get a better insight of the capabilities and proceedings of the Gryphon. There are two main programming to be done in the preparation of a work cell, Robot Programming and Workcell Programming. However, in this context, due to some constraints only the robot programming was performed and was successfully achieved.

The Robot Program

The screen, as seen on WALLI3, shows the data of the different axes involved for the demonstration project. The

Fig. 14. Gryphon In Action.

The requirements for proper connection to the Gryphon Host PC are described and integration of VNC and Windows NetMeeting are appropriately highlighted as shown in Fig. 15. Also, Windows NetMeeting 3.0 Setup is provided as download.


content - Microsoft Internet Explorer
File Edit View Favorites Tools Help
C:\Documents and Settings\Administrator\My Documents\Mootien\Final Yr Project\webpages\contentcorec.htm
Home Page
History Of Robots
Robotics
Virtual Network Comp
Gryphon Ec Robot
Use Of Walli3
Gryphon In Action
Procedures
Sign In
Comments
Done
My Computer

How To Connect?

Procedures to enable correct connection with Gryphon Ec Robot host computer.

(1.) Use Virtual Network Computing (VNC) Technology to run WALLI3.

If not yet download, DOWNLOAD NOW

Because of the fragility of the mechanical device, its access was restricted. To obtain the IP address, the user has to validate his User ID and Password to go on with the connection.

After validation, the user will be supplied an IP address and its associated Password.

After complete download of the VNCViewer in a proper location (e.g. Desktop), double click on icon to run the application.

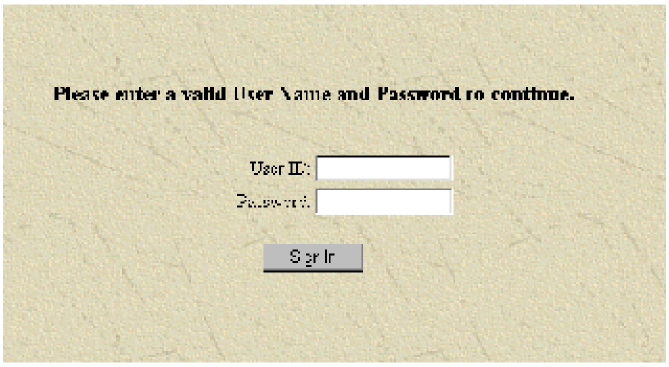
VNCViewer will prompt user to enter connections details, i.e. name or IP address of the Host Computer and afterwards, the password set on the server side. The connection can be achieved through Intranet or Internet facilities.

The User or Client PC connected, would then view the desktop of the Gryphon Host Computer. The user then has entire control of the host PC.

(2.) Using **NETMEETING** for Video Transmission

Fig. 15. Procedures For Connection.

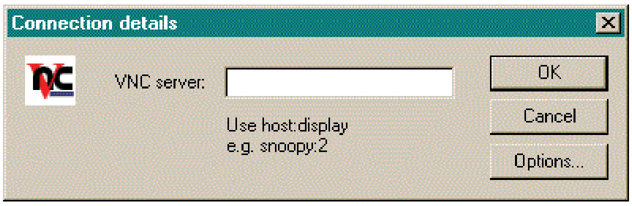
Authentication is an important process for this project. The control part of the web site is not freely available to any visitor. For this sensitive part, ASP technology is extensively used and right of entry is only possible after user name and password be validated. After acceptance of user ID and password (Fig. 16), a web page is available to provide necessary information, e.g. IP address, to establish connection to the remote Gryphon host PC (Fig. 17).


Please enter a valid User Name and Password to continue.

User ID:

Password:

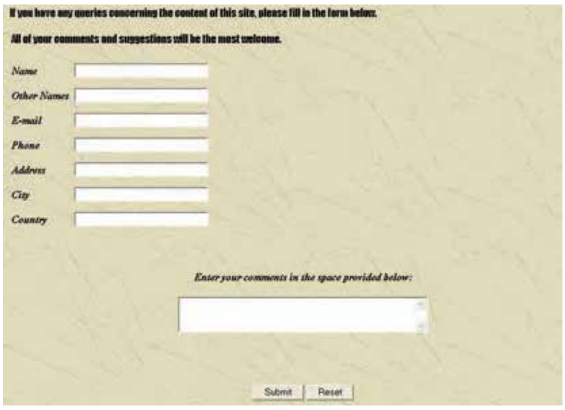
Fig. 16. Login Page.



Use host:display
e.g. snoopy:2

Fig. 17. VNCviewer.

To make the web site interactive with the user and developer, the final stage of the web site includes a Contact and Comment Form represented in Fig. 18. The form supplies sufficient fields so as to facilitate any user to enter his/her personal details, contact points and any comments/suggestions. After correct entry of the required data, the system validates the said entries at the client side. Upon submission of the form to the database, a reply is offered to the user to confirm receipt of the form in the database.


If you have any queries concerning the content of this site, please fill in the form below.

All of your comments and suggestions will be the most welcome.

Name

Other Names

E-mail

Phone

Address

City

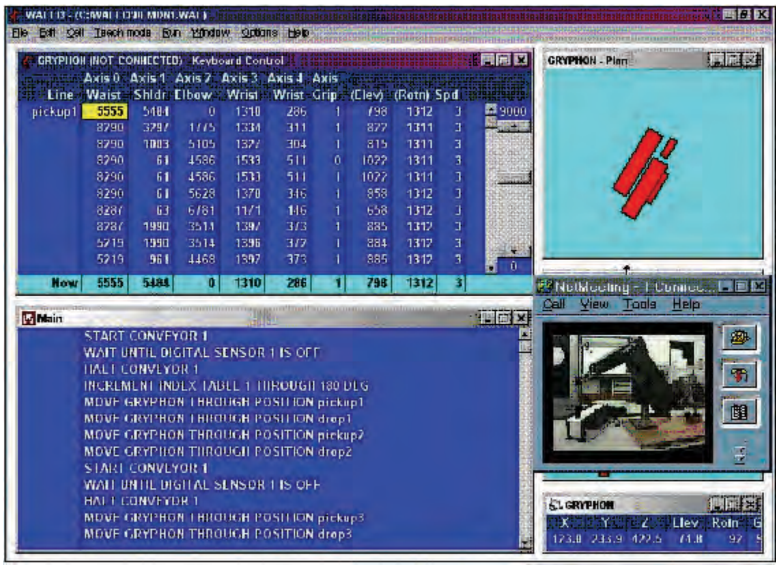
Country

Enter your comments in the space provided below:

Fig. 18. Comments Form.

7. Using the Web-Based Interface System

For the time being, the web site is only accessible on the University network and hence can only be viewed by any user who logs in the University intranet. The client's browser will generate the Homepage and then the user may travel along the site as any other one. The various pages mentioned before will always be available and the user may select the existing links and the program generates the request for each task. Whenever a user selects the Sign In page, the user is prompted to enter a valid user ID and a password. The server then processes the query of the user and if the ID and the password are missing in the database, logging is unsuccessful and the user is redirected to the initial login page. Now, if the ID and password are valid, another web page is displayed. In this page, the user is given the instruction on how to connect to the server where the Gryphon Robot is connected. An IP address, together with the associated password for the VNC is then given to the visitor. To use the VNC technology the user requires a VNCviewer, offered as download, and after running the application, it requires the given IP address and password for proper connection to the server. In addition to this tool, the new web page gives a direct connection for the video transmission. By simply clicking the link, video images of the robot environment will be readily available. After connection, the user may then view and control the movements of the Gryphon Robot remotely as shown in Fig. 19.


WALLI3 - (C:\WALLI3\LMON1.WAL)
File Edit Cell Teach mode Run Window Options Help
GRYPHON (NOT CONNECTED) Keyboard Control
9000
0
Main
GRYPHON - Plan
NetMeeting - 1 Connec...
Call View Tools Help
GRYPHON

	Axis 0	Axis 1	Axis 2	Axis 3	Axis 4	Axis			
Line	Waist	Shldr	Elbow	Wrist	Wrist	Grip	(Elev)	(Rotn)	Spd
pickup1	5555	5484	0	1310	286	1	798	1312	3
	8790	3797	1775	1334	311	1	877	1311	3
	8790	1003	5105	1377	304	1	815	1311	3
	8290	61	4586	1533	511	0	1022	1311	3
	8790	61	4586	1533	511	1	1022	1311	3
	8290	61	5628	1370	346	1	858	1312	3
	8287	63	6781	1171	146	1	658	1312	3
	8787	1990	3511	1397	373	1	885	1312	3
	5719	1990	3514	1396	377	1	884	1312	3
	5219	961	4468	1397	373	1	885	1312	3
Now	5555	5484	0	1310	286	1	798	1312	3

```
START CONVEYOR 1
WAIT UNTIL DIGITAL SENSOR 1 IS OFF
HALT CONVEYOR 1
INCREMENT INDEX TABLE 1 THROUGH 180 DEG
MOVE GRYPHON THROUGH POSITION pickup1
MOVE GRYPHON THROUGH POSITION drop1
MOVE GRYPHON THROUGH POSITION pickup2
MOVE GRYPHON THROUGH POSITION drop2
START CONVEYOR 1
WAIT UNTIL DIGITAL SENSOR 1 IS OFF
HALT CONVEYOR 1
MOVE GRYPHON THROUGH POSITION pickup3
MOVE GRYPHON THROUGH POSITION drop3
```

X	Y	Z	Elev	Rotn	G
173.8	233.9	422.5	71.8	97	5

Fig. 19. Remote Access.

7.1 Sample Program for Remote Control

A sample program with axis data entered remotely using WALLI3 are given in Fig. 20 and Table 1 respectively. This program demonstrates how the Gryphon robot can be operated using keyboard control mode. The program and table describe the Robot program which coordinates various devices and a sub-device. The overall system consists of a conveyor belt (Conveyor 1), an index table (Index table 1), a digital sensor (Sensor 1) and finally the Gryphon robot itself. The various axes are positioned in an orderly manner whereby the robot can smoothly perform a scenario.

```
MAIN PROGRAM – EXECUTING ONLINE
1 START CONVEYOR 1
2 IF DIGITAL SENSOR 1 IS ON THEN GOTO 3
3 HALT CONVEYOR 1
4 WAIT 5 SECONDS
5 MOVE GRYPHON THROUGH POSITION Pickup 1
6 INCREMENT INDEX TABLE 1 THROUGH 90 DEG
7 MOVE GRYPHON THROUGH POSITION Pickup 2
OK
```

Fig. 20. Sample Program.

	Axis 0	*Axis 1*	*Axis 2*	*Axis 3*	*Axis 4*	*Axis 5*			
Line	**Waist**	**Shldr**	**Elbow**	**Wrist**	**Wrist**	**Grip**	**(Elev)**	**(Rot)**	**Spd**
Start	4500	4950	790	790	790	1	790	800	3
Pickup 1	6500	240	5100	1410	370	0	890	1320	3
	6500	240	5100	1410	370	1	890	1320	3
	6500	1665	3950	1375	345	1	860	1315	3
	5000	3120	1900	1460	430	1	945	1315	3
	5000	850	3550	1550	500	1	1025	1325	3
	5000	850	3550	1550	500	0	1025	1325	3
	5000	2375	3500	1375	325	0	850	1325	3
	5000	3280	2015	1125	75	0	600	1325	3
Pickup 2	4400	1790	2480	1180	140	0	660	1320	3
	4400	1790	2480	1180	140	1	660	1320	3
	4400	2800	2500	1180	140	1	660	1320	3
	3200	2800	2500	1180	140	1	660	1320	3
	3200	1600	1450	1320	280	1	800	1320	3
	3200	1600	1450	1320	280	0	800	1320	1
End	3200	5500	0	1320	280	0	800	1320	1

Table 1. Keyboard Control Data.

The starting position (Start) of the robot may be defined by the user. The coordinates used may be as per the user's discretion and should eventually be the starting position of the robot. As soon as the program is run, the Main Program is executed. The first line of code clearly directs the conveyor to move. Then, as soon as an object hits the digital sensor, the conveyor automatically stops since the sensor should read ON. The main program is then redirected to the third line of codes, which is stopping the conveyor. The program continues running and the system now receives the order to wait for 5 seconds. All the mnemonics for the main program are built in the WALLI3 software. The user has just to feed in appropriate data in order to execute a program. After 5 seconds delay, the main program directs the system to the subroutine which includes the coordinates and movement of the Gryphon robot. The first movement or task that the Gryphon robot should perform is picking the object from the conveyor belt. The subroutine PICKUP 1 describes this task. The robot's arm moves towards the chosen object and grips it. Then, the arm moves away and put down the object on the index table. As soon as the PICKUP 1 subroutine is completed, the system is forwarded to the main program. The next instruction from the software is moving the index table through 90 degrees. This table can continuously rotate and digital sensors may be used as for the conveyor belt. However, in this context, the index table has been ordered to move through only 90 degrees. As soon as the index table has completed its task, the program is again redirected to the Gryphon subroutine PICKUP 2. Its function is to pick up the object on the index table and place it in a safe position as required by the user. Speeds varying between 1-3 may be used to move the robot, and 3 is the fastest. The robot has been programmed to dispose of the object slowly in a safe position. As the task of the subroutine is completed, the program executes the final line of codes and hence the robot moves to a safe position as determined by the user.

The remote user has the flexibility of changing the robot parameters and experiment with the program again to view how the robot can be effectively controlled and in the process learn the basics of robot programming. To end the session, the remote user simply closes the VNC and signs out. The user will be then directed to the login page, after which it can be linked to the rest of the site. At the closing stage of the site, a Comments Form has been included in order to enable further interaction with any user. The form is available upon request and the user has to fill all the available fields. On clicking the Submit button, all the fields are validated on the client side itself. If there are any missing or incorrect formats of information, the user will be prompted accordingly. If now, all the fields are correctly filled, connection is opened to the database and the data is stored. To confirm acceptance of the records by the database, a feedback is supplied to the user. All the records are stored on the same server as the mechanical device.

8. Conclusions

An introductory web site on robotics, with remote control of the Gryphon robot via the World Wide Web, has been successfully designed and implemented. This chapter has described the development of the web-based interface using the tools and technologies currently available. The web site gives concise and up-to-date facts about robots and genuine industries which employs them and it acts as a teaching tool for all those who have a particular interest in robotics. Moreover, by including the control of a mechanical device in the system, a remote user can easily make the link between the theoretical approach and real life systems. The user can change the robot parameters remotely and experiment with them to learn its control. However, at this instance, the remote control of the robot is restricted to a single user, but priorities may be set for either a local or a remote control. Part of the system is suitable for Engineering courses such as Robotics Technology, which is being newly added in the Mechatronics course at the University of Mauritius. Further works and programs may be developed for other courses in the future, where the World Wide Web can help students enhance their knowledge about other disciplines.

9. References

AT&T (1999). http://www.uk.research.att.com/vnc/index.html

Italtec (1997). *WALLI for Windows*. Italtec Company, Italy, revised 10/03/1997.

Jones, A. R. (2000). *Mastering Active Server Pages 3*. BPB Publications, Delhi.

Khan, B. H. (1997). *Web-Based Instruction*. Educational Technology Publications, Englewood Cliffs.

Microsoft (2000). http://www.microsoft.com/windows/netmeeting/default.asp

Mootien, S. (2003). *A Web-based Interface for the Gryphon Ec Robot*. Final Year Project, B. Eng. (Hons.) Mechatronics, University of Mauritius, Mauritius.

Richardson, T.; Stafford-Fraser, Q.; Wood K. R. & Hopper A. (1998). Virtual network computing. *IEEE Internet Computing*, Vol. 2, No. 1, pp. 33–38.

You, S. (2001). A low-cost internet-based telerobotic system for access to remote laboratories. *Artificial Intelligence in Engineering*, Vol. 15, No. 3, pp. 265-279.

11

Internet-Based Service Robotic System Using CORBA as Communication Architecture

Songmin Jia, Kunikatsu Takase
University of Electro-Communications,
1-5-1 Chofugaoka, Chofu-City, Tokyo 182-8585, Japan

1. Introduction

Many Internet robotic systems have been developed on the Web over the last few years. Internet telerobotic systems have typically been operated using two ways communication links between a client and a remote robotic system (Barney Dalton et al, 1998), one is using HTTP (HyperText Transfer Protocol) combined with CGI (Common Gateway Interface) and the other way is using Java. The University of Southern California's Mercury Project (Ken Goldberg et al, 2000) used HTTP combined with CGI (Common Gateway Interface) to develop a tele-excavation system that enabled web users to manipulate a remote "real world". CMU project developed Xavier autonomous indoor robot on the Web using CGI (Reid Simmons et al, 1998, Reid Simmons et al, 2000). KhepOnTheWeb (Patrick Saucy et al, 1998, Patrick Saucy et al, 2000) uses CGI, which allows the user to control a Khepera robot in a static environment. Another CGI system is the WebPioneer (WebPioneer, 1998), in which the user drives a Pioneer robot in an unknown room. CGI system causes latency and poor scalability because a new process must be started for each user request. The USA Puma Paint project (Matthew R. Stein et al, 1998, Matthew R. Stein, 2000) is a Web site allowing any user with a Java ™ compatible Web browser to control a PUMA760 robot to paint on an easel with real brushes and paint. Mobile robot on the Web project (Roland Siegwart et al, 1998) uses Java to establish a modular framework for mobile robots on the Web. Java is executable within a Web page. Java programs can be run as applets within the browser and supply an interactive interface instead of a static one. But Java client must know the location of all servers to which it wants to connect and Java allows applets to connect only to the host they were served from. In this paper, we propose using CORBA to implement networking connections between a client and a remote robotic system.
Our goal was to propose the advanced architecture to implement the Internet robotic system that can overcome the shortcomings of the other Internet robotic systems. CORBA uses an Object Request Broker (ORB) as the middleware that establishes client/server relationships between objects. This allows a client to invoke a method on a server across network transparently without needing to know where the application servers are located, or what programming language and operating system are used.
Our secondary goal was to implement the network-based robotic system to assist the aged or disabled in their homes because the elderly population is growing while the number of people to take care of them is declining. The primary task of the system would be to supply food and a cup of water or juice to aged and disabled persons. To

accomplish these tasks, the human caregivers can control the telerobotic system to retrieve and manipulate the desired tableware or the other objects, by viewing live image feedback from a remote computer. In our research, we first implement an Internet-based hand-eye robotic system using CORBA as a basic platform. The user can control the remote hand-eye robotic system at a task-level to recognize and manipulate the spoon, the fork, and so on by the intuitive user interface. Then we developed multi-robots to perform service tasks for supporting the aged or disabled. A video/audio conference system based on Robot Technology Middleware (RTM) was also developed to improve the interaction among the users and local robotic system, and enable web-user to monitor the state of robotic system working and the state of the aged or disabled, and to get a better understanding of what is going on in the local environment.
In summary, the paper contributions are:

- Propose using Common Object Request Broker Architecture (CORBA) to implement networking connections between a client and a remote robotic system. This makes the system has the distinct features as following:
  - Clients and servers do not need direct knowledge of each other; CORBA-based system has high scalability to allow clients and server objects, for example written in different languages, run in different operating system, and connected in different network to inter-operate.
  - The CORBA-based Internet robotic system can be implemented by a number of independent components, and these components of the system can be run independently to implement the application and easily be integrated with the other technologies into new comprehensive application systems. This facilitates network-distributed software sharing and improves the cost of writing and maintaining software.
  - The other components of the system can work normally even if there are problems with some of them.
- Develop a CORBA-based Internet hand-eye robotic system and CORBA application servers to allow the remote user to control the robot arm to recognize and manipulate the objects at a task-level.
- Develop multi-functional services robotic system and key technologies to enable the developed system to provide daily service task to support the aged or disabled.
- Experimental results verified that using CORBA to implement networking connections between a client and a remote robotic system is very flexible, effective and robust.

The rest of the paper consists of 7 sections. Section 2 presents Common Object Request Broker (CORBA) in detail. Section 3 describes the structure of the system hardware base. Section 4 details the functions and interfaces of the task-level robot arm control server, live image feedback server, mobile robot control server, and iGPS server. Section 5 introduces the developed network monitoring system based on RT middleware. Section 6 explains intuitive Web-based user interface. Experimental results are given in Section 7. Section 8 concludes the paper.

2. Common Object Request Broker Architecture (CORBA)

In our aging society, it would seem particularly important to integrate various kinds of network-distributed software and robotic systems for complicated applications aiding the elderly population. A distributed computing technology that can implement network-distributed software sharing and improve the cost of writing and maintaining software is in high demand.

CORBA combines the benefits of object-orientation and distributed computing and is a middle-ware architecture that provides an object bus that allows components/objects to communicate with each other across networks (Seán Baker, 1997). There are the others distributed computing technologies, like RMI (Remote Method Invocation), DCOM (Distributed Component Object Model), MOM (Messages Oriented Middleware). Sun's Java RMI (Java) is a Java-specific RPC middleware that provides a simple and direct model for distributed computation with Java objects. However, its simplicity is enabled by restricting all communication to Java objects only. DCOM is Microsoft's architecture for distributed component-based communication and is based on COM, which is both a specification and implementation developed by Microsoft Corporation. MOM (mom) provides process-to-process data exchange, enabling the creation of distributed applications. It is analogous to e-mail in the sense that it is asynchronous, requiring the recipients of messages to interpret their meaning and to take appropriate action. MOM is not an industry standard.

In contrast to all of these, CORBA uses an Object Request Broker (ORB) as the middleware that establishes a client-server relationship between objects, and it is an object-oriented extension of Remote Procedure Calls (RPCs). CORBA uses GIOPs (General Inter-ORB Protocols), ESIOPs (Environment Specific Inter-ORB Protocols) and IIOP (Internet Inter-ORB Protocols) to implement a truly heterogeneous distributed system. This heterogeneity enables CORBA to inter-operate ORBs purchased from different vendors and supported on different platforms.

The main components of CORBA are shown in Figure 1.

- The Object Request Broker (ORB core) is responsible for the delivery of client-requests from one computer to objects possibly on the other computer as well as for the return of the resulting information. By hiding all details of data transmission from the client, it facilitates the communication between clients and objects. To target an object, the client only has to specify the object reference, which is created when the object is created.
- OMG Interface Definition Language (OMG IDL) defines the object interface and specifies the types and operations that the objects support. IDL has features like language independence, distributed service specification, definition of complex data types, and hardware independence. By compiling this interface with an IDL compiler for mapping to client/server source code in a specific programming language, we can get client stub and server skeleton source code for communication between client and server (SJT Condie, 1999).
- Object Adapter (OA) provides a binding between an objects interface and a server's implementation. Object Adapter is responsible for the registration of an object at the ORB and the creation of the object reference. The most commonly used OA is the basic object adapter (BOA), which all CORBA vendors must supply.
- Dynamic Invocation/Skeleton Interface (DII/DSI) allows an application to issue requests for any interface, even to an interface that was unknown at the time when the application was compiled.

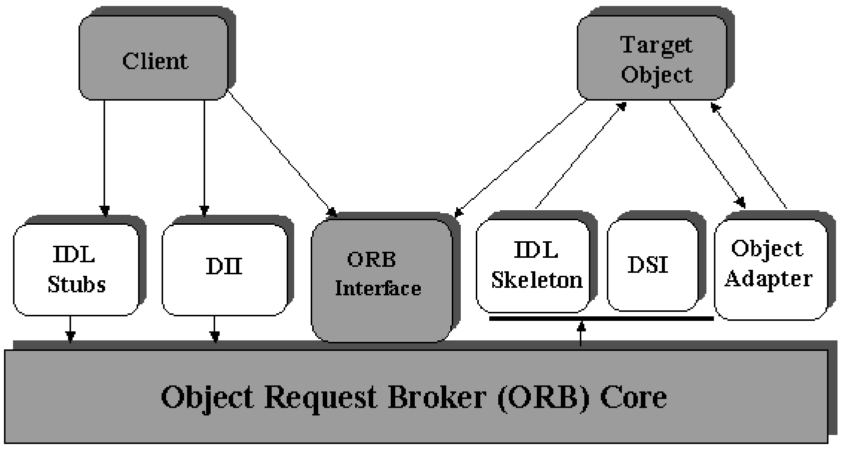


Fig. 1. Common Object Request Broker Architecture.

3. System Description

The population-aging problem is increasingly pressing society. According to statistical data from the Ministry of Welfare of Japan, those 65 years or older will comprise 25.2% of the Japanese population in 2015. Meanwhile, the population of the aged worldwide is increasing at approximately a 12% rate. This group requires special care and functional aids due to the progressively reduced degree of autonomy. Because of rising costs and the shortage of nursing home personnel, the current trends for long-term care for the elderly are at-home care or residence at small-scale distributed facilities. Thus, the development of supporting systems to improve care cost and the QoL (Quality of Life) of the elderly could not come at a better time.

Our HARSP (Human-Assistance Robotic System Project) project consists on developing a Human-Assistance Robotic Distributed System in order to provide daily service to aid aged or the elderly people (Jia et al, 2004). The proposed system has the potential to provide elderly persons local services in:

- Intelligent reminding: remind elderly persons about important activities such as taking medical, eating meals, visiting the bathroom, or scheduling medical appointments.
- Data collection and surveillance: many emergency conditions can be avoided with systematic data collection. For instance, robots can support the aged or disabled in using medical peripherals, such as a telephonic stethoscope, vital sign and glucose meters. Robotic systems may be soon able to call caregivers for assistance if they detect that an elderly person has fallen or the other emergency.
- Daily services: many elderly persons with some chronic disease give up independent living because of difficulty in moving their body to take something such as operating objects in a refrigerators. An omnidirectional mobile robot integrating with a skillful robot arm could help the aged or disabled overcome these barriers and supply them necessary daily services.
- Mobility aid: support the aged or disabled for getting up from the bed or a chair, and implement intelligent walking aid.

Multi-robot cooperation is indispensable in order to perform service tasks for supporting the aged or disabled. The mobile platform equipped with a dexterous manipulator is convenient, but it is very difficult to handle the objects (such as operating objects in a refrigerators) because of the difficulty to control the position and orientation of the mobile platform and the manipulator mounted on the mobile platform. In our system, we adopted using a robot arm with five degrees of freedoms cooperating with a mobile robot to implement the service tasks. This method makes it easier to operate the objects such as in a refrigerators. Figure 2 illustrates the hardware configuration of the developed system including robots, cameras, server computer and users' computers.

3.1 Robot Arm and Hand

The Mitsubishi Movemaster Super RV-E3J was used in this system. It consists of a manipulator with five degrees of freedoms, a controller, and a teaching box. The maximum speed of the robot arm is about 3500mm/sec; its load weight is about 3.5kgf. It is fixed to a place where there are many objects that need to be manipulated.
In order to implement soft grasp, the robot hand with force sensors has been designed. We also designed the CPU control circuit to measure the grasp force and the servo-driving circuit to drive the fingers. Futaba's S3801 high-torque metal gear servo was used to drive the fingers. Its weight is about 107g, its operating speed is 231°/sec, and its output torque is 14.0 Kg·cm. This servo can act on PWM (Pulse Width Modulation). The basic cycle is 20 ms. We generate input control pulses by a PTC (Programmable Timer Controller) to control the servo. The robot hand works according to the commands coming from the CPU controlled by the server computer (Jia et al, 2001).

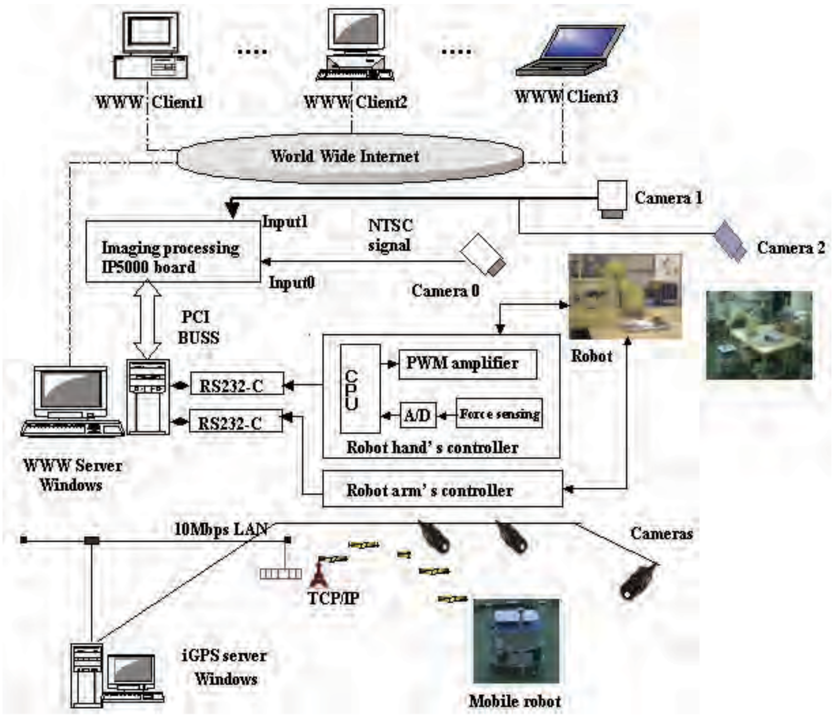


Fig. 2. Hardware base of the developed system.

3.2 Vision System

VC-CI MKII cameras are positioned at the place where they can provide user feedback images easy to understand, the other VC-CI MKII camera is used to take an image of the tableware or the other things scattered on the table in this system. A high-speed image processing board, IP5000, was used for processing the obtained images. The resolution of the vision system is 512 x 512 pixels, and it takes 3.6 milliseconds to process each image. For localizing the mobile robot, SONY XC-55 TV cameras with IR ray pass filters are mounted on the ceiling of the environment in which the mobile robot moves . The SHARP GPB-K image processing board processes images and localizes the mobile robot by detecting the IR LED units arranged on the mobile robot (Hada, 2004).

3.3 Mobile Robot

An omnidirectional mobile robot was used to deliver pills, juice or other objects to the place where the disabled or aged person lies or sits. It has a diameter of 45cm and stands 66cm tall. It can translate at speeds up to 300 mm and rotate at a speed up to 30 degrees per second. The mobile robot includes a three-wheel mobile base that can move in all directions, a control system to control the motion of the mobile robot and a network interface which supports a wireless TCP/IP Ethernet link (Hada, 2004). To cope with objects vibrating on the tray when the mobile robot is moving, special nonskid materials were used for the tray surface.

3.4 Server Computers and User's Computer

The server computer receives client's commands from worldwide network, does various kinds of processing and then sends commands to the devices that can implement the tasks. A high-performance computer is necessary if we want to implement a good-performance system.
For this system, linking to the Internet is the only requirement on user's computer. World Wide Web users can use any computer platform and operating system.
The communications between server and robot arm's controller and the robot hand's controller are through RS232-C links. Users' computers in the world communicate with the server via world wide Internet.

4. CORBA Application Servers

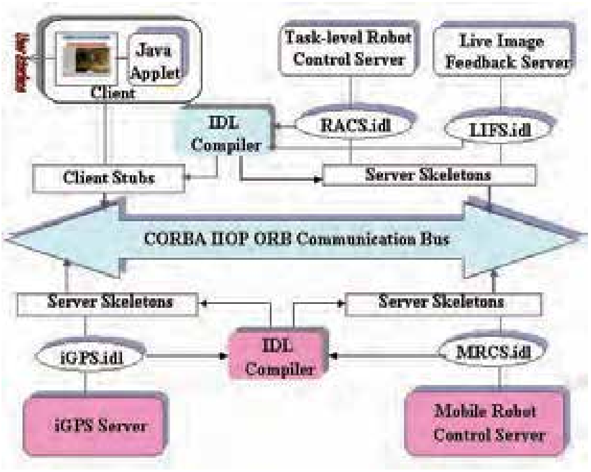
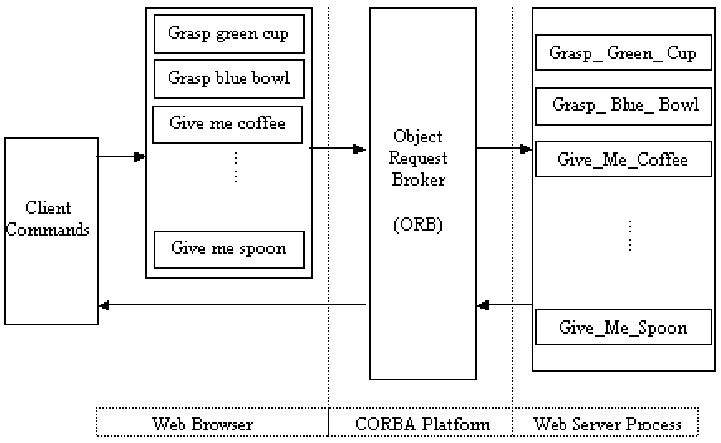


Fig. 3. Network connections of CORBA-based Internet robotic system.

We use CORBA as communication architecture to implement an Internet robotic system. We have implemented task-level robot control server, mobile robot control server and live image feedback server by the optimal programming language C++ because C++ is the most proper language for the core functions of the system, such as imaging processing, robot control. We have also implemented a client by a Java applet because Java applets are capable of directly accessing CORBA application servers via the Internet Inter-Object Request Broker (IIOP) and are easy to run within a Web browser that can be used by non-specialists from any Internet site. The network connections of CORBA-based Internet service robotic system are shown in Figure 3.

4.1 Task-Level Robot Control Server

Task-level robot control server allows the caregivers to control this remote robotic system at a task-level (Jia et al, 2002). The user can invoke the methods on this application server across the Internet to recognize and manipulate the tableware or the other object what they want. The assignments of the server is as follows (Figure 4):



Interface Name	Functions
Handle_ Cup	Provide service for manipulation of cups
Handle_ Bowl	Provide service for manipulation of bowls
Handle_Can	Provide service for manipulation of cans
⋮	
Handle_ Spoon	Provide service for manipulation of spoons

Fig. 4. Robot control server and its interface.

- Receive commands from the remote World Wide Web users.
- ORB intercepts the "call" and is responsible for finding an object that can implement the request, passes it the commands.
- Find the location and orientation of the tableware or the other objects scattered on the table by the vision system.
- Generate the task implementation plan and send the commands to the devices.
- ORB returns the feedback results to the remote clients.

For one method on the task-level robot control server, it consists of the following parts (Figure 5).

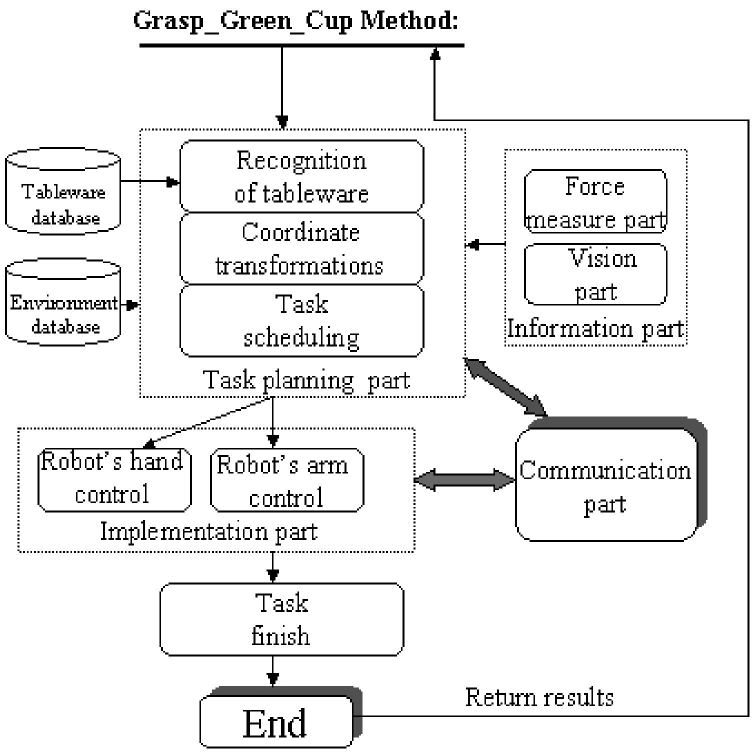


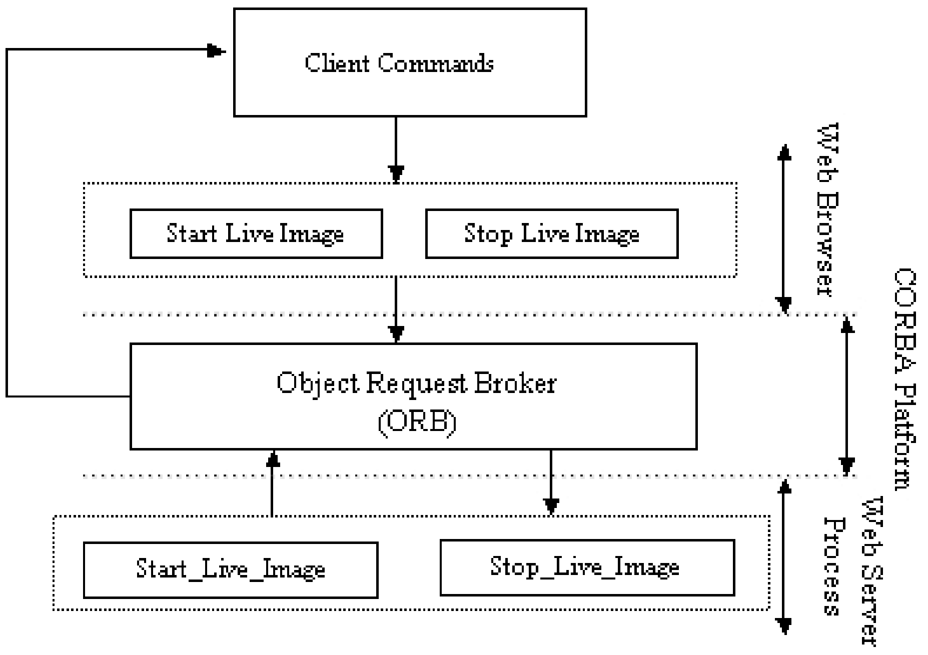
Fig. 5. Method of robot control server.

- The information part consists of a vision part and a force measure part. It is the source of information for the system.
- The task planning part receives the information from the information part, recognizes the location and orientation of the tableware or the other objects scattered on the table, transforms these coordinates to the manipulator's coordinate system, and generates task plan to achieve the goal.
- The implementation part executes motion scheduling generated by the task planning part, and it implements the task according to the commands coming from the server computer.
- The communication part is responsible for the communication between the server computer, the robot's arm and the robot hand's controller via RS232-C links.

For real-time recognition of circular tableware, the quick correlation calculation was used. Correlation calculation is a method that evaluates the matching between the template image registered beforehand and the input image obtained from the camera.
For long tableware or long objects such as spoons and forks, we assume that they consist of two different colour parts. Using this technique, long tableware or long objects recognition can be replaced by the simple mark recognition because there is one-to-one correspondence between tableware or objects and different combination of colour marks. We can create the tableware 's or the other objects' geometric model and entry them beforehand. By doing this we can recognize the tableware or the other objects easily and quickly.

4.2 Live Image Feedback Server

Live image feedback server provides live image feedback from the camera for the remote user and continuously sends live images of the working robot to the client according to the user's requests (Figure 6). It works as:



Interface Name	Functions
Control_Live_Feedback_Image	Provide live feedback image service

Fig. 6. Live image feedback server and its interface.

- Receive image control commands from the remote users.
- ORB intercepts the commands and transfers the requests to live image feedback server.
- Get the new image by camera0 and change the image into a BMP format by IP5000 image processing board.
- Compress this image into JPEG format.
- Return the new image with the last information to the client by ORB.

4.3 Mobile Robot Control Server and iGPS Server

In order to realize quick response to a request from a client, we implemented a distributed iGPS server run on another computer (iGPS server computer, connected in 100BASE-TX LAN, running Windows) to share the calculation of the position and orientation of the mobile robot.
The mobile robot control server allows the remote user to specify the best route for the mobile robot and control the mobile robot when mobile robot execute a service task in the real environment. It works as follows:

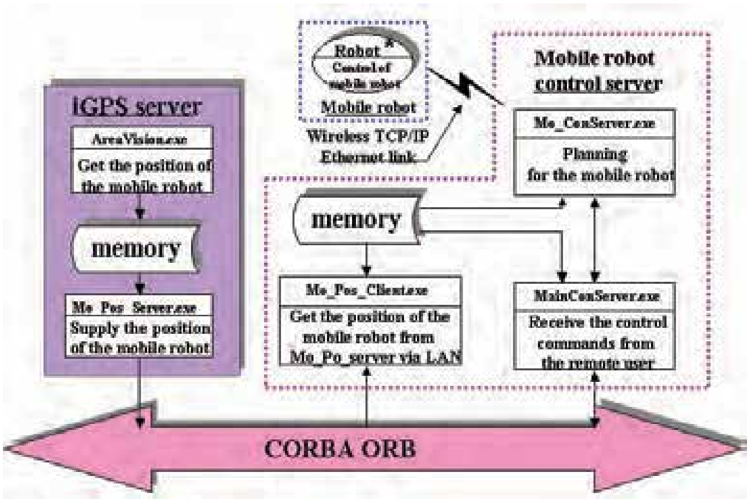

Robot
Control of mobile robot
Mobile robot
Wireless TCP/IP Ethernet link
Mobile robot control server
Mo_ConServer.exe
Planning for the mobile robot
iGPS server
AreaVision.exe
Get the position of the mobile robot
memory
Mo_Pos_Server.exe
Supply the position of the mobile robot
memory
Mo_Pos_Client.exe
Get the position of the mobile robot from Mo_Po_server via LAN
MainConServer.exe
Receive the control commands from the remote user
CORBA ORB

Fig. 7. Mobile robot server, iGPS server and their configuration.

- The mobile robot control server receives control commands from the remote World Wide Web users.
- ORB intercepts commands and transfers the requests to the mobile robot control server.
- The mobile robot control server plans and derives a collision-free path that is the shortest distance between the START and GOAL specified by the user.
- The mobile robot control server programs the omnidirectional mobile robot to move across wireless TCP/IP Ethernet link.
- ORB returns the feedback results to the remote clients.

Figure 7 illustrates Mobile robot server, iGPS server and their configuration. By viewing live feedback images the user can determine whether the mobile robot has arrived at a place where it can easily receive an object from the robot arm or pass the object to the disabled person. If there is displacement, the user can adjust the position of the mobile robot by fine adjustment commands using interface. Similarly, the remote user can also rotate the mobile robot to a certain angle.

5. Network Monitoring System Based on RT Middleware

In order to enable a remote users to get a better understanding of the state of the robots working and the state of the aged or disabled, we developed network distributed monitoring system to transmit media streams in real time to improve interaction in network distributed human-assistance robotic system. Network monitoring system using QuickCam Orbit cameras was implemented based on RTM. RT Middleware framework was developed by AIST (Agency of Industrial Science and Technology, Japan) to promote application of Robot Technology (RT) in various fields. RTM is based on CORBA (ominiORB), so the components of the system can be implemented by different programming languages, run in different operating system, or connected in different networks to inter-operate. Each application is made up of components; integration is supported by allowing other applications to communicate directly with these components. This facilitates network-distributed software sharing and improves the cost of writing and maintaining software.
RTM used OminORB 4.0.5 to implement framework. omniORB is a robust, high-performance CORBA 2 ORB, developed by AT & T. It is one of only three ORBs to be awarded the Open

Group's Open Brand for CORBA. This means that omniORB has been tested and certified CORBA 2.1 compliant. omniORB implements the specification 2.3 of the Common Object Request Broker Architecture (CORBA). We develop robotic functional elements as "RT software component", which makes application and system integration easier.

The QuickCam Orbit cameras were used in our system with high-quality videos at true CCD 640x480 resolution and automatic face tracking and mechanical Pan, Tilt feature. It mechanically and automatically turns left and right for almost a 180-degree horizontal view or up and down for almost 90-degree top-to-bottom view. Its maximum video frame rate is 30 fps (frames per second) and works with both USB 2.0 and 1.1. We implemented video stream RT component based on RT Middleware, and OmniCORBA IIOP is employed as message communication protocol between RT component and requester. Two QuickCam Orbit cameras were set up in the environment, and they can overlook the area the service robots moves in by adjusting the mechanical Pan and Tilt of the cameras. The structure of the developed network monitoring system was shown in Figure 8.

In addition, we developed a graphic user interface (GUI) for the video stream system that provides a remote video stream (format 320x288 was selected because image data transition's problem), a camera zoom and pan-tilt adjustment, and a chat function that allows a remote user to communicate with a local user. When the user sends a request for video, the system will autonomously display the GUI. The user can click "Connect" and input the IP address of the computer on which the RT video component is running to view a real-time video feed. The RT video stream component was implemented by the Visual C++, Microsoft visual studio.net 2003. A performance test of the developed real-time video stream was conducted to examine the possibility of using a live video feed to monitor the state of robotic system. The video server is run on Windows 2000 Professional (1.9GHz, Pentium4), and the video client is run on Windows XP (2.4GHz, Pentium4). The average frame rate is approximately 16.5fps.

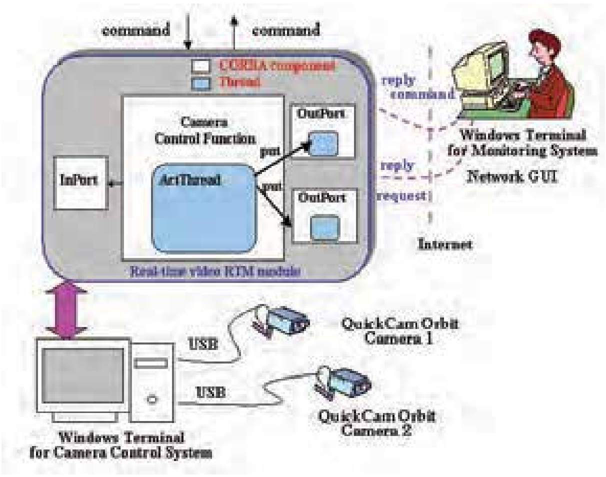


Fig. 8. Structure of network monitoring system based RTM.

6. Intuitive Web-Based User Interface

The challenge of designing a user interface is to provide enough information to let the user handle this system easily while minimizing transmitted data and maintaining a simple layout. Besides that, all kinds of complex calculations should be hidden from the user in order to support a wide range of users. The Web-based user interface designed is shown in Figure 9.

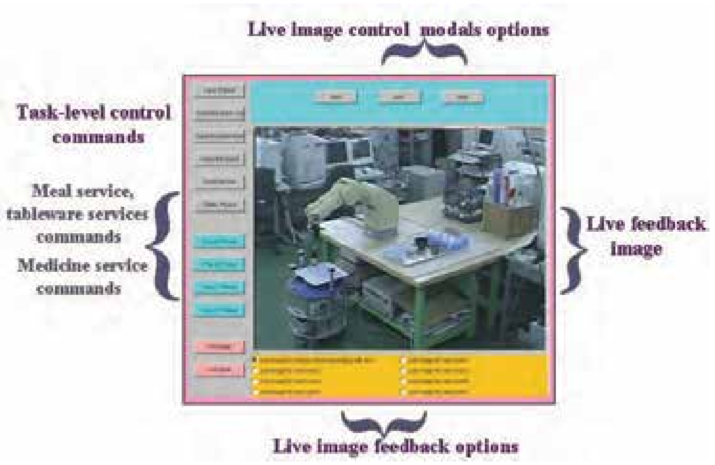

Live image control modals options
Task-level control commands
Meal service, tableware services commands
Medicine service commands
Live feedback image
Live image feedback options

Fig. 9. Web-based user interface of the developed system.

User interface designed consists of live image feedback part, options for different kinds of live feedback images part, and task-level robot arm control commands part. The task-level robot arm control commands allow the user to submit a task-level request to the system. It consists of meal service commands, drug service commands, and a common service command such as providing a book. These commands include ``Grasp the green cup," "Grasp the spoon," Juice, please" etc. Once one task-level button is pushed, the other task-level button will be invalid until this task is finished for safety. When the user push the button "grasp the spoon", this command will be sent to server and the necessary result message will be feedback to the client by popping one message text box on the Web-based user interface. Many options for different kinds of live feedback images have been provided, including images of the mobile robot cooperating with the manipulator, images of the state of the rooms of the disabled or aged. In addition, "auto" and "step" control modals of the live image feedback are provided to allow the user to see the continuous live feedback images or the "step" image feedback that refreshes the image once button is pushed.
When the remote user links to this homepage, a geometric 2D map models the environment of the robot system's working domain. Its geometric primitives can also be adjusted to add new obstacles if the environment of the robot system has been changed. The positions where the mobile robot can easily cooperate with the robot arm and pass objects to the disabled or aged are displayed as marks. The "Remove last point" button's function is to remove the last command that the user specified if the user wants to change the route. The "Remove all points" button allows the user to clear all the commands that the user specified and renew the route. In order to know the real position of the mobile robot in the indoor environment, the real trajectory of the mobile robot is also shown on the interface homepage. The "Up", "Down", "Left", and "Right" buttons allow the user to adjust the position of the mobile robot to facilitate cooperation with the robot arm. The "Rotate" button can also be used to control the mobile robot to rotate.

7. Experimental Results

- Performance Evaluation of CORBA

CORBA uses an Object Request Broker (ORB) as the middleware that establishes client/server relationships between objects. The client can invoke a method on a server object across a network transparently without knowing where the application servers are

located, or what programming language and operating system are used. This lets system overcome the shortcomings of the other Internet robotic systems. In addition, the components of the CORBA-based system can be implemented and run independently to implement the application, and also be integrated easily into new systems.
The experiments of evaluating the performance CORBA have been done. The client located at PC1 Pentium III, 600 MHz, and NT 4.0) sends a remote method call with 64 long words (4 bytes/word) to CORBA server (located at PC2, Pentium III, 700 MHz, NT 4.0) in a LAN (100BASE-TX), and the CORBA server returns the identical data to the client. This cycle is repeated 1000 times, which is called one set.Twenty sets have been done and the best performance is chosen to be the representativeresults from 20 sets, and its average of the round-trip time is about 0.29ms.The same experiment has also been done with ten simultaneous users, and there is no significant time difference between the result of only single user and that of multi-user.
We also have done the experiments such as: the task-level robot control server is working and the live image feedback server was down. In such a case, the user can still access the robot control server normally. Similarly, the user can still access the live image feedback server normallyeven though the other robot control server was down.

- Remote Accesses to the Developed Hand-Eye Robotic System

We useCORBA as a basic platform and connect our hand-eye robotic system to the Internet. The Web-based user interface was designed and it consists of live image feedback part, robot commands part, and message text box. This system has been accessed by the operators from Tama City, Saitama City, Kyoto University, Japan and Chicago, America etc. These users' computershave different specifications and network connections, and work on the different operating systems. Users operate this remote robotic system toretrieve and manipulate the table ware that they want and they can also monitor the state of the robot executing the task by viewingthe live image feedback coming from the robot site.

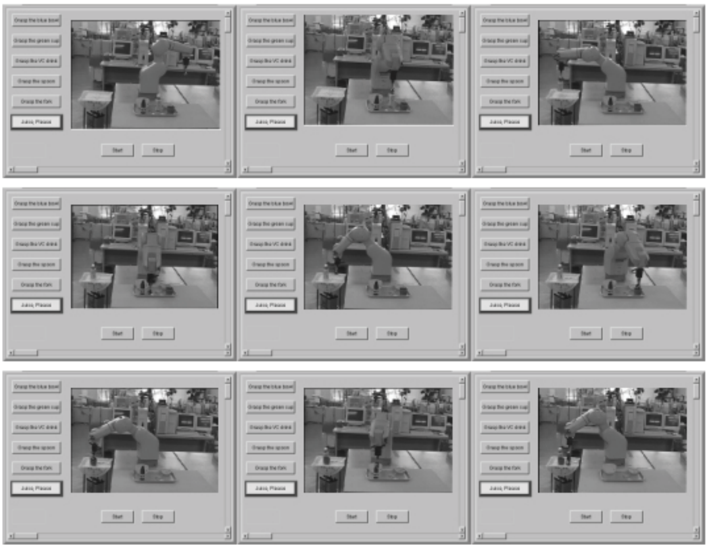


Fig. 10. Remote access to the hand-eye system; The remote user uses the telerobot system to manipulate the juice, fork, green cup and vitamin drink and put them to the tray mounted on the mobile platform.

When the remote user pushes the command "Juice, Please", the system can automatically find the juice and execute this task robustly. Figure 10 shows some on-line images that the remote user is accessing the developed hand-eye system to manipulate the fork; green cup and vitamin drink and put them to the tray mounted on the mobile platform. When the users pushing the "start" button, the new image with the latest information will be shown.

- Remote access to the developed multi-robots system to perform a service task.

Figure 11 illustrates some on-line images of the user operating the multi-robots to perform a service task. We developed a hardware base including a robot arm and an omnidirectional mobile robot and CORBA application servers including a task-level robot arm control server, live image feedback server, mobile robot control server and iGPS server, all of which can be distributed on the internet and executed in parallel.
By remotely controlling a mobile robot to cooperate with a robot arm using user interface designed, the developed system can realize some basic services to support the aged and disabled. When the operator links to the homepage, the operator first specifies the best route for the mobile robot and directs it to move to the place where it can cooperate with the robot arm by using the "specify mobile robot route" user interface. By live image feedback or video stream, the user can observe whether the mobile robot has arrived at a place where it can receive objects passed to it by the robot arm. If there is displacement, the user can adjust the position and rotation of the mobile robot. Then, the user operates the robot arm to manipulate the objects requested by the disabled or aged such as food, medicine or tableware. Lastly, the user can direct the mobile robot to move to the place where the disabled or aged person is. Of course, if the remote user can use the task-level control command, the developed system can perform a service task autonomously.

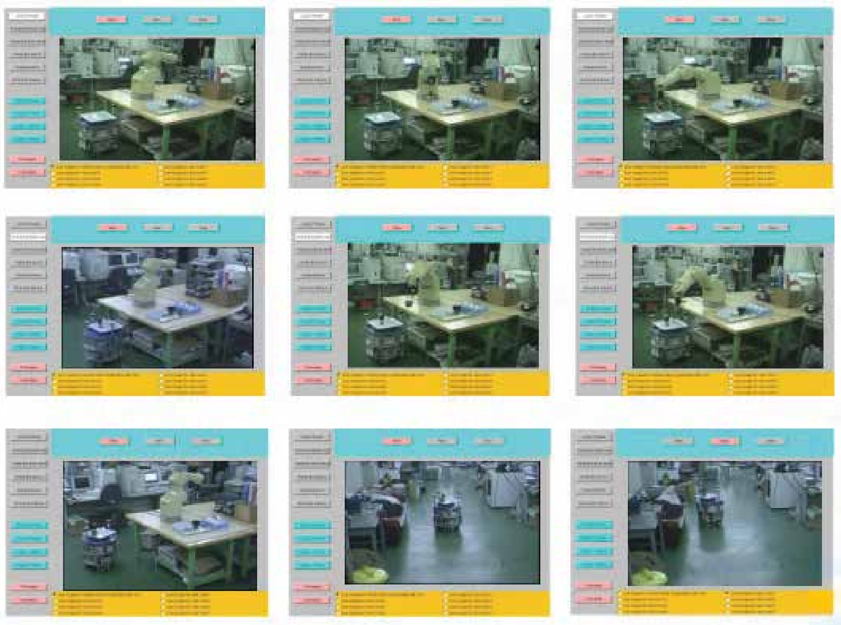


Fig. 11. On-line images remote user interacting with mobile robot to realize a local service task.

In order to enable a remote user to get a better understanding of the local environment, it is necessary to receive and transmit media streams in real time to improve interaction in network distributed service robotic system. Network monitoring system using QuickCam

Orbit cameras was developed to visualize the robotic system working, the state of the age and disabled, and to enable the remote user to know what is going on in local environment.

8. Conclusion

We used CORBA as communication architecture to implement networking connections between a client and the developed hand-eye robotic system. CORBA uses an Object Request Broker (ORB) as the middleware that establishes client/server relationships between objects. The client can invoke a method on a server object across a network transparently without knowing where the application servers are located, or what programming language and operating system are used. This lets system overcome the shortcomings of the other Internet robotic systems. In addition, the components of the CORBA-based system can be implemented and run independently to implement the application, and also be integrated easily into new systems. This facilitates network-distributed software sharing and improves the cost of writing and maintaining software. The other components of the system can work normally even if there are some problems with some of them. So, using CORBA to implement Internet robotic system is very effective, flexible and robust.
Using CORBA as communication architecture, we also developed a hardware base including a robot arm, omnidirectional mobile robot system and CORBA application servers including a robot arm control server, live feedback image server, mobile robot control server and iGPS server. By user controlling a mobile robot to cooperate with a robot arm using Web-based user interface, the developed system can provide some basic services to support the aged or disabled.
Network monitoring system based on RTM using QuickCam Orbit cameras was developed to transmit media streams in real time to improve interaction in network distributed human-assistance robotic system. It can enable remote users to get a better understanding of the state of the robots working and the state of the aged or disabled.
Some experiments that the remote user accesses the developed system to provide a service task to support the aged and the elderly people have been done. And experimental results verified the effectiveness of the developed system. For future work, improvements of the navigation of mobile robot and recognition of the obstacles using RFID technology and stereo camera are a main topic. In addition, we will develop the other additional function of the system as RT component in order to improve the flexibility of the system.

9. References

Barney Dalton and Ken Taylor (1998). A Framework for Internet Robotics, *Proceedings of 1998 IEEE/RSJ, Conference on Intelligent Robots and Systems; Workshop Web Robots*, pp. 15-23Victoria, B.C. Canada, October.

Hada, Y., Takase, K, Ohagaki, K. et al. (2004), Delivery Service Robot Using Distributed Acquisition, Actuators and Intelligence, *Proceeding of 2004 IEEE/RSJ International Conference on Intelligent Robots and System*, pp. 2997-3002.

Java remote method invocation:
http://java.sun.com/products/jdk/\ \rmi/index.html.

Ken Goldberg, Steven Gentner, Carl Sutter, et al. (2000), The Mercury Project: A Feasibility Study for Internet Robotics, *IEEE Robotics and Automation Magazine,* 7(1), pp. 35-40, ISSN.

Maja Matijašević, Kimon P. Valavanis, Denis Graćanin et al. (1999), Application of a Multi-User Distributed Virtual Environment Framework to Mobile Robot Teleoperation over the Internet, *Machine Intelligence and Robotic Control*, 1(1), 11-26.

Matthew R. Stein (2000). Interactive Internet Artistry, *IEEE Robotics and Automation Magazine*, 7(1), 28-32.

Matthew Stein (1998), Painting on the Web, The Puma Paint Project, *Proceedings of 1998 IEEE/RSJ,* pp.37-43, *Canada*, October 12-17.

MOM, Message-orientated middleware:
http://sims.berkeley.edu/courses/is206/f97/GroupB/mom/.

OMG, Object Management Group,
http://www.omg.org.

OOC, Object Oriented Concepts, Inc.,
http://www.omg.org.

Patrick Saucy, Francesco Mondada (2000), Open Access to a Mobile Robot on the Internet, *IEEE Robotics and Automation Magazine*, 7(1), 41-47.

Patric Saucy (1998), KhepOnTheWeb: One Year of Access to a Mobile Robot on the Internet, *Proceedings of IEEE/RSJ Conference on Intelligent Robots and Systems; Workshop on Web Robots*, pp. 23-31, Victoria, B.C. Canada, October.

Reid Simmons, Joaquin L. Fernandez, Richard Goodwin et al. (2000), Lessons Learned from Xavier, *IEEE Robotics and Automation Magazine*, 7(2), pp. 33-39, ISSN.

Reid Simmons: Xavier (1998), An Autonomous Mobile Robot on the Web*, Proceedings of IEEE/RSJ Conference on Intelligent Robots and Systems; Workshop on Web Robots,* pp. 43-49, Victoria, B.C. Canada.

Roland Siegwart, Cedric Wannaz, Patrick Garcia et al. (1998), Guiding Mobile Robots through the Web, Proceedings *of 1998 IEEE/RSJ Conference on Intelligent Robots and Systems,* Workshop on Web Robots, Victoria, B.C. Canada, October 12-17 1998, pp. 1-7.

RTM
http://www.is.aist.go.jp/rt/.

Seán Baker (1997), CORBA Distributed Objects Using Orbix, ACM Press, UK.

SJT Condie (1999), Distributed computing, tomorrow's panacea-an introduction to current technology, *BT Technol. J*, 17(2), pp.13-23.

Songmin Jia and Kunikatsu Takase (2001), Internet-based robotic system using CORBA as communication architecture, Journal *of Intelligent and Robotic System*, 34(2), pp. 121-134.

Songmin Jia, Yoshiro Hada, Gang Ye and Kunikatsu Takase (2002), Network Distributed Telecare Robotic Systems Using CORBA as Communication Architecture, *Proceeding of 2002 IEEE International Conference on Robotics and Automation*, pp. 2002-2007, USA

Songmin Jia, Weiguo Lin, Kaizhong Wang, Takafumi Abe, Erzhe Shang, and Kunikatsu Takase (2004), Improvements in Developed Human-Assisting Robotic System, *Proceedings. Of International Conference on Intelligent Mechatronics and Automation*, Invited paper, pp. 511-516.

WebPioneer Website, ActivMedia, Inc. (1998).
http://webpion.mobilerobots.com.

12

Path Coordination Planning and Control in Robotic Material Handling and Processing

Xuan F. Zha[1], Xiaoqi Chen[2]
[1]*National Institute of Standards and Technology & University of Maryland*
Gaithersburg, MD 20899
Email: zha@cme.nist.gov,xfzha@ieee.org
[2]*Singapore Institute of Manufacturing Technology*
71 Nanyang Drive, Singapore 638075
Email : xqchen@simtech.a-star.edu.sg

1. Abstract

This chapter presents a unified approach to coordination planning and control for robotic position and orientation trajectories in Cartesian space and its applications in robotic material handling and processing. The unified treatment of the end-effector positions and orientations is based on the robot pose ruled surface concept and used in trajectory interpolations. The focus of this chapter is on the determination and control of the instantaneous change laws of position and orientation, i.e., the generation and control of trajectories with good kinematics and dynamics performances along such trajectories. The coordination planning and control is implemented through controlling the motion laws of two end points of the orientation vector and calculating the coordinates of instantaneous corresponding points. The simulation and experiment in robotic surface profiling/finishing processes are presented to verify the feasibility of the proposed approach and demonstrate the capabilities of planning and control models.
Keywords: Robot pose ruled surface, Unified approach, Trajectory planning and control, Off-line programming, Robotics polishing

2. Introduction

Motion coordination planning and control play a crucial role in robot applications to Cartesian task operations with the consideration of kinematics and dynamics constraints. To effectively carry out the design and application of robots, an efficient algorithm for motion planning needs to be developed by generating, simulating and evaluating, and optimizing robot trajectories in a virtual CAD off-line programming environment (Zha 1993, Zha et al 1994, Zha and Du 2001). Many joint-space and Cartesian space based planning schemes have been made out by proposing new algorithms or improving the existing algorithms to describe and plan robot motions and trajectories (Thompson and Patel 1987; Bobrow et al 1985; Shin and McKay 1985; Pfeiffer and Johanni 1987; Yang and Jin 1988; Slotine and Yang 1989; Shiller and Lu 1990; Patel and Lin 1995; Konjovic and Vukobratovic 1994; Lee 1995;

Seereeram and Wen 1995). Some criteria are identified for comparing and evaluating both trajectories and trajectory planners of robot (Thompson and Patel 1987):

(1) Trajectories should be effective both to compute and to execute.
(2) Trajectories should be predictable and accurate, and they should not degenerate unacceptably near a singularity.
(3) The position, velocity, and acceleration, and even the rate of change of acceleration, called the jerk, should be smooth functions of time.
(4) Trajectory planner should be possible to determine efficiently whether a proposed trajectory requires the robot end effector to move to a point outside its workspace or move with a velocity or acceleration that is physically impossible. Both of these are controlled with a good model.

The coordination planning and control for robot motions in Cartesian space has long been recognized as a most interesting but difficult research field not only for motion control but also for advanced virtual/real design and planning. Some of the existing methods are, for the most part, based on kinematics considerations and geometric flavor. They may render to have limited capabilities for handling some cases where the limits of maximum acceleration and maximum deceleration along the solution curve are no longer met, or where singular points or critical points of robot configuration exist. Another problem with existing methods to plan trajectories is that the unmanageable complexity could occur both in computation cost and storage.

This chapter aims to present a unified approach to coordination planning and control of pose trajectories for robot end effector in Cartesian space. The organization of the chapter is as follows. Section 2 introduces some issues related to robot task analysis and control, including the evaluation of robot kinematics performance; Section 3 proposes a new approach to robot pose trajectory planning by generating robot pose ruled surface. Section 4 deals with the optimization solutions to the problem of pose trajectory planning. Section 5 provides simulation of the proposed approach in virtual environment and examples. Section 6 demonstrates the industrial application for robotic polishing. Section 7 summarizes the chapter with concluding remarks.

3. Analysis and Control for Robot Manipulators

There are different levels of abstraction at which a robot task can be defined. For simplicity, the lowest level task description is adopted so that a robot task is specially defined as a collection of working points to be followed by the end-effector in the task space. If the robot is to follow a prescribed path, this task can be approximated by a set of points along the path. Other task specifications such as obstacle avoidance, position accuracy, and static capability at task points can also be included as part of the definition of a task. In the following context, the task point is referred to as the pose (position and orientation) of the end-effector, which is dependent on the configuration of robot manipulator.

3.1 Robot Pose and Configuration

The robot configuration in Cartesian space can be described by the position reference point, $\boldsymbol{P}$, and the orientation vector Φ on the line S passing through the point $\boldsymbol{P}$. Thus, the configuration equation is expressed as follows (Makino et al 1982),

$$\mathrm{X}=[\boldsymbol{P},\ \Phi]^{T} \tag{1}$$

The continuous motion of configuration in three-dimensional Cartesian space forms a ruled surface, called robot pose ruled surface (Zha 1993). Two end points of the orientation vector $\mathbf{\Phi}$ are supposed to be ***P*** and ***Q*** on the ***S***, and thus the starting and ending points of robot motion are P_s and Q_s, P_e and Q_e, as shown in Figure 1, respectively. The two base curves of the robot pose ruled surface, i.e., the robot pose trajectories P_sP_e and Q_sQ_e can be expressed as the vector equations of a function of joint variable with respect to time parameter *t*, as follows,

$$\begin{cases} \mathbf{P}(\mathrm{t}) = \mathbf{r}_1(\mathrm{t}) \\ \mathbf{Q}(\mathrm{t}) = \mathbf{r}_2(\mathrm{t}) \\ \mathbf{\Phi}(\mathrm{t}) = \mathbf{r}_2(\mathrm{t}) - \mathbf{r}_1(\mathrm{t}) \end{cases} \tag{2}$$

where, $\mathrm{t} \in [\mathrm{t}_1,\mathrm{t}_2]$, t_1 and t_2 are the start and end time of motion respectively. Using a standard mathematical equation to express the robot pose ruled surface, Eq.(2) can be rewritten as (Zha and Du 2001)

$$\mathbf{r}(t,\lambda) = \mathbf{r}_1(t) + \lambda[\mathbf{r}_2(t) - \mathbf{r}_1(t)] = \mathbf{r}_1(t) + \lambda\mathbf{\Phi}(t) \tag{3}$$

where, $\lambda \in [0,1]$.

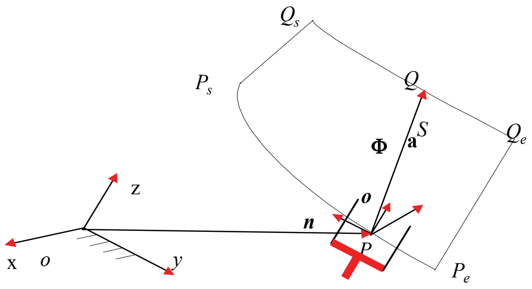


Fig. 1. Robot configuration and pose ruled surface.

From Eq.(3), when $\lambda = 0$, $\mathbf{r}(t,0) = \mathbf{r}_1(t)$ represents a pure position trajectory; while $\lambda = 1$, $\mathbf{r}(t,1) = \mathbf{r}_2(t)$ represents a pure orientation trajectory. The equation of the robot pose ruled surface is therefore flexible to represent its motion (both for position and orientation) trajectories. The orientation vector can be either an equivalent angular displacement vector or any other representational orientation vectors, e.g., an Euler angular vector (θ, ϕ, ψ). The properties of the equivalent angular displacement vector used as the orientation vector are different from those of normal vectors in mathematics (Makino et al 1982). It is noted that the representation for robot motions using the pose ruled surface is exceptional to handle the case when the orientation vector is in line with or parallel to the position vector as in these two cases no ruled surface can be formed or generated.

3.2 Motion Analysis

From the configuration equation, Eq.(1), the Cartesian velocity and acceleration, $\dot{\mathrm{x}}$ and $\ddot{\mathrm{x}}$, of the robot can be obtained by use of the first-order and the second-order derivatives, respectively, as follows

$$\dot{\mathbf{X}} = [\dot{\mathbf{P}}, \dot{\mathbf{\Phi}}]^T = J\,\dot{\mathbf{q}} = [\dot{\mathbf{r}}_1(t), \dot{\mathbf{\Phi}}(t)]^T \tag{4}$$

$$\ddot{\mathbf{X}} = [\ddot{\mathbf{P}}, \ddot{\mathbf{\Phi}}]^T = \mathbf{J}\ddot{\mathbf{q}} + \dot{\mathbf{J}}\dot{\mathbf{q}} = [\ddot{\mathbf{r}}_2(t), \ddot{\mathbf{\Phi}}(t)]^T \tag{5}$$

where, $\dot{\mathbf{P}}$ and $\ddot{\mathbf{P}}$ are the Cartesian linear velocity and acceleration of the robot position; $\dot{\mathbf{q}}$ and $\ddot{\mathbf{q}}$ are the joint velocity and acceleration; $\dot{\mathbf{\Phi}}$ and $\ddot{\mathbf{\Phi}}$ represent the orientation velocity and acceleration; and $\mathbf{J}$ is a Jacobian matrix. Based on the theory of velocity addition, $\dot{\mathbf{\Phi}}$ can be written as

$$\dot{\mathbf{\Phi}} = \sum_{j=1}^{n} \dot{q}_j \delta_j \bar{e}_j \tag{6}$$

where, $\bar{e}_j = (\prod_{i=0}^{j} E^{\,\hat{e}_i \delta_i q_i})\hat{\hat{e}}_j$ is a unit vector of j-th joint axis in the base coordinate system; $\hat{\hat{e}}_j$ is a unit vector of j-th joint axis in dynamic coordinate system; δ_j has the same meaning as δ_i mentioned in (Makino et al 1982). Hence, the Jacobian matrix $J(q)$ can be expressed as

$$\mathbf{J} = \begin{bmatrix} \mathbf{J}_v \\ \mathbf{J}_w \end{bmatrix} = \begin{bmatrix} \frac{\partial P}{\partial q_1} & \frac{\partial P}{\partial q_2} & \cdots & \frac{\partial P}{\partial q_n} \\ \delta_1 \bar{e}_1 & \delta_2 \bar{e}_2 & \cdots & \delta_n \bar{e}_n \end{bmatrix} \tag{7}$$

where, $\mathbf{J}_v$ and $\mathbf{J}_w$ are velocity Jacobian and angular Jacobian corresponding to $\mathbf{V}$ and $\dot{\mathbf{\Phi}}$ respectively.

3.3 Motion Performances Evaluation

The performance constraints are defined to ensure the feasibility of a robot configuration while performing the given task. Great advances have been made in recent years in the study of robotic kinematics performance. For example, in order to obtain an energy-saving motion mode, an efficient way is to minimize the joint motion, i.e., $\min(\sum_{i=1}^{n} | q_i(t+\Delta t) - q_i(t) |^2)$ or the sum weighted distance or path (Section 6). However, it is not sufficient to consider only the position and orientation of manipulators for task planning. In fact, in many cases, it is required to satisfy certain constraints for not only the position and orientation but also the velocity and even the acceleration, i.e., kinematics and dynamics and control constraints.

3.3.1 Manipulability

The most important and commonly used concept is the robot manipulability, which is a measure of manipulating ability of robotic mechanisms in positioning and orientating the end effector which is beneficial for design and control of robots and for task planning. The measure for manipulability was first defined by Yoshikama (1985), as

$$W = \sqrt{\det(\mathbf{J}\mathbf{J}^T)} \tag{8}$$

Since W is equal to zero at the singular positions, $W \neq 0$ and its increase will enlarge the dexterity of the system and assure avoiding "avoidable" singularities. For a non-redundant manipulators, i.e., $m=n$, the measure W reduces to $W=|\det(\mathbf{J})|$. The manipulability constraint is only used to detect whether a manipulator is at or nearby a singular pose when its end-effector reaches each of the task points, and thus it is a local manipulability index. For convenience, the *condition index* (CI) can also be employed to formulate the manipulability constraint (Angeles 1992):

$$\sigma_{\min} / \sigma_{\max} \geq \varepsilon \tag{9}$$

where, ε is the user defined lower bound of CI for singularity avoidance, which usually takes a small value, 0.001 for instance.

3.3.2 Area of Pose Ruled Surface and its Change Ratios

The area of the robot pose ruled surface with two base trajectory curves P_sP_e and Q_sQ_e can be obtained (Zha 1993, Zha 2002), as follows,

$$A=\int_{P_s}^{P_e}\int_{Q_s}^{Q_e} dA \tag{10}$$

where dA is the differential of area of ruled surface. From Eq (3), the area of robot pose ruled surface A can be further written as

$$A=\int_0^1 d\lambda \int_{t_1}^{t} \left|\dot{\mathbf{r}}_1(t)+\lambda\dot{\mathbf{\Phi}}(t)\right|\left|\mathbf{\Phi}(t)\right| dt \tag{11}$$

With respect to time t, the first-order and the second-order change ratios of the pose ruled surface, dA/dt and d^2A/dt^2, can be obtained respectively as

$$dA/dt=\int_0^1 |\dot{\mathbf{r}}_1(t)+\lambda\dot{\mathbf{\Phi}}(t)|\,\varphi(t)d\lambda \tag{12}$$

$$d^2A/dt^2=d\left(\int_0^1 |\dot{\mathbf{r}}_1(t)+\lambda\dot{\mathbf{\Phi}}(t)|\,\varphi(t)d\lambda\right)/dt \tag{13}$$

From Eqs (11-13), both the area of the robot pose ruled surface and its change ratio are functions of the pose trajectory equations, $\mathbf{r}_1(t)$ and $\mathbf{r}_2(t)$, and the velocity equations, $\dot{\mathbf{r}}_1(t)$ and $\dot{\mathbf{r}}_2(t)$. When a robot operates in a certain speed, the area of the robot pose ruled surface and its change ratios can indicate the kinematics and dynamics performances of the robot manipulator.

3.4 Dynamics Analysis and Control

The acceleration control can be considered as an extension of the velocity control, which controls the robot moving along the given motion parameters by imposing force and/or torque on each joint of robot. Thus, the main task of dynamics control is to determine the generalized force imposing on the robot joint to obtain the given acceleration. This can be achieved from the dynamics equation of the robot, as follows

$$\boldsymbol{\tau}=\boldsymbol{I}(\mathbf{q})\,\ddot{\mathbf{q}}+\mathbf{V}\,\dot{\mathbf{q}}+f(\mathbf{q},\dot{\mathbf{q}})+g(\mathbf{q}) \tag{14}$$

where, $\boldsymbol{\tau}$ is the jont driving force; $\boldsymbol{I}(\mathbf{q})$ is the $n\times n$ inertia matrix; $\mathbf{V}$ is the $n\times n$ damping matrix; $f(\mathbf{q},\dot{\mathbf{q}})$ is the n- dimensional nonlinear function of certrifugal and Coriolis terms; $g(\mathbf{q})$ is the gravitational terms. Furthermore, the corresponding dynamics equations in the Cartesian space can be derived as follows:

$$F = V(q)\ddot{\mathbf{X}} + U(\mathbf{q}, \dot{\mathbf{q}}) + P(\mathbf{q}) \quad (15)$$

where, F is the generalized operational force in Cartesian space; $V(\mathbf{q})$ is $m\times m$ kinetic matrix, i.e., inertia matrix in Cartesian space; $U(\mathbf{q}, \dot{\mathbf{q}})$ is the nonlinear function of certrifugal and Coriolis terms in Cartesian space; $P(\mathbf{q})$ is the gravitational terms in Cartesian space.
As discussed above, the problem of planning robot pose trajectories is equivalent to that of the generation of robot pose ruled surface. It means that the motion locus of configuration can be planned for the robot end effector in task space by generating the robot pose ruled surface. The change laws of robot pose can be determined by fitting or interpolating key or knot pose points obtained from artificial teaching-by-showing or measurement and even by the latest technologies (e.g. data glove) in virtual environment. Thus, the corresponding points of the entire motion path can be calculated by interpolation.

4 Coordination Planning for Robot Trajectories

4.1 Trajectory Generation

When a robot operates in task space, it must meet some requirements and constraints. Constraints for robotic motion trajectories are dependent on that the application requires zero-order (C^0), first (C^1)- or second (C^2)-order continuity, and kinematics or dynamics performances to yield a position and orientation continuous curve. Given a serial manipulator with prismatic and revolute joints operating in task space, consider the class of trajectories in Cartesian space, $X(t)=[\mathbf{r}_1(t), \mathbf{\Phi}(t)]^T \Rightarrow [\mathbf{r}_1(t), \mathbf{r}_2(t)]^T$, which satisfy the following kinematics, control and dynamics constraints and requirements :

(a) The robot end effector is desired to pass through the m specified or given knot pose points $\mathbf{r}_1(t_i)$, $\mathbf{r}_2(t_i)$, $i=1,2,3,...m$, accurately in workspace;

(b) When the robot moves along the planned trajectory in workspace, its motion is expected to be smooth with C^2 continuity and even small jerk, i.e., the existence of $\dot{\mathbf{r}}_1(t)$, $\ddot{\mathbf{r}}_1(t)$, $\dot{\mathbf{r}}_2(t)$ and $\ddot{\mathbf{r}}_2(t)$, or sometimes, $|\dddot{\mathbf{r}}_1(t)| \le J_{P_{\max}}$, $|\dddot{\mathbf{r}}_2(t)| \le J_{Q_{\max}}$, where, $J_{P_{\max}}$ and $J_{P_{\max}}$ are maximum pose jerks.

(c) The robot must avoid the singularity , W≠0, i.e., $\det(\mathbf{JJ}^T)\neq 0$ or CI ($\sigma_{\min}/\sigma_{\max} \ge \varepsilon$);

(d) The robot motion cannot exceed the joint range limits and maximum joint velo1city range, i.e., $|q_{i_{\min}}| \le q_i \le |q_{i_{\max}}|$, $|\dot{q}_i| \le \dot{q}_{i_{\max}}$.

(e)The robot motion cannot exceed the maximum joint acceleration and the maximum joint driving forces / torques, i.e., $|\ddot{q}_i| \le \ddot{q}_{i_{\max}}$, $|\tau_i| \le \tau_{i_{\max}}$, ($i=1,2,3,\ldots,n$)

(f) The robot must have better kinematics performances, e.g., maximum velocity-space or shortest motion path, or minimum area of pose ruled surface, or minimum change ratios of area of ruled surface.

There are many methods for fitting or interpolating key pose points to generate robot motion trajectories. These include segment interpolation (e.g. straight-line segment and transition segment), polynomial curves, cubic curves (e.g., Hermite cubic splines, Bezier curve, B-spline, NURBS), and so on (Boehm 1985, Patel and Lin 1988, Dierckx 1993, Ge and Ravani 1994, Ge and Kang 1995, Gerald Farin 1991). Suppose that P_sP_e and Q_sQ_e are formulated by one of the methods mentioned above, as shown in Figure 2, and they are described as follows:

$$\begin{cases} \mathbf{P} : \mathbf{r}_1(t) = x_1(t)\mathbf{i} + y_1(t)\mathbf{j} + z_1(t)\mathbf{k} \\ \mathbf{Q} : \mathbf{r}_2(t) = x_2(t)\mathbf{i} + y_2(t)\mathbf{j} + z_2(t)\mathbf{k} \\ \mathbf{\Phi} : \mathbf{\Phi}(t) = \mathbf{r}_2(t) - \mathbf{r}_1(t) \end{cases} \tag{16}$$

where, t∈ $[t_1,t_2]$. This means that both the position and orientation of robot vary from P_s to P_e and from Q_s to Q_e on the curves, respectively. If $\mathbf{r}_1(t)$ and $\mathbf{r}_2(t)$ or $\mathbf{r}_1(t)$ and $\mathbf{\Phi}$(t) are determined, the trajectory planning process is accomplished, and the coordinates of corresponding points on the pose trajectories can be calculated.

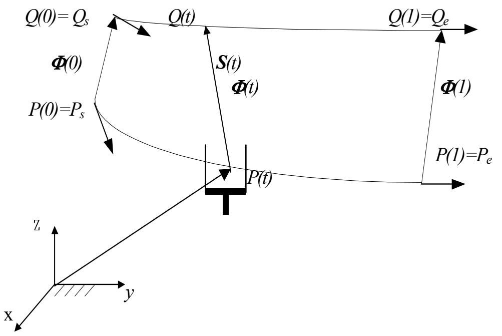


Fig. 2. Robot trajectory generation with constraints.

Here, we discuss the case when parametric cubic splines are used to interpolate pose data (Zha 2002). The pose trajectory curves, P_sP_e and Q_sQ_e, are assumed to be k-order space polynomial curves which can be explicitly expressed as

$$\begin{cases} \mathbf{P} : \mathbf{r}_1(t) = (\sum_{i=0}^{k} a_{1i}t^i, \sum_{i=0}^{k} b_{1i}t^i, \sum_{i=0}^{k} c_{1i}t^i)^T \\ \mathbf{Q} : \mathbf{r}_2(t) = (\sum_{i=0}^{k} a_{2i}t^i, \sum_{i=0}^{k} b_{2i}t^i, \sum_{i=0}^{k} c_{2i}t^i)^T \\ \mathbf{\Phi} : \mathbf{\Phi}(t) = [\sum_{i=0}^{k} (a_{2i} - a_{1i})t^i, \sum_{i=0}^{k} (b_{2i} - b_{1i})t^i, \sum_{i=0}^{k} (c_{2i} - c_{1i})t^i]^T \end{cases} \tag{17}$$

where, (a_{1i},b_{1i},c_{1i}) and (a_{2i},b_{2i},c_{2i}) (i=0,1,2,...,k) are polynomial coefficients of 3D coordinates of $\mathbf{r}_1(t)$ and $\mathbf{r}_2(t)$ respectively. The pose trajectory curves can also be represented as matrix equation as follows (Zha 2002)

$$\begin{bmatrix} \mathbf{r}_1(t) \\ \mathbf{r}_2(t) \end{bmatrix} = C_{P-Q}T = \begin{bmatrix} a_{10} & a_{11} & a_{12} & \cdots & a_{1k} \\ b_{10} & b_{11} & b_{12} & \cdots & b_{1k} \\ c_{10} & c_{11} & c_{12} & \cdots & c_{1k} \\ a_{20} & a_{21} & a_{22} & \cdots & a_{2k} \\ b_{20} & b_{21} & b_{22} & \cdots & b_{2k} \\ c_{20} & c_{21} & c_{22} & \cdots & c_{2k} \end{bmatrix} \begin{bmatrix} 1 \\ t \\ t^2 \\ t^3 \\ \vdots \\ t^k \end{bmatrix} \tag{18}$$

where, C_{P-Q}∈ $R^{6\times(k+1)}$ is the polynomial coefficients matrix to be determined, and T =$(1,t, t_2,...,t_k)^T$. Substituting Eq.(17) or Eq. (18) into Eqs (10-13), the area function and its change ratios can be determined. When the control points and constraints are determined, the polynomial coefficients, (a_{1i},b_{1i},c_{1i}) and (a_{2i},b_{2i},c_{2i}) (i=0,1,2,...,k), i.e., the coefficient matrix C_{P-Q}, can be obtained through the corresponding fitting or interpolating algorithms, and then the pose trajectories can therefore be obtained.

4.2 Trajectory Optimization

The optimal trajectory planning of robot manipulator is to formulate a task-oriented optimization model that consists of three parts: trajectory design parameters, objective function, and constraints. Trajectory design parameters and constraints are discussed for trajectory generation above. The objective function is related to performance evaluation, i.e., "goodness" of a robot trajectory. The straightforward way to define the objective function is to select one of performance measures or some of them in a weighted sum manner such as manipulability, reachability, joint and/or velocity range or space availability, motion jerks, and even the area of pose ruled surface and it change ratios.

Based on the above discussions, the area of robot pose ruled surface and its change ratios can indicate the kinematics performance. During the course of trajectory planning or task planning, the manipulability control must be considered. Therefore, the objective function can be defined as the area of pose ruled surface with good performance

$$\text{F}=\int_{P_s}^{P_e}\int_{Q_s}^{Q_e}\frac{1}{PI}dA \tag{19}$$

where,; P_s and P_e are the starting point and end point of position trajectory respectively; Q_s and Q_e are starting point and end point of orientation trajectory respectively; dA is a differential of area of ruled surface; PI is performance index, W, manipulability measure. Based on Eq.(11), Eq. (19) can be rewritten as

$$\text{F}=\int_0^1 d\lambda\int_{t_1}^{t}\frac{1}{PI}\left|\dot{\mathbf{P}}[q(t)]+\lambda\dot{\mathbf{\Phi}}[q(t)]\right|\left|\mathbf{\Phi}(t)\right|dt \tag{20}$$

After all the optimization variables or coefficients are determined, the optimal pose trajectories are obtained. The models discussed above are suitable for the generation of not only the position but also orientation trajectory of robot.

In some cases, a simplified model could be used. For example, for the position trajectory planning, λ =0 and the area differential dA of robot ruled surface is changed into arc length differential dS, and then the objective function becomes the shortest path, which can be specified and simplified as,

$$\text{F}=\int_{P_s}^{P_e}\frac{1}{PI}dS=\int_{t_1}^{t_2}\frac{1}{PI}\left|\dot{\mathbf{P}}(t)\right|dt \tag{21}$$

The objective function can also be the sum weighted distance as described in Section 6, The optimal trajectories can be found with a commercial optimization software package such as MATLAB optimization toolbox (1998) in which the most two typical optimization methods, the grid method and the stochastic constraint method, were employed. Details about the simulation and optimization will be discussed in Sections 5 and 6.

5. Coordination Control for Robot Trajectories

In order to control robot conveniently, curve length variables l_1 and l_2 are often selected as the path parameters of P_sP_e and Q_sQ_e,. In general case, coordinates of corresponding points on the robot pose trajectories can be determined by interpolation algorithms, as follows:

$$\begin{cases} \mathbf{P}: \mathbf{r}_1(l_1) = x_1(l_1)\mathbf{i} + y_1(l_1)\mathbf{j} + z_1(l_1)\mathbf{k} \\ \mathbf{Q}: \mathbf{r}_2(l_2) = x_2(l_2)\mathbf{i} + y_2(l_2)\mathbf{j} + z_2(l_2)\mathbf{k} \\ \mathbf{\Phi}: \mathbf{\Phi}(l_1, l_2) = \mathbf{r}_2(l_2) - \mathbf{r}_1(l_1) \end{cases} \tag{22}$$

where, $l_1=l_1(t)$, $l_2=l_2(t)$. This means that robot pose trajectories can be controlled by path variables, l_1 and l_2, which obey a specified motion laws. The conditions to be satisfied for the coordination control of pose trajectories can be derived as follows,

$$\Delta l_1 = \sqrt{\Delta x_1^{\ 2} + \Delta y_1^{\ 2} + \Delta z_1^{\ 2}}\ ,\ \Delta l_2 = \sqrt{\Delta x_2^{\ 2} + \Delta y_2^{\ 2} + \Delta z_2^{\ 2}} \tag{23}$$

$$l_1 = l_1(t) = \int_{t_1}^{t} \Delta l_1 dt\ ,\ l_2 = l_2(t) = \int_{t_1}^{t} \Delta l_2 dt\ ,\ \begin{cases} p_s = 0 \\ p_e = \int_{t_1}^{t_2} \Delta l_1 dt \end{cases},\ \begin{cases} q_s = 0 \\ q_e = \int_{t_1}^{t_2} \Delta l_2 dt \end{cases} \tag{24}$$

As discussed above, the trajectory coordination and the calculation of corresponding pose point coordinates are dependent on and controlled by the motion laws of path parameters, such as uniform motion, constant acceleration, uniform and constant deceleration motion, etc. Two typical motion laws of path parameters were discussed in (Zha and Chen 2004). According to Eq.(22), the corresponding pose coordinates can be calculated for each pair of possible curves, such as line-line, line-arc, arc-line, arc-arc, high-order polynomials, etc. Consequently, the pose and its velocities and accelerations at any time can be determined or controlled. The problem determining the control laws of pose in Cartesian coordinate space is thus solved.

6. Planning and Control Simulation

To verify the proposed models above, simulation for coordinated planning and control should be carried out in an integrated environment. In this section, the simulation of the proposed approach is discussed.

6.1 Simulation Process

Using an optimization method, the optimal trajectory planning can be fulfilled (Zha and Du 2001). After the pose trajectories or the pose ruled surface are determined, the system calculates the corresponding position and orientation point coordinates based on the specified motion laws of path parameters. Finally, the system carries out the motion animation and outputs the joint angles for robot controller, and the simulation process is thus finished. The flowchart of the planning and simulation process in an integrated environment can be described in Figure 3.

The simulation environment is developed using Robot Toolbox (Corke 1999), Spline Toolbox and Optimization Toolbox, which are all based on the MATLAB package (1999). The task evaluation consists of two parts: performance constraint detection and fitness value computation. If a pose trajectory or a pose ruled surface can satisfy all the kinematics, dynamics and control constraints such as reachability, joint range availability, and manipulability, joint torques, discussed in Section 3, its fitness can be calculated by Eq.(23); otherwise the fitness is assigned to an infinite large number indicating such a trajectory or a pose ruled surface is infeasible.

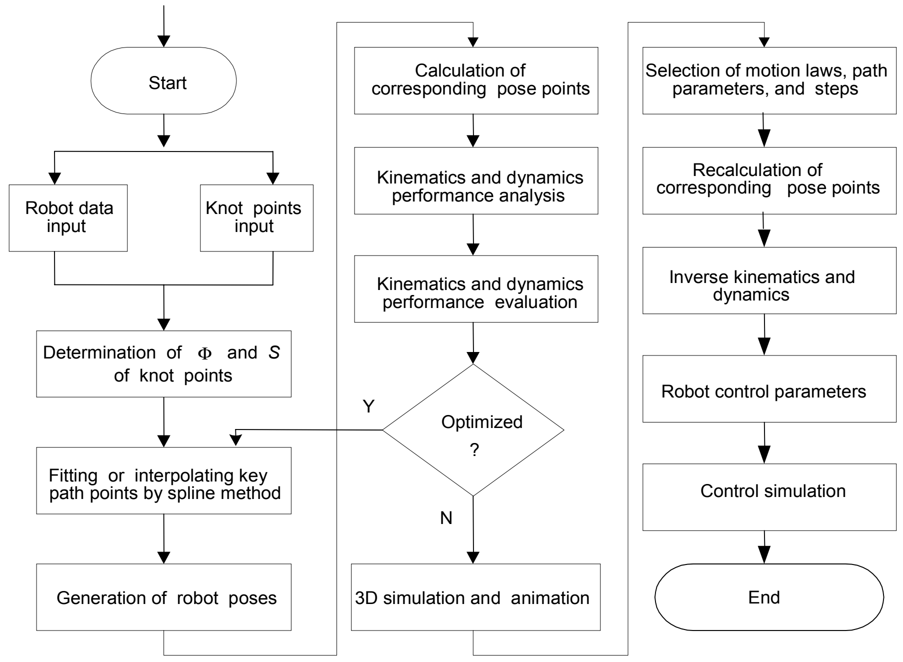

Start
Robot data input
Knot points input
Determination of Φ and S of knot points
Fitting or interpolating key path points by spline method
Generation of robot poses
Calculation of corresponding pose points
Kinematics and dynamics performance analysis
Kinematics and dynamics performance evaluation
Optimized ?
Y
N
3D simulation and animation
Selection of motion laws, path parameters, and steps
Recalculation of corresponding pose points
Inverse kinematics and dynamics
Robot control parameters
Control simulation
End

Fig. 3. Flowchart of robot trajectory planning and control simulation.

5.2 Two Simulation Examples

The first example is a 3-DOF planar robot manipulator (Zha 2002), as shown in Figure 4. The link length of this mechanism is represented by l_1, l_2, and l_3, where l_1=l_2=0.400, and l_3=0.200, respectively. The robot end effector is supposed to move along a trajectory in task space from P_s (-0.400,0.200,0)^T to P_e(0.400,0.200,0)^T with a constant orientation and operation force defined as $\Phi=(0,0,-\frac{\pi}{2})^T$ and F= $(1,1,1)^T$, respectively. The motion time is required to be within T=60 seconds, which is the same as the time taken by the robot end effector to move from the start point to the end point. From the configuration of robot and geometric relationships, the position trajectory in workspace can be given by

$$r_1(t)=(l_1c_1+l_2c_{12}+l_3c_{123}, l_1s_1+l_2s_{12}+l_3s_{123}, 0)^T \tag{25}$$

where, $c_1=\cos\theta_1$, $s_1=\sin\theta_1$, $c_{12}=\cos(\theta_1+\theta_2)$, $s_{12}=\sin(\theta_1+\theta_2)$, $c_{123}=\cos(\theta_1+\theta_2+\theta_3)$, $s_{123}=\sin(\theta_1+\theta_2+\theta_3)$, and $\theta_1+\theta_2+\theta_3=\frac{3\pi}{2}$. By derivative of Eq (25), the following is obtained

$$\dot{r}_1(t)=[-l_1\dot{\theta}_1 s_1-l_2(\dot{\theta}_1+\dot{\theta}_2)s_{12}, -l_1\dot{\theta}_1 c_1+l_2(\dot{\theta}_1+\dot{\theta}_2)c_{12}, 0]^T=J\dot{q} \tag{26}$$

where, $\dot{q}=(\dot{\theta}_1,\dot{\theta}_2,\dot{\theta}_3)^T$, and J is a Jacobian matrix as

$$J=\begin{bmatrix} -l_1s_1-l_2s_{12} & -l_2s_{12} & 0 \\ l_1c_1+l_2c_{12} & l_2c_{12} & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{27}$$

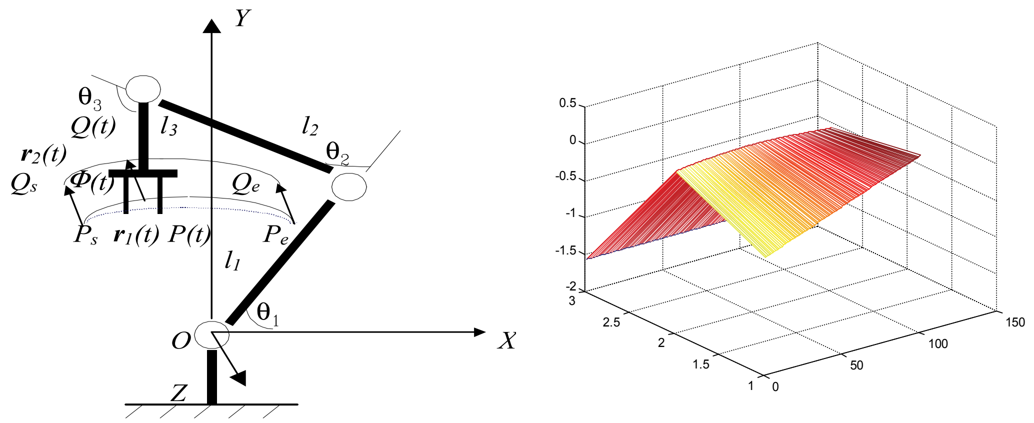


Fig. 4. Trajectory planning for a 3-dof manipulator (with redundancy).

According to Eq (27), the manipulability measure of robot can be written as

$$W=\sqrt{\det[JJ^T]}=|\det J|=l_1 l_2 |s_2| \tag{28}$$

The pose trajectories P_sP_e and Q_sQ_e for the manipulator are supposed to be 6-order space polynomial curves respectively, i.e., as follows

$$r_1(t)=(\sum_{i-0}^{6} a_{1i}t^i, \sum_{i=0}^{6} b_{1i}t^i, 0)^{\mathrm{T}} \tag{29}$$

$$r_2(t)=(\sum_{i-0}^{6} a_{1i}t^i, \sum_{i=0}^{6} b_{1i}t^i, -\pi/2)^{\mathrm{T}} \tag{30}$$

where, $t\in[0,1]$ are corresponding to P_s and Q_s ;P_e and Q_e respectively. Thus, the optimization objective function or fitness function can be chosen as

$$\mathrm{F}=\frac{\pi}{2l_1l_2}\int_0^1 \frac{1}{|s_2|}\sqrt{(\sum_{i=1}^{6} ia_{1i}t^{i-1})^2+(\sum_{i=1}^{6} ib_{1i}t^{i-1})^2}\,dt \tag{31}$$

Based on the fact that t=0 and t=1 is corresponding to P_s ,Q_s and P_e and Q_e respectively, the optimization constraints are expressed as

$$a_{10}=-0.400, \quad b_{10}=0.200, \quad \sum_{i=0}^{6} a_{1i}=0.400, \sum_{i=0}^{6} b_{1i}=0.200$$

Using the traditional optimization method, stochastic constraints in MATLAB optimization toolbox or the genetic algorithm, the coefficients (a_{1i},b_{1i}) (i=0,1,2,...,6) for the optimal trajectory P_sP_e with the shortest path and maximum flexibility can be obtained (Table 1).

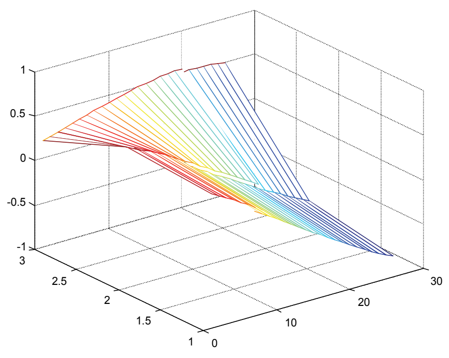
i	0	1	2	3	4	5	6
*a_{1i}	-0.4000	0.7903	-0.0022	0.1673	-0.2364	0.07275	-0.0083
*b_{1i}	0.2000	0.2770	-0.2544	0.0483	-0.2511	0.1826	-0.0023

Table 1. Optimal coefficients of trajectory equations for 3-DOF robot manipulators.

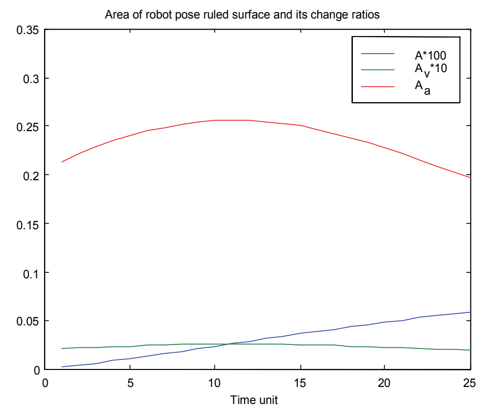
The second example is for PUMA 560 robot used for arc welding (Zha 2002). Assume that the end effector of robot PUMA 560 is required to move along a trajectory passing through the configuration points in task space. These key pose points are supposed to be fitted by 6-order space polynomial curves. Using the proposed model and approach, the optimal polynomial coefficients for the trajectory equations satisfying the kinematics and dynamics constraints, with the minimum path or pose ruled surface area and maximum flexibility, can be obtained. Table 2 lists the optimized coefficients for the trajectories. Figures 5-7 demonstrate the simulation and animation results for PUMA 560 robot moving along the optimized trajectories.

i	0	1	2	3	4	5	6
$*a_{1i}$	0.8000	0.0000	-3.7909	48.2443	-167.4964	198.5294	-79.3606
$*b_{1i}$	0.4000	0.0000	16.8877	-54.3584	39.1373	17.4540	-19.8360
$*c_{1i}$	0.2000	0.0000	-15.4005	66.5505	-103.0331	63.3715	-14.2356
$*a_{2i}$	0.6000	0.0000	17.5912	-58.9944	28.1370	43.7332	-32.8612
$*b_{2i}$	0.4000	0.0000	15.0318	-28.7452	-49.2458	121.6272	-60.6053
$*c_{2i}$	0.4000	0.0000	-25.8490	101.9592	-126.8347	48.5275	0.8091

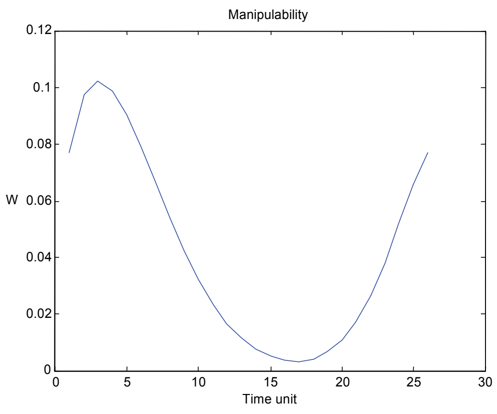
Table 2. Optimal coefficients of trajectory equations for PUMA 560 robot.

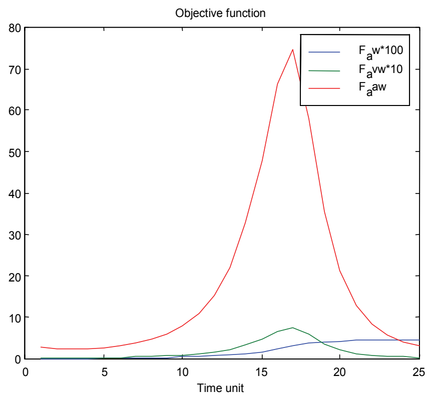


(a) Robot pose ruled surface



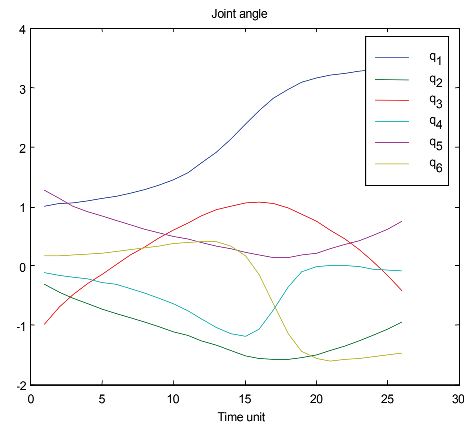
(b) Area of pose ruled surface and its change ratios



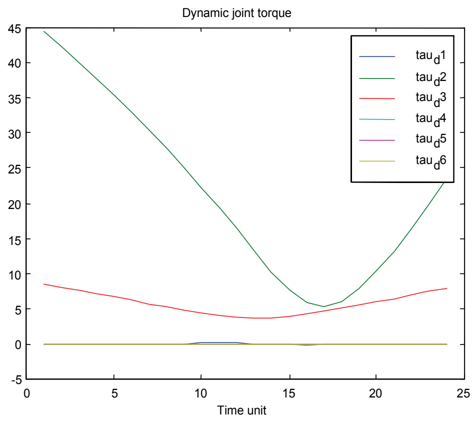


(c) Manipulability (d) Objective function values (PI=W)

Fig. 5. The optimized performance indexes and objective functions for PUMA 560 robot.

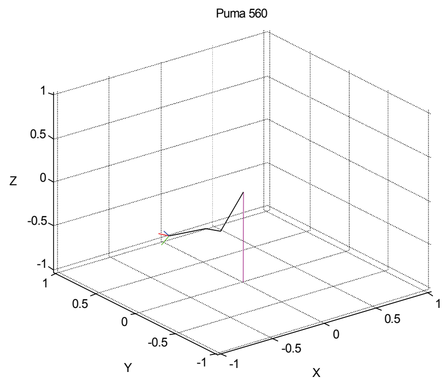


(a) Joint angle



(b) Joint torque

Fig. 6. Kinematics and dynamics analysis for PUMA 560 robot moving along the optimized trajectories.



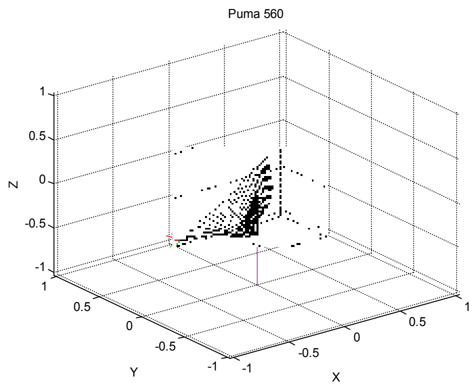


Fig. 7. Motion animations for PUMA 560 robot moving along the optimized trajectories.

7. Industrial Application for Robotic Polishing

7.1. Profile Reconstruction for Distorted Surface

One challenging task for robotic applications lies in precise materials surface processing. A robot is required to mimic an operator to manipulate a processing tool to remove materials from free-form surface. Fig. shows the schematics of a high pressure turbine (HPT) vane. The vane consists of an airfoil having concave and convex surfaces, an inner buttress and an outer buttress. After operating in a high-temperature and high-pressure environment, vanes incur severe distortions as large as 2 mm in reference to the buttress. On the airfoil surface there are hundreds of cooling holes. After a number of operational cycles, defects such as fully or partially blocked cooling holes, micro cracks and corrosions begin to occur. Because of the high cost of the components, it is common practice to repair these parts instead of scrapping them. The repairing process starts with cleaning and covering the defective areas with the braze material. The purpose of brazing is to fill up the defects, but unavoidably, the brazed areas will be higher than the original surface.

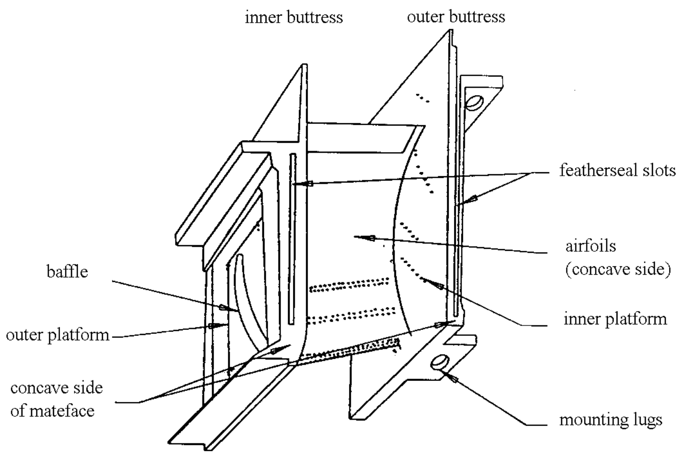


Fig. 8. Schematics of a HPT Vane.

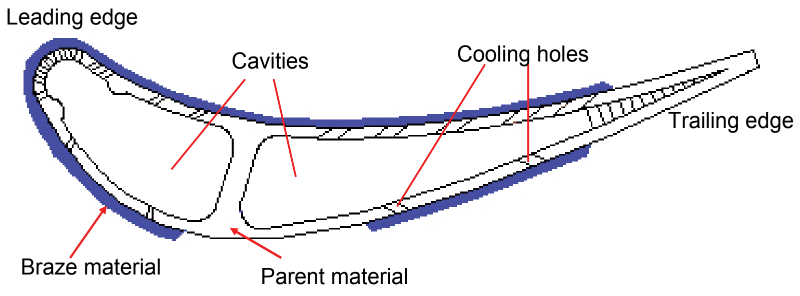


Fig. 9. Turbine airfoils repaired with braze material.

Fig. shows a cross section of the airfoil brazed with a repair material. The grinding and polishing process is intended to remove the excessive materials and make the brazed area flush with the original surface within a tight tolerance about 100 μm. Current manual polishing uses belt machine to remove the braze without undercuts and overcuts. For a robot to carry out such a delicate task, dexterous motion and control is needed to achieve compliant contact and contact force between airfoil and contact wheel.
The turbine airfoils to be repaired have severe distortions and twists after operations in the high-temperature and high-pressure environment. A teach-and-play robot cannot cope with the distorted profile. Neither can a commercial off-line programming system, which generates robot path according to the design data. It is absolutely critical and necessary to have profile sampling and distortion compensation system in this specific application to deal with part-to-part variations. Before any distortion compensation, the actual profile has to be sampled. A Linear Variable Differential Transducer (LVDT) has been integrated into the robotic system to carry out profile measurement. After gripping the part, the robot approaches the measurement probe, as shown in Fig. 10.
The sensor has good resolution and accuracy and reliability. As compared with off-line measurement, in-situ measurement enables the robot to act as a measurement instrument. Consequently, common fixturing and datum can be used for both measurement and polishing. This is advantageous in minimising fixture errors.

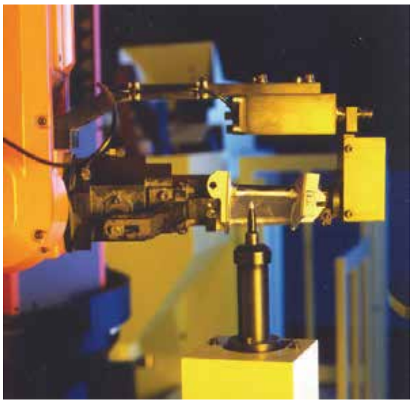


Fig. 10. Robot holding the workpiece probed the LVDT.

Selected points on the vane airfoil, many of which are covered by the brazing material, are sampled by the sensor. Approximation is made to offset the measured points to the prior-to-braze airfoil surface. The sensory readings only give the displacements in Z axis. In order to obtain the true coordinates of the measured points, corresponding robot coordinates have to be used for computation in conjunction with displacement readings.
The Optimal Profile Fitting (OPF) algorithm (Chen, X.Q. et al, 2002) fits a template to the actual measurement points with minimum sum of errors. The sectional template profiles are established based on design data using Cubic Spline Interpolation. It ensures that not only the interpolation is continuously differentiable (C^0, C^1) on the interval, but also has a continuous second derivative (C^2) on the interval.

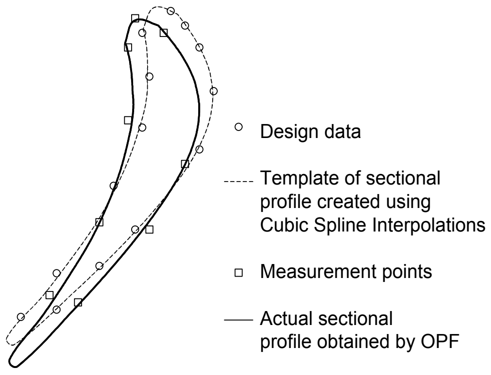

○ Design data
----- Template of sectional profile created using Cubic Spline Interpolations
□ Measurement points
—— Actual sectional profile obtained by OPF

Fig. 11. Actual 2D sectional profile obtained by OPF.

The measurement data are first transformed from global coordinate system to local (tool) coordinate system. The OPF is carried out in the tool coordinate system on each 2D cross sectional profile. Three 2D sectional profiles are taken as templates to do the fitting. The template for each section is the design profile at the respective section, which is obtained through cubic spline interpolations. We assume that a template is a rigid 2D profile. Therefore there are three degrees of freedom in the optimal fitting, namely X axis shift, Y axis shift, and rotation around a certain point. Fig illustrates the concept of OPF.
Each cross section is computed individually. The complete 3D airfoil profile is obtained through interpolating the cross sections**.** The goal of the optimal fitting is to find the optimal X-Y shift and

rotate values for the template, so that after the transformation, the sum distance from the template to the measures points at the given height is minimum. In other words, the optimal fitting problem is a multi-dimensional minimization problem and the sum distance is used as the performance index function in the minimization process. We have developed and implemented the Search-Extend-Focus (SEF) minimization algorithm (Chen, X.Q. et al, 2002). SEF finds the optimal point at which the index function is minimum.
By varying the definition of the performance index function (i.e. the sum distance), different optimal fitting results can be obtained. Here, the following definition of the sum distance is used:

$$d_{sum} = w_h h^2 + \sum w_i d_i^2 \tag{32}$$

where d_{sum} is the sum distance defined, d_i is the distance from ith measure point to the profile, h is distance from the given height to the profile, w_i and w_h are the respective weights in the sum distance. When different weights w_i and w_h for different measure points are chosen, different priorities are given to different measure points. By choosing high weights for groups of measure points in turn, multiple optimal fitting can be carried out for one sectional optimal fitting. The weights in the optimal profile fitting are given in Table 3.

	Weight for Convex Side Measure Points					**Weight for Concave Side Measure Points**					**Weight for height**
	1	2	3	4	5	6	7	8	9	10	
Convex Fitting	10.0	2.0	2.0	2.0	2.0	1.0	1.0	1.0	1.0	1.0	10.0
Concave Fitting	1.0	1.0	1.0	1.0	1.0	2.0	2.0	2.0	2.0	10.0	10.0

Table 3. Weights for Measure Points and Height in Optimal Fitting.

Note: Measure Point 1 and 10 are on the trailing edge of the profile without brazing material on the surface, and the other measure points are all on the profile with brazing material on the surface.
Combining the multiple optimal fitting results, the following fitting objectives are achieved:

1. All portions of the profile are optimally fitted to the measure points.
2. Whole profile is smooth.
3. No distortion of the profile shape at each portion. Minimum distortion exits only at adjacent areas of two portions of the profile for smooth transition.

7.2. Robot Polishing Path Planning for 3D Robotic Surface Finishing

Having an accurate description of the airfoil profile is not the ultimate aim. The computed profile based on the sensory data must be used to automatically generate the robotic polishing path. This process has to be repeated for every part.
The robot polishing path is a point-to-point motion in the Cartesian coordinate system. With the Euler angle computation and the coordinate transformation, the robot end-effector location (position and orientation) in Cartesian coordinates are generated from the contact points in the tool coordinate system. Along a curve between any two points, the robot automatically moves using the cubic spline line motion. Thus, we only concern with the formalism of deriving the robot coordinates (*X, Y, Z, A, B, C*) of the path points which the robot must travel along in the Cartesian coordinate system. The coordinates of every path point are calculated based on the robot kinematics model. In addition, certain system setup

parameters, such as polishing wheel size and global locations of all tool heads, have been built into the model. With this feature, intuitive re-calibration of tooling can be done rapidly and accurately. Fig. 12 shows one path point computed by ARP.

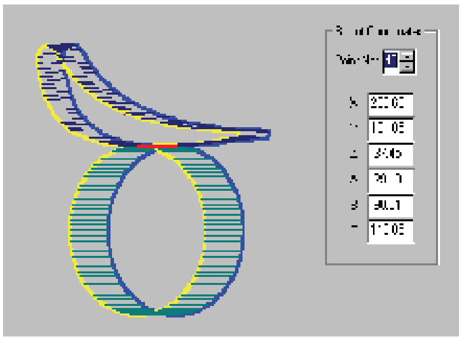


Fig. 12. Computer simulation of path generated by ARP.

The space curve that the robot end-effector moves along from the initial location (position and orientation) to the final location is called the *path* (Fu, K. S.; Gonzalez, R. C. & Lee, C. S. G, 1987). It deals with the formalism of describing the desired the robot end-effector motion as sequences of points in space (position and orientation of the robot end-effector) through which the robot end-effector must pass, as well as the space curve that it traverses.

Path planning scheme generates the desired path by a series of points (the endpoints and intermediate points) in Cartesian coordinates. They are specified in Cartesian coordinates not in joint coordinates because it is easier to visualize the correct configurations in Cartesian coordinates than in joint coordinates. In this work, the robot polishing path is a point-to-point motion in Cartesian coordinate system. With the Euler angle computation and the coordinate transformation, the robot end-effector location (position and orientation) in Cartesian coordinates can be generated from the local coordinates of the polishing path points on the surface of work piece in robot end-effector's coordinate system for polishing process. Each path knot point of the robot end-effector is described by six robot coordinates (*X, Y, Z, A, B, C*), where coordinates (*X, Y, Z*) specify the robot end-effector position and coordinates (*A,B,C*) specify the robot end-effector orientation.

For the space curve between any two points, the robot automatically moves using the cubic spline line motion. Thus, we only concern with the formalism of deriving the robot coordinates(*X, Y, Z, A, B, C*) of the points which the robot must .traverses in Cartesian coordinate system.

There are many different types of Euler angle representations. The scheme we adopted is illustrated in Fig. 1. The resultant Euler rotation matrix is given by:

$$R_{A,B,C} = R_{Z,A} R_{Y',B} R_{Z'',C}$$
$$= \begin{bmatrix} \cos A \cos B \cos C - \sin A \sin C & -\cos A \cos B \sin C - \sin A \cos C & \cos A \sin B \\ \sin A \cos B \cos C + \cos A \sin C & -\sin A \cos B \sin C + \cos A \cos C & \sin A \sin B \\ -\sin B \cos C & \sin B \sin C & \cos B \end{bmatrix} \tag{33}$$

With the above defined Euler angle rotation matrix, the orientation of the robot end-effector can be derived with respect to the reference global coordinate system. Then plus the translation of the robot end-effector, the exact location of the robot end-effector (path point) after moving robot coordinates (*X,Y,Z,A,B,C*) in global coordinate system is developed.

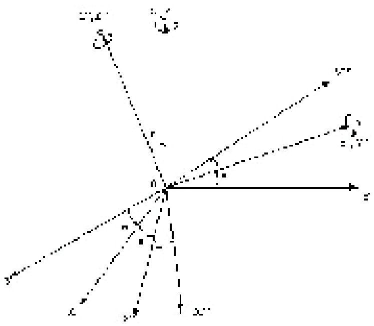


Fig. 13. Euler angles system.

A robot path is a sequence of points to polish the surface of the work. Each point can be described in terms of robot coordinates (*X, Y, Z, A, B, C*) which can be recognized by the robot controller. An illustration about robot path generation is given in Fig. 1. There are four 3D coordinates systems:

(1) Coordinate system A: This is the global coordinate system.
(2) Coordinate system B: Robot end-effector coordinate system.
(3) Coordinate system C: Tool coordinates system.
(4) Coordinate system D: Part coordinate system constructed for each polishing position as shown in Fig. 15.

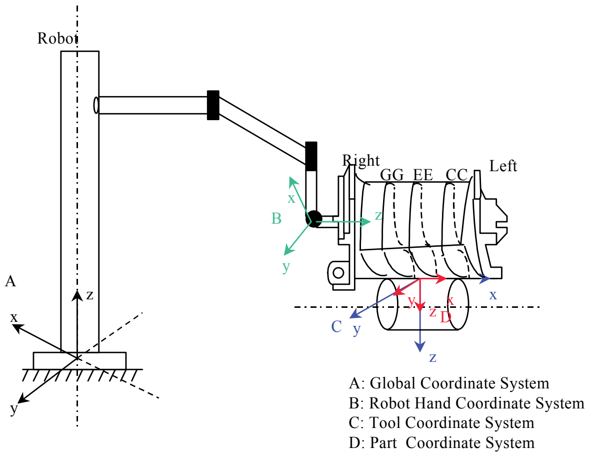

Robot
Right
GG EE CC
Left
A
B
C
D
x
y
z
A: Global Coordinate System
B: Robot Hand Coordinate System
C: Tool Coordinate System
D: Part Coordinate System

Fig. 14. Robot Polishing Path Generation.

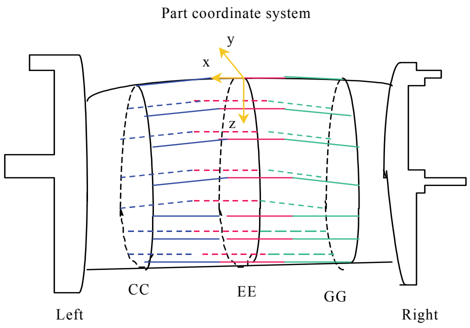


Fig. 15. Part coordinate system.

After the part's 3D profile is obtained by using the Optimal Template Fitting Method, the part coordinates system is constructed on the 3D profile for each polishing point. The part and tool coordinate systems are constructed such that, for each polishing point, the part is at desired polishing position when the two coordinate systems are overlapped. Based on this, we can derive the position of robot end-effector's coordinate system B by Coordinate System Transformation, and further more, the robot coordinates (*X, Y, Z, A, B, C*).

The robot path generation is the inversion process of coordinates transformation by translation and Euler angles rotation. The following is the procedure to compute the robot coordinates of a polishing point:

Step 1. Compute the polishing points in robot end-effector coordinate system.

Step 2. Construct a part coordinate system for a polishing point.

Step 3. Assume that the part coordinate system is overlapped with tool coordinate system, compute the robot end-effector coordinate system by using the coordinates system transformation.

Step 4. From the position of robot coordinate system in the global coordinate system, derived the robot coordinates (*X, Y, Z, A, B, C*).

Step 5. Repeat Step 2 to Step 4 to obtain a series the robot coordinates (*X, Y, Z, A, B, C*).

Through above steps, a series of robot coordinates (*X, Y, Z, A, B, C*) are developed for each desired polishing positions, which forms the robot polishing path.

7.3. Integrated Mechatronic Control of Robotic Surface Finishing

In robotic applications based on position control, path information is sufficient. In the sophisticated constrained application like surface finishing, contact force, compliance and tool wearing become indispensable. Process model including polishing tool geometry, contact stiffness, pre-load, Z displacement, robot travel speed and direction, etc. have been obtained from extensive laboratory prototyping on process optimization. These optimum process parameters are encapsulated in the process knowledge base. Operational parameters such as Z-axis offset, robot travel speed are associated with each path point to generate desired airfoil profile. Tool wear compensation is automatically done by associating process properties to the polishing path. Process development and optimization (Huang, H. et al, 2002) is critical to achieving an integrated mechatronic solution to the 3D polishing application.

Fig. 16 shows the architecture of the Knowledge-Based Adaptive Robotic System for 3D profile polishing. It comprises three inter-related hierarchical layers, namely, Device and Process, Knowledge-Based Process Control (KBPC), and Data-Driven Supervisory Control (DDSC). The Supervisory Controller is driven by product and process data including scheduling, system configuration, product design data, cross sectional data (template), and tool coordinates.

The Device/Process Control sub-system controls actuators and sensors to fulfil the required processes, and coordinates the process flow. In addition, it also acquires measurement data and exchanges data and information with the Supervisory Controller. Process Control relies on the following process knowledge bases:

Historical Process Database. It holds records of individual parts, such as measurement data, processing time, measurement data of finished profile (optional), breakdowns, uptime, and downtime.

Tool Compensation Knowledge Base. It stores tool compensation parameters such as the abrasive belt speed and the workpiece feed rate.

Path Optimisation Knowledge Base. It contains optimum process parameters such as Z-axis offset, approaching angle, robot travel speed.

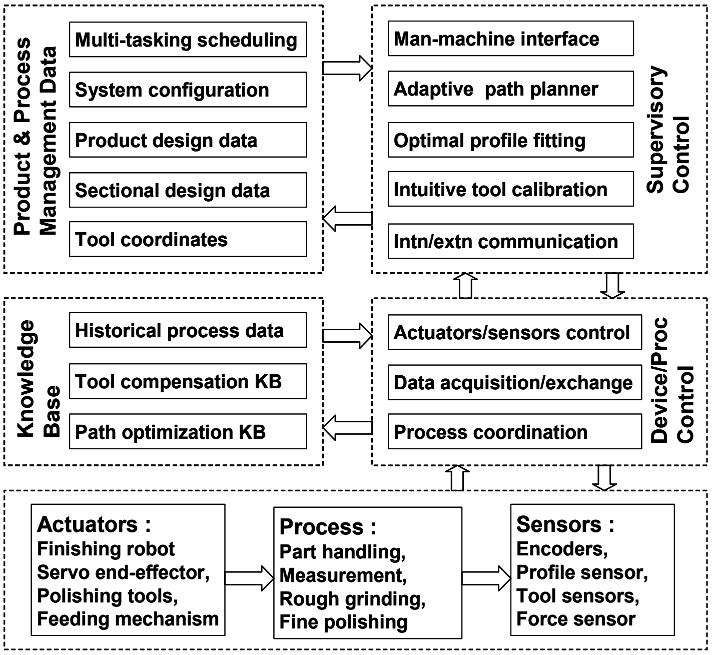


Fig. 16. Control System Architecture for robotic surface finishing.

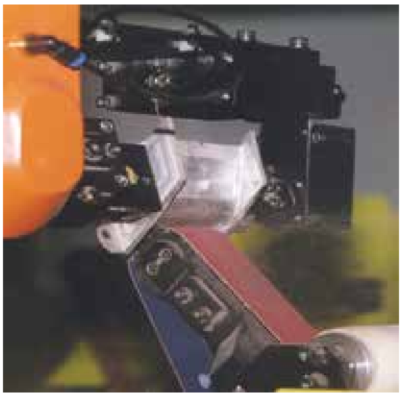


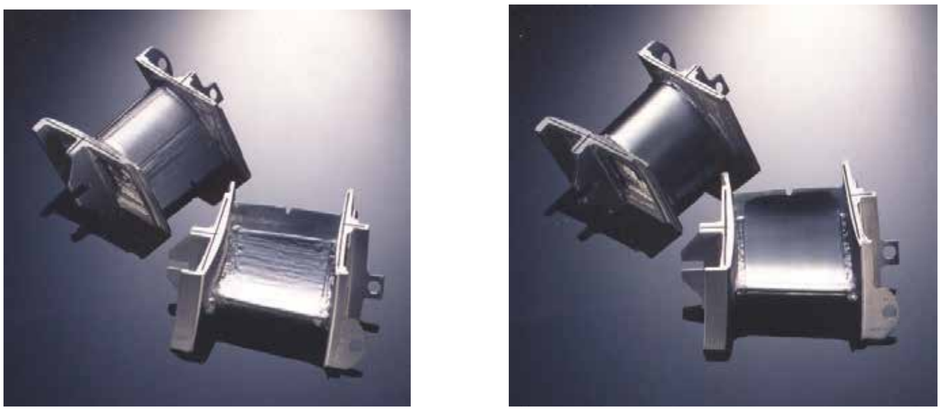
Fig. 17. Robot holding workpiece in contact with the polishing belt.

Figure 17 shows Robot holding workpiece in compliant contact with the polishing belt. It carries out polishing with the planned tool path and optimum process parameters. Optimum parameters can be inferred based on the part conditions: curvature, braze thickness, leading edge height, and trailing edge distortions. As a result, a smooth finish profile, free of transition lines, overlapping marks and burning marks, can be obtained.
The Supervisory Controller contains the following control functions:

- **Internal / External Communication**. Internal communication involves information and data exchange between control modules in DDSC. External communication allows data exchange with KBPC.

- **Intuitive Tool Calibration (ITC)**. During machine re-calibration, set-up or re-installation, position data of tool stations, measurement stations, and index table can be manually clocked. These data are keyed into the database. The mathematical model of workspace and robot kinematics is automatically generated.
- **Optimal Profile Fitting (OPF)**. It generates the actual profile based on the in-situ measurement data. The robust fitting algorithm uses the sectional data as templates, and maps them with the measurement points. 3D free form surface is generated through interpolation of cross sectional profiles.
- **Adaptive Path/Strategy Planner (ARP)**. It automatically generates the optimum tool path based on individual part conditions, and furthermore synthesises the robot programs from the resultant path points.
- **Human-Machine Interface (HMI)**. It allows the user to change system parameters, configure system, select product configurations, enter data, and make queries.

Through adaptive robot polishing path planning and knowledge based process control, very smooth airfoil finishing profiles were achieved by. Vanes before and after polishing are shown in Fig. 18. Further visual inspection shows no visible transition lines from the non-brazed area to the brazed one, no visible polishing marks in the cutting path overlap areas, and no burning marks. The curvature transition from the concave to convex airfoil is very smooth and more consistent than one generated by manual polishing.


(a) (b)
Fig. 18. Vanes before (a) and after (b) robotic grinding and polishing.

8. Summary and Conclusions

This chapter presented a unified approach to robot trajectories coordination planning and control in Cartesian space. The unified treatment of the robot end-effector's position and orientation is based on the pose ruled surface concept and used in trajectory interpolations. The generation and control of pose trajectories for robot end effector could be carried out by generating the 3D robot pose ruled surface or path and determining the minimum or maximum value of its area function with kinematics and dynamics constraints.

The models and algorithms as demonstrated in this chapter are generally reliable and effective if control parameters are appropriately selected. Comparing with existing methods, the proposed planning and control models and algorithms are more generic, and can be used as an alternative to reduce the computing cost and storage in robot planning and control. During the planning and control of pose trajectories, the position, velocity and acceleration, and even the jerk and the torque limits of the robot joints are taken into consideration as constraints with smooth functions of time. It can be ensured that the robot moves along the planned trajectories with good kinematics and dynamics performances. Thus, the obtained trajectories are efficient both to compute and to execute, and the planned motions are uniform and can rule out bang-bang trajectories. The simulation and experimental study in arc welding and surface finishing process show that the planned trajectories are optimal with respect to the performance indexes, and can be realized through animation without violating any constraints of the robot actuators. Potentially, the proposed model and approach are also useful for computer graphics, CAD-based off-line programming, CAD/CAM (e.g., CNC, wire-EDM) in virtual/real design environments.

9. Disclaimer

Any opinions, findings, conclusions, or recommendations expressed in this material are those of the authors. No approval or endorsement for any commercial product, service or company by the National Institute of Standards and Technology is intended and implied.

10 References

Angeles, J. (1992). The design of isotropic manipulator architectures in the presence of redundancies, *International. Journal of Robotics Research*, 11(3), pp.196-201

Barr, A., Curin, B. Gabriel, S. and Hughes, J. (1992). Smooth interpolation for orientations with angular velocity constraints using quaternions, *Computer Graphics*, Vol.26, pp.313-320

Bokeberg,E.H., Hunt, K.H. and Ridely, P.R. (1992). Spatial motion-I: acceleration and the differential geometry of screws, *Mech. Mach. Theory*, Vol.27, No.1, pp: 1-15

Bobrow, J.E., Dubowski, S. and Gibson, J.S. (1985). Time-optimal control of robotic manipulators along specified paths, *International Journal of Robotics Research.*, Vol.4, No.3, pp.3-17

Boehm,W. (1985). Curvature continuous curves and surfaces, *Computer-Aided Geometric Design*, Vol.2, pp.313-323

Craig, J. J. (1989). *Introduction to Robotics: Mechanics and Control*, Addison Wesley Longman, ISBN 0-201-09528-9, Singapore.

Chen X. Q., Gong Z. M., Huang H., Ge S. S. , and Zhu Q. (1999). Development of SMART 3D polishing system, *Industrial Automation Journal*, Vol. 8, no. 2, pp. 6-11

Chen X. Q., Chang H., Chan K., Fong A. M., Kwek T. Y., Kerisnan B., Tan C. S., Lim P. H. (1998). Design and implementation of multi-sensory autonomous welding system for high quality precision welding of large boxed structure, *Proceedings of the Fifth International Conference on Control, Automation, Robotics and Vision*, pp.15-19, Singapore

Chen, X. Q.; Gong, Z. M.; Huang, H.; Ge, S. Z. & Zhou, L. B. (2002). Adaptive robotic system for 3D profile grinding and polishing, In: *Advanced Automation Techniques in Adaptive Material Processing*, Chen, X. Q.; Devanathan, R. & Fong, A. M. (Ed.), pp. 55-90, World Scientific, ISBN 981-02-4902-0, Singapore.

Corker, P. I. , Robot Toolbox for MATLAB Reference Guide, 1993-1999

Fu, K. S.; Gonzalez, R. C. & Lee, C. S. G (1987). *Robotics: Control, Sensing, Vision, and Intelligence*, McGraw-Hill Inc., ISBN 0070226253, New York.

Ge, Q.J. and Ravani, B. (1994). Geometric construction of Bezier motions, *ASME Journal of Mechanical Design*, Vol.116, No.3, pp.749-755

Ge, Q.J. and Kang, D.L. (1995). Motion interpolation with G^2 composite Bezier motions, *Transactions of ASME, Journal of Mechanical Design*, Vol.117 ,No.3, pp.520-525

Gong Z. M., Chen X.Q., and Huang H. (2000). Optimal profile generation in surface finishing, *Proceedings of the 2000 IEEE International Conference on Robotics and Automation*, pp. 1557-1562, San Francisco, pp14-28

Gerald Farin (editor) (1991), NURBS for Curve & Surface Design, Society for Industrial & Applied Mathematics

Grudic, G.Z. and Lawrence, P.D. (1993). Iterative inverse kinematics with manipulator configuration control, *IEEE Trans. on Robotics and Automation*, Vol.9, No.4, pp. 476-483

Huang, H.; Gong, Z. M.; Chen, X. Q., and Zhou, L. B. (2002). Robotic Grinding/Polishing for Turbine Vane Overhaul. *Journal of Materials Processing Technology*, Volume 127 (2002), pp. 140-145.

Hunt, K.H. (1978). Kinematics Geometry of Mechanisms, Clarendon Press, Oxford

Jun, Bong-Huan, Lee, Jihong, and Lee, Pan-Mook (2006), Repetitive Periodic Motion Planning and Directional Drag Optimization of Underwater Articulated Robotic Arms, *International Journal of Control, Automation, and Systems*, vol. 4, no. 1, pp. 42-52, February 2006

Konjovic,Z., Vukobratovic, M. and Surla, D. (1994). Synthesis of the optimal trajectories for robotic manipulators based on their complete dynamic models, *International Journal of Robotics and Automations*, Vol.9, No.1, pp. 36-47

Kyriakopoulos, K.J. and Saridis, G. N. (1994). Minimum jerk for trajectory planning and control, *Robotica*, 12, pp.109-113

Lee, J.H. (1995). A dynamic programming approach to near minimum-time trajectory planning for two robots, *IEEE Trans. on Robotics and Automations*, Vol.11,No.1, pp.160-164,1995

Leu, M.C., and Singh, S.K. (1989). Optimal planning of trajectories robots, *CAD Based Programming for Sensory Robots*, B. Ravani (ed.), Springer-Verlag

Kim, J. Y. (2004), CAD-Based Automated Robot Programming in Adhesive Spray Systems for Shoe Outsoles and Uppers, *Journal of Robotic Systems*,21(11), 625–634 (2004)

Manocha,D. and Canny, J. F. (1994). Efficient inverse kinematics for general 6R manipulators, *IEEE Trans. on Robotics and Automations*, Vol.10, No.5, pp.648-657

Mitsi,S., Bouzakis, K.-D., and Mansour, G. (1995). Optimization of robot links motion in inverse kinematics solution considering collision avoidance and joint limits, *Mech. Mach. Theory*, Vol.30,No.5, pp.653-663

Makino,H., Xie, C.X. and Zhen, S.X. (1982). Spatial mechanisms and robotic mechanisms, *China Mechanical Industry Press*, Beijing

MATLAB 5.3 Reference Guide, The MathWorks Inc., 1998

Osman Gursoy (1992). Some results on closed ruled surfaces and closed space curves, *Mech. Mach. Theory*, Vol.27,No.3, pp:323-330

Patel, R.V. and Lin, Z. (1988). Collision-free trajectory planning for robot manipulators, *Proceedings of 1988 IEEE Int. Conf. on Systems, Man, and Cybernetics*, Vol.2, pp.787-790

Paul Dierckx (1993), Curve and Surface Fitting with Splines, *Oxford University Press*

Pfeiffer,F. and Johanni, R. (1987). A Concept for manipulator trajectory planning, *IEEE Journal of Robotics and Automation*, Vol.RA-3, No.3, pp.115-123

Pachidis, T.P. Tarchanidis, K.N., Lygouras, J.N., Tsalides, P. G. (2005), Robot Path Generation Method for a Welding System Based on Pseudo Stereo Visual Servo Control, EURASIP *Journal on Applied Signal Processing*, 2005:14, 2268–2280

Ridely,P.R., Bokelberg, E.H. and Hunt, K.H. (1992). Spatial motion-II: acceleration and the differential geometry of screws, *Mech. Mach. Theory*,Vol.27,No.1,pp:17-35

Roth, B. (1967). On the screw axes and other special lines associated with spatial displacements of a rigid body, *ASME Journal of Engineering for Industry*, Series B, Vol.89,No.1,pp.102-110

Seereeram, S. and Wen, J.T. (1995). A global approach to path planning for redundant manipulators, *IEEE Trans. on Robotics and Automation*, Vol.11, No.1, pp.152-160

Shin, K.G. and McKay, N.D. (1985). Minimum-time control of robotic manipulators with geometric path constraints, *IEEE Trans. Automat. And Cont.*, Vol. AC-30, No.6,pp.531-541

Slotine, J.J.E and Yang, H.S. (1989). Improving the efficiency of time-optimal path-following algorithms, *IEEE Trans. Robotics and Automation*, Vol.5, pp.118-124

Shiller,Z., and Lu, H.-H. (1990). Robust computation of path constrained time optimal motions, 1990 *Proc. IEEE Int. Conf. Robotics and Automation*,pp.144-149

Tabarah, E. , Benhabib, B., and Fenton, R.G. (1994). Optimal motion coordination of two robots-a polynomial parameterization approach to trajectory resolution, *Journal of Robotic Systems*, 11(7), pp.615-629

Thompson, S.E. and Patel, R.V. (1987). Formulation of joint trajectories for industrial robots using B-splines, *IEEE Trans. Industrial Electronics*, Vol,.34, No.2, pp.192-199

Yang, T. and Jin, W.M. (1988). Study on the kinematics performance of robot manipulators using the theory of velocity space, *Proceedings of 1988 IEEE Int. Conf. on Systems, Man, and Cybernetics*, Vol.2, pp.1364-1366

Yoshikawa,T. (1985). Manipulability of robotic mechanisms, *Int. J. of Robotics Research*, Vol.4 , No.2, pp. 3-9

Zeid,I. (1993). CAD/CAM Theory and Practice, *McGraw-Hill International Editions*, Computer Science Series

Zha, X.F. (1993). On the harmonious planning and simulation of the trajectories of position and orientation for robot end effector, *Proceedings of the 9th Int. Conf. on Computer Aided Production Engineering*, pp.296-301, Nanjing, China

Zha, X.F., Wang, M., and Choi, A.C.K. (1994). Optimal trajectory planning for robot manipulators, *Proceedings of the 2nd Asian Conference on Robotics and Applications*, International Academics Publishers, pp.569-575, Beijing, China

Zha, X.F., Du, H. (2001). Generation and simulation of robot trajectories in a virtual CAD-based off-line programming environment, *International Journal of Advanced Manufacturing Technology*, Vol.17:610-624

Zha, X.F. (2002). Optimal Pose Trajectory planning for robot manipulators, *Mechanism and Machine Theory*, Vol.37: 1063–1086

Zha, X.F., Chen, X.Q. (2004). Trajectory coordination planning and control for robot manipulators in automated material handling and processing, *International Journal of Advanced Manufacturing Technology*, Vol.23:831-845

13

Optimal Motion Planning for Manipulator Arms Using Nonlinear Programming

Jong-keun Park
Kyungnam University
South Korea

1. Introduction

The optimal motion planning problems for manipulator arms have been actively researched in robotics in the past two or three decades because the optimal motions that minimize actuator forces, energy consumption, or motion time yield high productivity, efficiency, smooth motion, durability of machine parts, etc. These merits are more apparent when the manipulator arms execute repeated motions.
This subject is roughly divided into two categories according to the tasks that the manipulator arms should perform. These categories are characterized by motions as with or without geometric path constraints.
If the geometric path of the end-effector of a non-redundant manipulator is predetermined, the motion has one degree of freedom (DOF) and can be represented by a scalar path variable. In this case, rigorous solutions were obtained subject to constant bounds on the actuator forces (Bobrow et al., 1985; Shin & McKay, 1985). Subsequently, the study was extended to cases where the path included certain singular points on it (Shiller, 1994) or where the actuator jerks and actuator torques were limited within constant bounds (Constantinescu & Croft, 2000).
Most manipulator tasks—except arc welding, painting or cutting—are essentially motions without geometric path constraints. In this case, obstacle avoidance should be considered simultaneously with motion optimization. These types of manipulator motions have the same DOF as the number of lower pair joints, and the corresponding optimal motions are more complicated than those discussed above. The subject of this chapter lies in this category.
Various types of methods have been developed to solve the optimal motion planning problem in the presence of obstacles. Optimal control theory (Bryson & Meier, 1990; Bessonnet & Lallemand, 1994; Formalsky, 1996; Galicki, 1998), nonlinear programming (Fenton et al., 1986; Bobrow, 1988; Singh & Leu, 1991), dynamic programming (Jouaneh et al., 1990), tessellation of joint (Ozaki & Lin, 1996) or configuration space (Shiller & Dubowsky, 1991), and a combination of these (Schlemmer & Gruebel, 1998; Hol et al., 2001) are the main techniques used.
By the application of the optimal control theory, Pontryagin's maximum principle leads to a two-point boundary value problem. Some researchers have attempted to solve these equations directly (Bryson & Meier, 1990; Formalsky, 1996) while others have attempted to

solve them through parameter optimization (Hol et al., 2001). Although this theory and its solutions are rigorous, it has been used to solve equations for the motions of 2-link or at most 3-link planar manipulators due to the complexity and the nonlinearity of the manipulator dynamics.

Approximation methods have been studied to obtain the solutions for three or more DOF spatial manipulators; however, the solutions obtained have not been proved to be optimal. These approximation methods are roughly divided into two groups depending on whether or not they utilize gradients.

Most algorithms based on nonlinear programming use gradients (Fenton et al., 1986; Bobrow, 1988; Singh & Leu, 1991; Bobrow et al., 2001; Wang et al., 2001). For stable convergence, the objective functions and constraints must be locally convex and their first derivatives must be continuous. Numerically calculated gradients have been used to find minimum time motions (Bobrow, 1988); however, the simulation model was a 3-link spatial manipulator in the presence of a relatively simple obstacle model. Subsequently, analytically calculated gradients were used to minimize actuator torques for various multibody systems (Bobrow et al., 2001) or spatial 6-link manipulators (Wang et al., 2001). However, torque or energy minimizations show more stable convergence properties than the minimum time motions because the motion time is fixed.

Other approximation methods that do not utilize gradients are mainly based on (1) approximations in small time-intervals (Singh & Leu, 1991; Jouaneh et al., 1990; Hol et al., 2001) or (2) discretization/tessellation (Ozaki & Lin, 1996; Shiller & Dubowsky, 1991; Schlemmer & Gruebel, 1998) of joint or configuration spaces. The former requires less CPU time but may accumulate numerical or modeling errors in small time-intervals and thus lower the accuracy of the results. The latter assures stable convergence but the CPU time may increase exponentially for the refinement of tessellation.

Because of the complex dynamics and kinematics of robot manipulators, various assumptions or simplifications were introduced for the online implementation or simplification of the algorithms.

Using geometric simplifications, obstacles have been ignored (Bessonnet & Lallemand, 1994; Fenton et al., 1986; Jouaneh et al., 1990; Hol et al., 2001; Lee, 1995) or modeled as circles (Galicki, 1998; Singh & Leu, 1991; Ozaki & Lin, 1996) or as finite surface points (Schlemmer & Gruebel, 1998), and robot links have been modeled as lines (Galicki, 1998; Ozaki & Lin, 1996; Lee, 1995), finite surface points (Singh & Leu, 1991), or ellipsoids (Schlemmer & Gruebel, 1998).

Using kinematic simplifications, motions were restricted in a plane (Formalsky, 1996; Galicki, 1998; Ozaki & Lin, 1996; Shin & Zheng, 1992; Lee, 1995), the orientation of the end-effector was ignored (Singh & Leu, 1991; Shiller & Dubowsky, 1991), or the joint velocity profiles (Fenton et al., 1986) or joint acceleration profiles (Jouaneh et al., 1990; Schlemmer & Gruebel, 1998; Cao et al., 1998) were pre-specified.

Using dynamic simplifications, manipulator dynamics were ignored subject only to the kinematic constraints (Fenton et al., 1986; Shin & Zheng, 1992; Cao et al., 1998).

To the best of the author's knowledge, the optimal motions for six or more DOF manipulators or multiple robot arms have not yet been obtained without simplifying any of the geometric, kinematic, or dynamic models of manipulators or obstacles.

In this study, we transform the optimal control problem in a function space into a nonlinear programming problem in a finite-dimensional vector space. Joint displacements are

represented by the linear combinations of finite-term quintic *B*-splines. If a sufficient number of terms are used an exact solution will be obtained. Using numerically calculated gradients, the optimal coefficients of the splines are obtained.
The novel contribution made by this study is the concept of the *minimum overload trajectory* with fixed total motion time. The minimum time motions are defined rigorously by this concept and they are found successfully by the sequential searches for minimum overload trajectories. In the minimum overload searches, the convergence is quite stable because the performance index and all the constraints are locally convex and smooth and the total motion time is fixed.
To compute the minimum overload trajectory, the total motion time is initially specified to be very small so that the actuators require more force than they can produce. Then, using an efficient numerical optimization, the actuator overloads are minimized during the motion. Using the information obtained from the minimum overload trajectory, we predict the motion time of the next minimum overload search. These successive searches continue until we find the least time at which the minimum overload vanishes.
Obstacle information is evaluated by penetration growth distances (Ong & Gilbert, 1996), and obstacle avoidance is achieved by incorporating a penalty term that is included in an augmented performance index. The usefulness of the penetration growth distance will be shown in the simulation results.
The complete geometric, kinematic, and dynamic models of a spatial 6-link manipulator and the obstacles in its path are considered in this study. The effects of friction are the only variables that are ignored.
The manipulator dynamics are calculated by the outward and inward iteration method (Craig, 1986). This iteration method requires only joint-by-joint recursive calculations and accumulates a slight numerical error. In addition, it can be applied without knowing the equations of motions explicitly in higher DOF models.
In most of the other studies, the constraints on the actuator forces are constant, regardless of joint velocities. It is more practical for the bounds of the actuator forces to be dependent on the joint velocities, as was done in this study.
Trial applications to a spatial 3-link and a 6-link Puma 560 manipulator in the presence of polyhedral obstacles demonstrate the effectiveness and numerical stability of this algorithm.

2. Formulations of Optimal Motions

2.1 Actuator Characteristics

We consider two different actuator characteristics as shown in Fig. 1: One is that the actuator force limits depend on the joint velocities as shown in Fig. 1(a); the other is that the two limits are constant as shown in Fig. 1(b).
In most practical cases, the limits of the actuator forces are dependent on the joint velocities. In this case, the constraints on the actuators are as follows:

$$|\tau_i| \le -\frac{\tau_i^c}{\omega_i^c}|\omega_i| + \tau_i^c, \quad i = 1, \cdots, n \tag{1}$$

where τ_i and ω_i are the generalized actuator force and joint velocity of the *i*-th joint, respectively; τ_i^c, ω_i^c are the absolute values of their limits; *n* is DOF of the system, which is equal to the number of lower pair joints.
If we define *equivalent actuator forces* τ_i^e as

$$\tau_i^e = |\tau_i| + \frac{\tau_i^c}{\omega_i^c}|\omega_i|, \quad i = 1, \cdots, n \tag{2}$$

then the actuator constraints (1) become

$$\tau_i^e \le \tau_i^c, \quad i = 1, \cdots, n \tag{3}$$

If the limits are constant as shown in Fig. 1(b), the actuator and velocity constraints can be simply expressed as:

$$|\tau_i| \le \tau_i^c, \quad i = 1, \cdots, n \tag{4}$$

$$|\omega_i| \le \omega_i^c, \quad i = 1, \cdots, n \tag{5}$$

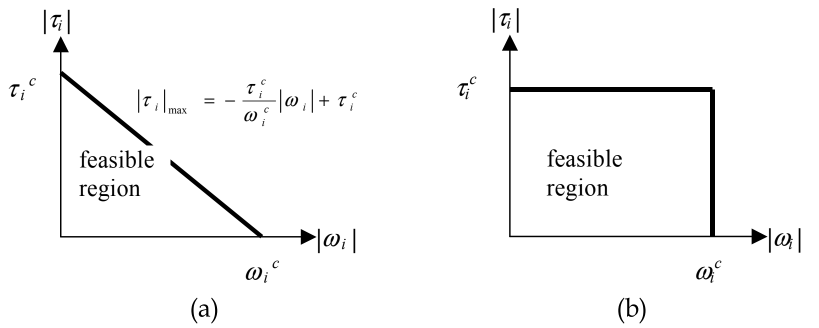


Fig. 1. Two kinds of actuator characteristics; (a) actuator force limit depends on joint velocity, (b) the limits of actuator force and joint velocity are constant.

2.2 Optimization in a Function Space

The equations of motions for a manipulator arm are as follows:

$$\tau = \mathbf{M}(\theta)\alpha + \mathbf{v}(\theta, \omega) + \mathbf{g}(\theta) \tag{6}$$

where $\boldsymbol{\tau}$ denotes the generalized actuator forces ($n \times 1$); $\mathbf{M}$ is the inertia matrix ($n \times n$); $\boldsymbol{\theta}$, $\boldsymbol{\omega}$, and $\boldsymbol{\alpha}$ are the generalized joint displacements, velocities, and accelerations ($n \times 1$), respectively; $\mathbf{v}$ is the centrifugal and coriolis forces ($n \times 1$); and $\mathbf{g}$ is the gravitational force ($n \times 1$).
In this study, four different performance indices are minimized. These are as follows:

$$J_{d1} = \frac{1}{T}\int_0^T \sum_{i=1}^n \left(\frac{\tau_i}{\tau_i^c}\right)^2 dt \tag{7}$$

$$J_{d2} = \frac{1}{T}\int_0^T \sum_{i=1}^n \left(\frac{\tau_i \omega_i}{\tau_i^c \omega_i^c}\right)^2 dt \tag{8}$$

$$J_{d3} = \frac{1}{T}\int_0^T \sum_{i=1}^n \left\{\frac{\tau_i^e}{\tau_i^c} - 1\right\}_+^2 dt \tag{9}$$

$$J_{d4} = \frac{1}{T}\int_0^T \sum_{i=1}^n \left[\left\{\frac{\tau_i}{\tau_i^c} - 1\right\}_+^2 + \left\{\frac{\omega_i}{\omega_i^c} - 1\right\}_+^2\right] dt \tag{10}$$

where T is the total motion time and the plus operator $\{\bullet\}_+$ is defined as

$$\{\bullet\}_+ = \begin{cases} \bullet, & if \bullet \geq 0 \\ 0, & if \bullet < 0 \end{cases} \tag{11}$$

(7) and (8) are the performance indices for the minimum torque and minimum energy motions, respectively.
Since τ_i^c and ω_i^c in (7) and (8) simply act as weighting factors of each joint, they do not ensure that the actuator forces and the joint velocities do not exceed their limits. (9) and (10) are the performance indices for the *minimum overload motions*. "Overload" implies that the actuator forces or the joint velocities exceed their limits. (9) is the case in which the actuator force limits depend on the joint velocities as shown in Fig. 1(a). On the other hand, (10) is the alternate case where the two limits are constant as shown in Fig. 1(b).
If the total motion time is greater than or equal to the minimum time, the minimum values of (9) and (10) must be zero. On the other hand, if it is less than the minimum time, they must be positive. In Section 4, the minimum time motions are defined rigorously by this concept and they are found successfully by the sequential searches for the minimum overload trajectories.
To formulate the obstacle avoidance constraints, we use the so-called *growth function* (Ong & Gilbert, 1996). We briefly review the approach here. Assume that there exists a convex *object* **A** in a three-dimensional workspace. The *object* is defined as a set of all the points inside and on the surface of a rigid body. Let $\mathbf{p}_A$ be an arbitrary point (*seed point*) fixed inside **A**, then the *growth model* $\mathbf{G}_A(\sigma)$ is defined as

$$\mathbf{G}_A(\sigma) = \{\mathbf{y} \mid \mathbf{y} = \mathbf{p}_A + \sigma(\mathbf{x} - \mathbf{p}_A),\ \mathbf{x} \in \mathbf{A}\} \tag{12}$$

where σ is a non-negative scalar.
Consider another convex object **B** in the same workspace and let $\mathbf{G}_B(\sigma)$ be the growth model of **B** wrt. a seed point $\mathbf{p}_B$ fixed inside **B**, then the growth function $\sigma^*(\mathbf{A}, \mathbf{B})$ is defined as follows:

$$\sigma^*(\mathbf{A}, \mathbf{B}) = \min\ \{\sigma \mid \mathbf{G}_A(\sigma) \cap \mathbf{G}_B(\sigma) \neq \emptyset\} \tag{13}$$

The growth function can be calculated by linear programming if **A** and **B** are convex polyhedra. The dimension of this linear programming problem is 4; thus, the active set method (Best & Ritter, 1985) is efficient for such a low- dimensional LP problem.
Consider that there are m obstacles in a workspace. We assume that all the obstacle models $\mathbf{O}_1,\ldots,\mathbf{O}_m$ and the link models $\mathbf{R}_1(t),\ldots,\mathbf{R}_n(t)$ are convex polyhedra. However, non-convex models are permissible if they can be decomposed into multiple convex models. If the growth function σ^* of a link model $\mathbf{R}_i$ and obstacle model $\mathbf{O}_j$ has a value less than one, one model penetrates into the other and the following *penetration growth distance* D_{ij} indicates the extent of the penetration.

$$D_{ij} = (d_i + d_j)\{1 - \sigma^*(\mathbf{R}_i, \mathbf{O}_j)\}_+ \tag{14}$$

where d_i and d_j are the appropriate positive real numbers that represent the actual geometric sizes of $\mathbf{R}_i$ and $\mathbf{O}_j$, respectively, and the plus operator has been defined above. In general, the penetration growth distance is not equal to the minimum translational distance separating the objects.
From the above notation, the obstacle avoidance constraints become

$$\mathbf{D} = \mathbf{0}, \quad \forall t \in [0, T] \tag{15}$$

where **D** is a matrix ($n \times m$) whose elements are (14).
To obtain obstacle-free optimal motions, we define the following augmented performance indices as

$$J_i = J_{di} + w_o J_o, \quad i = 1, \cdots, 4 \tag{16}$$

where w_o is a sufficiently large weighting coefficient and the obstacle term is

$$J_o = \frac{1}{T} \int_0^T \sum_{i=1}^{n} \sum_{j=1}^{m} \left(D_{ij} \right)^2 dt \tag{17}$$

Motions at the start and goal positions are specified as

$$\theta(0) = \theta_s, \quad \theta(T) = \theta_f \tag{18}$$

$$\omega(0) = \omega_s, \quad \omega(T) = \omega_f \tag{19}$$

$$\alpha(0) = \alpha_s, \quad \alpha(T) = \alpha_f \tag{20}$$

The acceleration conditions (20) are given to assure smooth motions at the start and goal positions.
The obstacle-free optimal motion planning problem can be stated as–

Find $\theta(t)$ (n×1) that minimize (16) subject to (6), (18) - (20) (21)

The optimal motion planning problem (21) is transformed into a finite-dimensional nonlinear programming problem in the following section.

Nodes(s)	j-3	j-2	j-1	j	j+1	j+2	J+3
$B_j(s)$	0	1/120	26/120	66/120	26/120	1/120	0
$B_j'(s)$	0	1/24	10/24	0	-10/24	-1/24	0
$B_j''(s)$	0	1/6	2/6	-6/6	2/6	1/6	0

Table 1. Nodal values of $B_j(s)$ and its derivatives.

2.3 Optimization in a Finite-Dimensional Vector Space

An infinite number of linearly independent basis functions form a complete set in a function space and this set can represent an arbitrary piece-wise continuous function defined on a closed interval. A finite number of these functions can express a piece-wise continuous function approximately. Many researchers have used cubic B-splines as the basis functions. However, in this research, quintic B-splines have been used.
Both splines play almost the same role when they are used in robot motion planning. The trajectories expressed by quintic B-splines, despite a larger computational burden, have the following merits: 1) Accelerations and jerks are third and second order polynomials, respectively and are therefore continuous; 2) they can express various types of displacement functions more accurately; 3) they have a wider range of feasible directions if used in nonlinear programming; and 4) for a given number of basis functions, the value of the optimal performance indices is usually less than that found using cubic B-splines.
The quintic B-spline used in this research is (Prenter, 1975)

$$B_j(s) = (1/120)[\{s-(j-3)\}_+^5 - 6\{s-(j-2)\}_+^5 + 15\{s-(j-1)\}_+^5 - 20\{s-j\}_+^5 \\ +15\{s-(j+1)\}_+^5 - 6\{s-(j+2)\}_+^5 + \{s-(j+3)\}_+^5], \quad s\in[-\infty,\infty] \tag{22}$$

where j is an arbitrary integer. The basis function $B_j(s)$ is positive for $j - 3 < s < j + 3$ and zero otherwise. The nodal values of (22) and its s-derivatives are listed in Table 1.

If we choose $[0, k]$ as the interval of the parameter s to express the manipulator motions in the time interval $[0, T]$, then $k + 5$ splines i.e., $B_{-2}(s),\ldots,B_{k+2}(s)$ have nonzero values in $[0, k]$. The joint trajectories are expressed by the linear combinations of the $k + 5$ splines as follows:

$$\theta(s) = \mathbf{C}\,\mathbf{B}(s) \tag{23}$$

$$s = \beta t \tag{24}$$

where s is a dummy variable connecting joint variables with time; $\mathbf{C}$, a coefficient matrix ($n \times k + 5$); $\mathbf{B}(s)$, a column vector ($k + 5 \times 1$) whose elements are $B_{-2}(s),\ldots,B_{k+2}(s)$; and β ($=k/T$), a time-scale factor that defines the motion time. Thus, k remains constant although the total motion time varies.

Differentiating (23) wrt. time

$$\omega(s) = \beta\,\mathbf{C}\mathbf{B}'(s) \tag{25}$$

$$\alpha(s) = \beta^2\,\mathbf{C}\mathbf{B}''(s) \tag{26}$$

where the primes (', '') imply differentiation wrt. s.

The initial and final motion conditions (18)–(20) can be used to reduce the dimension of the coefficient matrix. By algebraic manipulation (Park & Bobrow, 2005), the joint trajectories satisfying the initial and final motion conditions are written as follows:

$$\theta(s) = \mathbf{F}_s(s) + \mathbf{C}_m\mathbf{B}_m(s) + \mathbf{F}_f(s) \tag{27}$$

where the boundary condition splines $\mathbf{F}_s(s)$ and $\mathbf{F}_f(s)$, each of dimension ($n \times 1$), can be determined from the initial and final motion conditions. $\mathbf{C}_m$ is *the reduced coefficient matrix* ($n \times k - 1$) and $\mathbf{B}_m(s)$ is *the reduced B-spline basis function* ($k - 1 \times 1$).

The velocities and accelerations are

$$\omega = \beta\left[\mathbf{F}_s'(s) + \mathbf{C}_m\mathbf{B}_m'(s) + \mathbf{F}_f'(s)\right] \tag{28}$$

$$\alpha = \beta^2\left[\mathbf{F}_s''(s) + \mathbf{C}_m\mathbf{B}_m''(s) + \mathbf{F}_f''(s)\right] \tag{29}$$

(27) together with (24) implies that the arbitrary joint trajectories subject to the initial and final motion conditions are represented approximately by a point in an $n(k - 1)$-dimensional linear vector space spanned by the reduced coefficient matrix $\mathbf{C}_m$.

The obstacle-free optimal motion planning problem (21) can be stated as–

$$\textbf{Find } \mathbf{C}_m \in \mathcal{R}^{n(k-1)} \textbf{ that minimizes (16) subject to (6), (27)-(29)} \tag{30}$$

3. Nonlinear Programming using a Quasi-Newton Method

3.1 BFGS Method

If the initial and final motion conditions are given, we can calculate the objective functional (16) by assigning arbitrary specific values to $\mathbf{C}_m$. Along the successive search directions determined by the BFGS algorithm (Fletcher, 1987), we can find the optimal $\mathbf{C}_m^*$.

The first convergence criterion is

$$\left(J_i^{*j-1} - J_i^{*j}\right) / J_i^{*j-1} \le \varepsilon_1 \ , \quad i = 1, \cdots, 4 \tag{31}$$

where J_i^{*j} is the minimum value of (16) at the end of the jth line search.
If (31) is satisfied, the following condition is tested to terminate the process:

$$J_o \le \varepsilon_2 \tag{32}$$

The algorithm is:

1. Choose the number k that is at the end of s-interval and choose the total motion time T.
2. Using the motion conditions (18)–(20), calculate the coefficients of the boundary condition splines $\mathbf{F}_s(s)$ and $\mathbf{F}_f(s)$ in (27) (Park & Bobrow, 2005).
3. Determine the initial values of $\mathbf{C}_m$ (Park & Bobrow, 2005).
4. Divide the s-interval $[0, k]$ into l equal subintervals, where, $3k$–$5k$ is appropriate as l.
5. At l +1 nodal points including the two end points, calculate
   5.1. Joint displacements, joint velocities, and joint accelerations (27)–(29), where the s-derivatives are calculated analytically beforehand.
   5.2. Actuator forces (6), which are calculated by outward and inward iteration method (Craig, 1986), and equivalent forces (2).
   5.3. Penetration growth distances (14) between individual link models and individual obstacle models.
6. Calculate one of the four performance indices (7)–(10) and (17) by the trapezoidal integral formula.
7. Determine the initial value of w_o in (16) so that the second term of (16) is about ten times as much as the first term.
8. Calculate the gradient of (16) wrt. $\mathbf{C}_m$ numerically.
9. Determine the search direction by the BFGS method.
10. Perform the line search by the golden section search method.
11. If (31) is satisfied, go to next step; otherwise, go to Step 8.
12. If (32) is satisfied, terminate the process; otherwise, increase w_o by about 10 times and go to Step 8.

The gradient in Step 8 is calculated by the central difference method as follows:

$$(\mathbf{G})_{ij} \cong \frac{J\left((\mathbf{C}_m)_{ij} + \delta\right) - J\left((\mathbf{C}_m)_{ij} - \delta\right)}{2\delta} \ , \quad i = 1, \cdots, n \ , \quad j = 1, \cdots, k-1 \tag{33}$$

The B-spline basis functions have certain merits. Since a B-spline has a nonzero value in a small interval, the increment of the objective functional (16) in the small interval is the only data required to calculate the corresponding component of the gradient.
The radii of the minimum circumscribed spheres of the two models are appropriated as the values of the parameters d_i and d_j in (14). k in Step 1 has an effect on the accuracy of the joint trajectories and l in Step 4 has an effect on the accuracy of the numerical integrations. 15 or 20 is assigned to k. 10^{-12}, 10^{-7}, and 10^{-7} are assigned to ε_1, ε_2, and δ, respectively, in the double precision numerical process. All programs have been written in FORTRAN, not using any type of package programs that include IMSL libraries. It takes about ten minutes in the Hewlett Packard workstation x2000 to obtain the optimal motions for a Puma 560 type of manipulator arm in the presence of one hexahedral obstacle.

3.2 Global Search in an Obstacle Field

The numerical optimization described above can be modified to search for the global optimum motions in a given obstacle field. The modification involves repeating the implementations mentioned in Section 3.1 while changing the positions of the seed points of the obstacles. If a link model penetrates into an obstacle model during the optimization process, the path is updated so that the Euclidean distance between the two seed points of the link model and the obstacle model increases. If we locate the seed point in an inner corner of the obstacle model, the link tends to avoid the obstacle by turning around the opposite vertex. By changing the locations of the seed points, we can find all the local minima and choose the global one that has the least minimum performance indices. Although we have no proof that this method will always produce the global minima, it has done so for all of the examples we have solved, including those discussed in this chapter.

4. Minimum Time Motions

If the models of a manipulator and obstacles, and the actuator characteristics are specified, the minimum overloads J_{d3}^*, J_{d4}^* are functions of the total motion time T. Thus, we can define the minimum times as follows:

Definition (Minimum times)

The minimum times $T_i^* \equiv \min \{ T \mid J_{di}^*(T) = 0 \}$, $i = 3, 4$.

Theorem

For an obstacle-free point-to-point manipulator motion, if $T < T_i^*$, then the minimum overloads $J_{di}^*(T) > 0$, $i = 3, 4$. If $T \geq T_i^*$, then $J_{di}^*(T) = 0$, $i = 3, 4$.

Proof

For $T < T_i^*$ and the fact that $J_{di}(T)$ in (9) and (10) are non-negative, the above definition implies that $J_{di}^*(T) > 0$, $i = 3, 4$. For $T \geq T_i^*$ and the fact that $J_{di}^*(T)$ in (9) and (10) are monotonically decreasing functions of T, $J_{di}^*(T_i^*) = 0$ and it must remain zero for all $T \geq T_i^*$, $i = 3, 4$.

The theorem shows that as we increase the motion time T, starting from a time less than T^*, at some point we will achieve T^* if such a time exists. Thus, a simple line search can be used to find T^*. However, we achieved superior performance with the heuristic algorithm (Park & Bobrow, 2005). Detailed explanations are omitted in this chapter. The heuristic algorithm is not the only method but the problem of efficiency to find T^*.

5. Simulations

5.1 Example 1 (Spatial 3-Link Manipulator)

The model is a 3-link arm shown in Fig. 2, where all joints are revolute pairs around their z axes, and it is a configuration at zero-displacement. The base coordinates are the same as the first link coordinates. The specifications are listed in Table 2; in this case, gravity acts in the z_0 direction and $\boldsymbol{\tau}_c$ is about twice the static actuator torque necessary to endure gravity in the fully stretched configuration. The sizes of the two hexahedral obstacles are the same and their dimensions are (0.4, 0.4, 0.5) m and the centers are (0.76, –0.47, –0.25) and (0.76, 0.47, 0.25) m in base coordinates. The orientations of the obstacles are equal to the base coordinates. The seed points of all the links are at their geometric centers, but those of all the obstacles are located in the corners in order to find the various local paths. The manipulator moves from (–60°, 30°, -60°) to (60°, –30°, –120°) in the joint space. The velocities and accelerations at the two end points are zero.

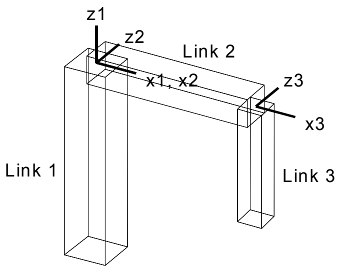


Fig. 2. Spatial 3-link manipulator.

Links	Masses	l_x, l_y, l_z (in link coor.)	τ_c	ω_c
1st	100	0.2, 0.2, 1.0	1000	6
2nd	50	0.8, 0.15, 0.15	1000	6
3rd	30	0.12, 0.6, 0.12	200	6

Table 2. Specifications of Example 1 (SI units).

We repeated the global search for the minimum overload trajectories, mentioned in Section 3.2, starting from T = 0.6 s and increasing it by 0.01 s; where J_3 in (16) is minimized.
Fig. 3 shows the convergence stability, where the minimum overloads decrease monotonically until they vanish at the minimum times. The convergences are therefore quite stable. This figure also demonstrates that we can find the global optimal trajectory by adjusting the seed points of the obstacles. To check the global search clearly, we aligned the obstacles exactly in the path of the initial motion. We have found four local minima shown in Fig. 4 and we can observe that the local minimum (A) is the global one whose minimum overload is the least among the four.

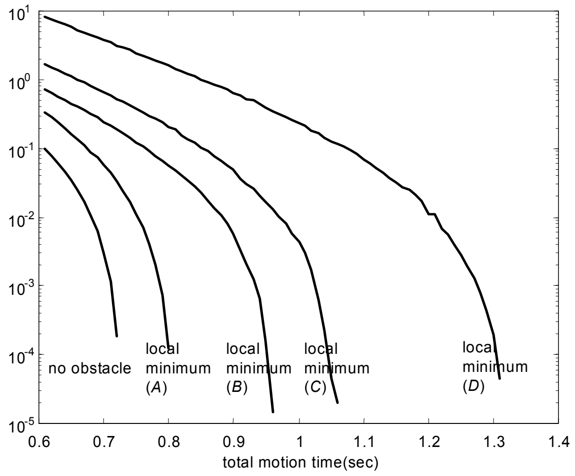
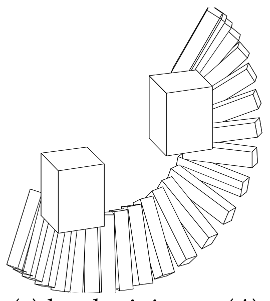
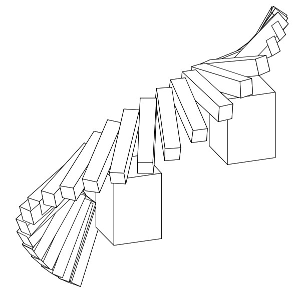
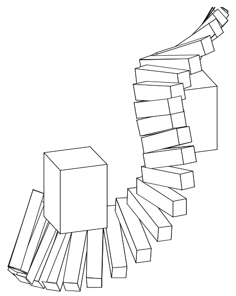


Fig. 3. Minimum overloads J_{d3}^* in Example 1.


(a) local minimum (A)


(b) local minimum (B)


(c) local minimum (C)

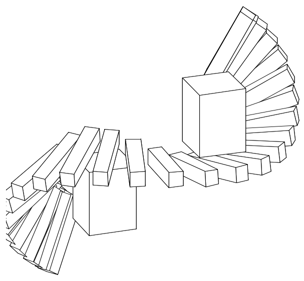

(d) local minimum (D)

Fig. 4. Local minimum trajectories of last link in Example 1

The minimum times are 0.728 s when ignoring the obstacles and 0.807 s, 0.961 s, 1.08 s and 1.32 s in the local minima (A)–(D), respectively. In the case of no obstacles (not shown in Fig. 4) the manipulator turns around the z_1 axis with Link 3 bent downward to reduce the moment of inertia about that axis and the moment arm of gravity about the z_2 and z_3 axes.
In the optimal motions, the manipulator slightly touches the surfaces of the obstacles; this may be considered as imperfect obstacle avoidance. The minimum clearance to assure safe avoidance must be added to the actual geometric sizes of the obstacles.

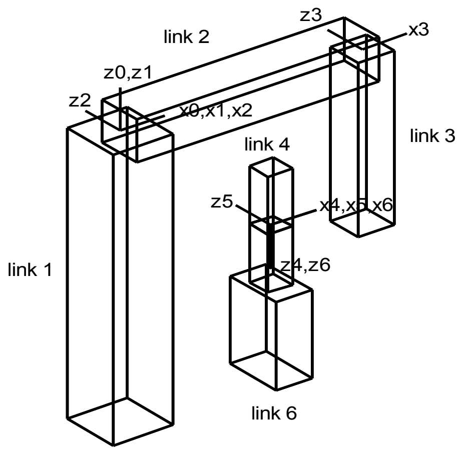


Fig. 5. Spatial 6-link manipulator.

Links	a_{i-1}	a_{i-1}	d_i	θ_i
1st	0	0	0	θ_1
2nd	-90°	0	0	θ_2
3rd	0	0.8	0	θ_3
4th	-90°	0	0.8	θ_4
5th	90°	0	0	θ_5
6th	-90°	0	0	θ_6

Table 3. Link parameters in Denavit-Hartenberg notation of Example 2 (SI units).

5.2 Example 2 (Spatial 6-Link Manipulator)

Fig. 5 shows a configuration of a PUMA 560 type of manipulator at zero-displacement. All joints are revolute pairs around their *z* axes.

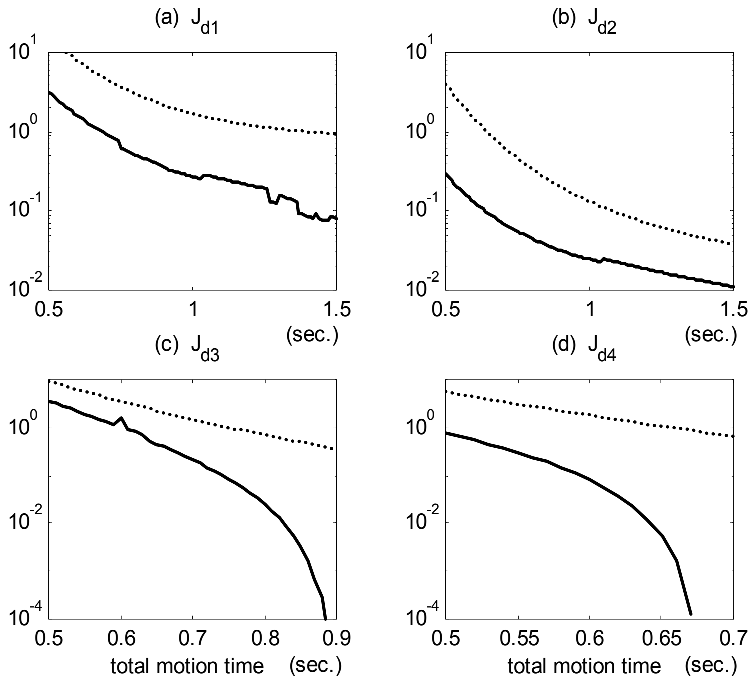


Fig. 6. Initial (upper curves) and minimum (lower curves) performance indices vs. total motion time in Test 1 of Example 2.

The base coordinates are the same as the first link coordinates at zero-displacement. Link 4 is connected to Link 3. The link parameters in the Denavit-Hartenberg notation and the specifications are listed in Table 3 and 4, respectively; in this case, gravity acts in the – z_0 direction and τ_c is about twice the static actuator torque necessary to endure gravity in the fully stretched configuration. The mass of the last link includes that of a tool and is heavier than Link 5. The dimensions of one hexahedral obstacle are (1.2, 2.0, 1.2) m and the center is (1.2, 0.0, 0.0) m in base coordinates. The orientation of the obstacle is equal to the base coordinates. The velocities and accelerations at the start and goal positions are all zero.

Links	Mass	l_x, l_y, l_z (in link coor.)	τ_c	ω_c
1st	100	0.2, 0.2, 1.0	1500	6
2nd	50	0.8, 0.15, 0.15	1500	6
3rd	30	0.12, 0.6, 0.12	500	6
4th	5	0.08, 0.08, 0.2	75	6
5th	5	0.08, 0.2, 0.08	75	6
6th	10	0.12, 0.2, 0.3	5	6

Table 4. Specifications of Example 2 (SI units).

Test 1

The manipulator moves from (–20°, 60°, –120°, 0°, –30°, 0°) to (20°, –60°, –60°, 0°, 30°, 0°) in the joint space without considering the obstacle. 20 and 80 are assigned to k and l, respectively. The initial joint trajectories are fifth order polynomials that satisfy (18)–(20) (Park & Bobrow, 2005).

To check the stability of convergences and the reliability of solutions, the author repeated the optimization process while increasing the total motion time. Fig. 6 shows the initial (upper curves) and minimum (lower curves) performance indices vs. the total motion time. It is expected that all the four minimum performance indices must decrease gradually as the total motion time increases. However, as shown in this figure, the minimum torque motions (figure (a)) are unstable, especially when the total motion time is greater.

There are various local minimal motions according to the directions in which the links bend. The joint displacements of the minimum torque motions are greater than those of the other three optimal motions and they increase with the total motion time. Thus, the minimum torque motions may converge to different local minimal motions, thus yielding an unstable convergence. On the other hand, the motions that minimize J_{d4} (figure (d)) show the most stable convergences. Fig. 6(c) and 6(d) show that the minimum overloads decrease gradually until they vanish at the minimum times 0.89 s for J_{d3} and 0.67 s for J_{d4}.

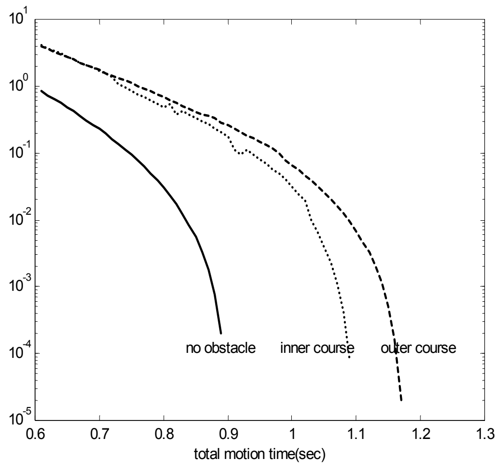


Fig. 7. Minimum overloads J_{d3}^* in Test 2 of Example 2.

Test 2

The manipulator moves from (20°, 60°, –120°, 0°, –30°, 0°) to (–20°, –60°, –60°, 0°, 30°, 0°) in the joint space in the presence of the obstacle described above; where J_3 in (16) is minimized.

Similar to Example 1, we repeated the global search at every time step. The results are shown in Fig. 7. When ignoring the obstacle, the minimum overload decreases gradually until it vanishes at the minimum time 0.90 s. We can see that the convergence is quite stable and it converges to only one optimal motion, regardless of the total motion time. On the other hand, when considering the obstacle, the convergence property in Fig. 7 is not as stable as that in Fig. 3. They converged to different local minima, which will be shown in Fig. 9–12.

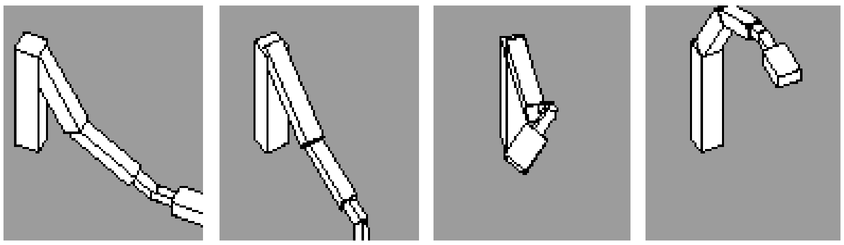


Fig. 8. Minimum time motion ignoring the obstacle.

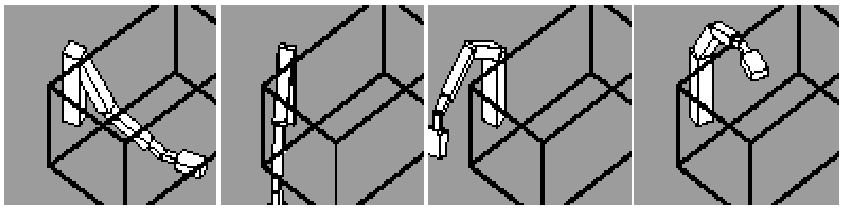


Fig. 9. Local minimum trajectory (*E*) in inner course.

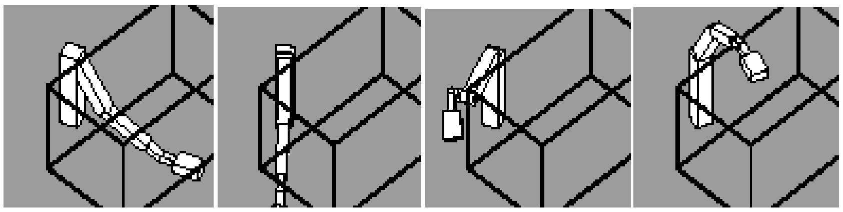


Fig. 10. Local minimum trajectory (*F*) in inner course.

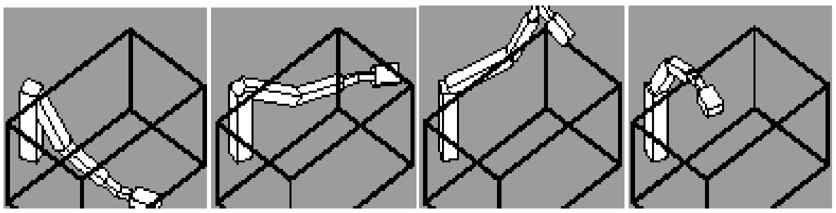


Fig. 11. Local minimum trajectory (*G*) in outer course.

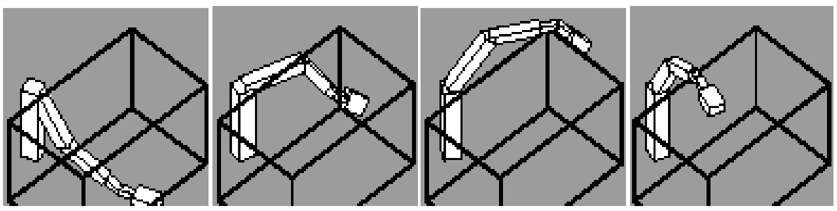


Fig. 12. Local minimum trajectory (*H*) in outer course.

Fig. 8 depicts the minimum time motion, ignoring the obstacle; in this case, the four frames are frames 1, 6, 11, and 16 among the total 16 equal time-interval frames. We can observe in this motion that the manipulator turns around the z_1 axis with the last link bent downward to reduce the moment of inertia about that axis and the moment arm of gravity about the z_2 and z_3 axes. We can also observe that the manipulator maximizes the joint coupling effect by the "overactions" of the "underloaded" joints to reduce the torques of the "overloaded" joints. Here, Joint 1 is "underloaded" and Joints 2 and 3 are "overloaded." We will see these phenomena in Fig. 13 as well. We can see the "overaction" of Joint 1 at the eleventh frame in Fig. 8; that is, Joint 1 turns left over the goal position.

By adjusting the location of the seed point of the obstacle, we have found two obstacle-free courses, namely, the inner and outer courses as shown in Fig. 9–12, where the four frames are frames 1, 6, 11, and 16 among the total 16 equal time-interval frames.

In the inner course, the local minimum shown in Fig. 9 has less minimum overload than Fig. 10. We can observe this fact in Fig. 7, where the curve representing the inner course appears to be composed of several segments of two parallel curves. The difference in the minimum overloads between the two parallel curves becomes smaller as the total motion time approaches the minimum time. Both the minimum times are equal—1.12 s.

In the outer course, the local minimum shown in Fig. 11 has less minimum overload than Fig. 12. The difference is quite small in Fig. 7, but it becomes larger as the total motion time approaches the minimum time. The minimum times in Fig. 11 and 12 are 1.17 and 1.21 s, respectively.

Fig. 13 shows the saturation state of the actuators. Joints 2, 3, and 5 are overloaded in the initial motions (dotted lines), where the maximum equivalent torques exceed twice their actuator limits. After the optimization, the motion time is increased from 0.8 s to 0.90 s in the case of no obstacle and is increased to 1.12 s to avoid the obstacle. Moreover, all the joints are close to saturation during the minimum time motions (solid lines). This is consistent with Pontryagin's maximum principle since a necessary condition for the minimum time motion (assuming no singular arcs) is that all the joints should be in saturation during the motions.

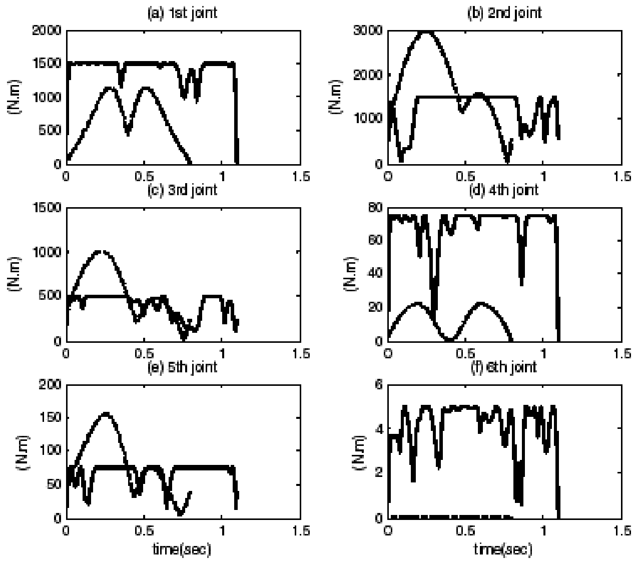


Fig. 13. Equivalent torques in Test 2 of Example 2; solid lines are minimum time motions in local minimum (E) and dotted lines are initial motions, T = 0.8 s.

Our algorithm is not tailored to obstacles moving in dynamic environments. However, if the motions of the obstacles are given as functions of time, we can solve the minimum time problem in exactly the same manner as in this chapter. In the case where obstacles are moving in a dynamic and unpredictable manner in the environment, there is really no method to optimally plan motions around them. Khatib's "elastic bands" could potentially handle this situation, but the solution would be suboptimal.
In future works, we will consider the frictional effects in this method. Our method cannot directly handle Coulomb friction because it creates a discontinuity in the gradients. If the frictional effects are considered successfully, a more effective control method could be found to track this optimal trajectory.

6. Conclusions

In this chapter, we present a practical and reliable method for finding the minimum torque, minimum energy, minimum overload, and minimum time motions for the manipulators moving in an obstacle field, subject to the limits of velocity-dependent actuator forces. Arbitrary point-to-point manipulator motions are represented by a point in a finite-dimensional vector space parameterized by quintic *B*-splines.
The novel idea in this work is the concept of the minimum overload trajectory, in which the actuator-overloads achieve their minimum values with the total motion time fixed. The minimum time motion is defined rigorously with this concept and it is obtained by successive searches for the minimum overload trajectory.
There are various local minimal motions according to the directions in which the manipulator links bend. We can perform global searches in a certain obstacle field by adjusting the locations of the *seed points* of the obstacle models or the link models.
In the resultant optimal motions, the manipulator turns with the last link bent inward in order to reduce the moment of inertia about the z_1 axis and the moment arms of gravity about the z_2 and z_3 axes.
In the resultant minimum time motions, 1) the manipulator maximizes the joint coupling effect by the overactions of underloaded joints to reduce the torques on the overloaded joints, and 2) almost all the actuators are close to saturation during the motion, which is consistent with Pontryagin's maximum principle.

7. References

Bessonnet, G. & Lallemand, J. P. (1994). On the optimization of robotic manipulator trajectories with bounded joint actuators or joint kinetic loads considered as control variables, *Trans. ASME J. Dynamic Systems Measurement and Control*, Vol. 116, 819-826, ISSN 0022-0434

Best, M. J. & Ritter, K. (1985). *Linear Programming: Active Set Analysis and Computer Programs*, Prentice-Hall

Bobrow, J. E.; Dubowsky, S. & Gibson, J. S. (1985). Time-optimal control of robotic manipulators along specified paths, *International J. Robotics Research*, Vol. 4, 3-17, ISSN 0278-3649

Bobrow, J. E. (1988). Optimal robot path planning using the minimum-time criterion, *IEEE J. Robotics and Automation*, Vol. 4, 443-450, ISSN 0882-4967

Bobrow, J. E.; Martin, B.; Sohl, G.; Wang, E. C.; Park, F. C. & Kim, J. (2001). Optimal robot motions for physical criteria, *J. Robotic Systems*, Vol. 18, 785-795, ISSN 0741-2223

Bryson, Jr. A. E. & Meier, E. B. (1990). Efficient algorithm for time-optimal control of a two-link manipulator, *J. Guidance, Control, and Dynamics*, Vol. 13, 859-866, ISSN 0731-5090

Cao, B.; Dodds, G. I. & Irwin, G.. W. (1998). A practical approach to near time-optimal inspection-task-sequence planning for two cooperative industrial robot arms, *International J. Robotics Research*, Vol. 17, 858-867, ISSN 0278-3649

Constantinescu, D. & Croft, E. A. (2000). Smooth and time-optimal trajectory planning for industrial manipulators along specified paths, *J. Robotic Systems*, Vol. 17, 233-249, ISSN 0741-2223

Craig, J. J. (1986). *Introduction to Robotics; Mechanics and Control*, Addison-Wesley, ISBN0201095289

Fenton, R. G.; Benhabib, B. & Goldenberg, A. A. (1986). Optimal point-to-point motion control of robots with redundant degrees of freedom, *Trans ASME J. Engineering for Industry*, Vol. 108, 120-126, ISSN 0022-0817

Fletcher, R. (1987). *Practical Method of Optimization*, 2nd ed., John Wiley & Sons, ISBN0471494631

Formalsky, A. M. (1996). The time-optimal control of the bending of a plane two-link mechanism, *J. Applied Mathematics and Mechanics*, Vol. 60, 243-251, ISSN 0021-8928

Galicki, M. (1998). The planning of robotic optimal motions in the presence of obstacles, *International J. Robotics Research*, Vol. 17, 248-259, ISSN 0278-3649

Galicki, M. & Ucinski, D. (2000). Time-optimal motions of robotic manipulators, *Robotica*, vol. 18, 659-667, ISSN 0263-5747

Hol, C. W. J.; Willigenburg, L. G.; Henten, E. J. & Straten, G. (2001). A new optimization algorithm for singular and non-singular digital time-optimal control of robots, *Proceedings of IEEE International Conference on Robotics and Automation*, 1136-1141, ISBN078036578X

Jouaneh, M. K.; Dornfeld, D. A. & Tomizuka, M. (1990). Trajectory planning for coordinated motion of a robot and a positioning table: part 2, *IEEE Trans. Robotics and Automation*, Vol. 6, 746-759, ISSN 1042-296X

Lee, J. (1995). Dynamic programming approach to near minimum-time trajectory planning for two robots, *IEEE trans. Robotics and Automation*, Vol. 11, 160-164, ISSN 1042-296X

Ong, C. J. & Gilbert, E. G. (1996). Growth distances: new measures for object separation and penetration, *IEEE trans. Robotics and Automation*, Vol. 12, 888-903, ISSN 1042-296X

Ozaki, H. & Lin, C.-j. (1996). Optimal B-spline joint trajectory generation for collision-free movements of a manipulator under dynamic constraints, *Proceedings of IEEE International Conference on Robotics and Automation*, 3592-3597, ISBN0780329880

Park, J.-k. & Bobrow, J. E. (2005). Reliable computation of minimum-time motions for manipulators moving in obstacle fields using a successive search for minimum overload trajectories, *J. Robotic Systems*, Vol. 22, 1-14, ISSN 0741-2223

Prenter, P. M. (1975). *Splines and Variational Methods*, John Wiley & Sons, ISBN0471504025

Schlemmer, M. & Gruebel, G. (1998). Real-time collision-free trajectory optimization of robot manipulators via semi-infinite parameter optimization, *International J. Robotics Research*, Vol. 17, 1013-1021, ISSN 0278-3649

Shiller, Z. & Dubowsky, S. (1991). On computing the global time-optimal motions of robotic manipulators in the presence of obstacles, *IEEE Trans. Robotics and Automation*, Vol. 7, 785-797, ISSN 1042-296X

Shiller, Z. (1994). On singular time-optimal control along specified paths, *IEEE Trans. Robotics and Automation*, Vol. 10, 561-566, ISSN 1042-296X

Shin, K. G. & McKay, N. D. (1985). Minimum-time control of robotic manipulators with geometric path constraints, *IEEE trans. Automatic Control*, AC-30, 531-541, ISSN 0018-9286

Shin, K. G. & Zheng, Q. (1992). Minimum-time collision-free trajectory planning for dual-robot systems, *IEEE trans. Robotics and Automation*, Vol. 8, 641-644, ISSN 1042-296X

Singh, S. K. & Leu, M. C. (1991). Manipulator motion planning in the presence of obstacles and dynamic constraints, *International J. Robotics Research*, Vol. 10, 171-186, ISSN 0278-3649

Wang, C.-Y. E.; Timoszyk, W. K. & Bobrow, J. E. (2001). Payload maximization for open chained manipulators: finding weightlifting motions for a Puma 762 robot, *IEEE trans. Robotics and Automation*, Vol. 17, 218-224, ISSN 1042-296X

Video Applications for Robot Arms

Vincenzo Niola, Cesare Rossi
Department of Ingegneria Meccanica per l'Energetica
University of Napoli - "Federico II"
Via Claudio 21, 80125 Napoli
ITALY

1. Introduction

The "Artificial Vision" permits industrial automation and system vision able to act in the production activities without humane presence. So we suppose that the acquisition and interpretation of the imagines for automation purposes is an interesting topic.
Industrial application are referred to technological fields (assembly or dismounting, cut or stock removal; electrochemical processes; abrasive trials; cold or warm moulding; design with CAD techniques; metrology), or about several processes (control of the row material; workmanship of the component; assemblage; packing or storages; controls of quality; maintenance).
The main advantages of this technique are:

1) elimination of the human errors, particularly in the case of repetitive or monotonous operations;
2) possibility to vary the production acting on the power of the automatic system (the automatic machines can operate to high rhythms day and night every day of the year);
3) greater informative control through the acquisition of historical data; these data can be used for successive elaborations, for the analysis of the failures and to have statistics in real time;
4) quality control founded on objective parameters in order to avoid dispute, and loss of image.

In addition, another interesting application can be considered: the robot cinematic calibration and the trajectories recording. First of all it is important to consider that, by a suitable cameras calibration technique, it is possible to record three dimensional objects and trajectories by means of a couple of television cameras. By means of perspective transformation it is possible to associate a point in the geometric space to a point in a plane. In homomogeneous coordinates the perspective transformation matrix has non-zero elements in the fourth row.

2. The perspective transform

In this paragraph, an expression of perspective transformation is proposed, in order to introduce the perspective concepts for the application in robotic field.
The proposed algorithm uses the fourth row of the Denavit and Hartemberg transformation matrix that, for kinematics' purposes, usually contains three zeros and a scale factor, so it is useful to start from the perspective transform matrix.

2.1 The matrix for the perspective transformation

It is useful to remember that by means of a perspective transform it is possible to associate a point in the geometric space to a point in a plane, that will be called "image plane"; this will be made by using a scale factor that depends on the distance between the point itself and the image plane.

Let's consider fig.1: the position of point P in the frame O,x,y,z is given by the vector **w**, while the same position in the frame Ω,ξ,η,ζ is given by vector $\mathbf{w_r}$ and the image plane is indicated with $\mathcal{R}$this last, for the sake of simplicity is supposed to be coincident with the plane ξ,η.

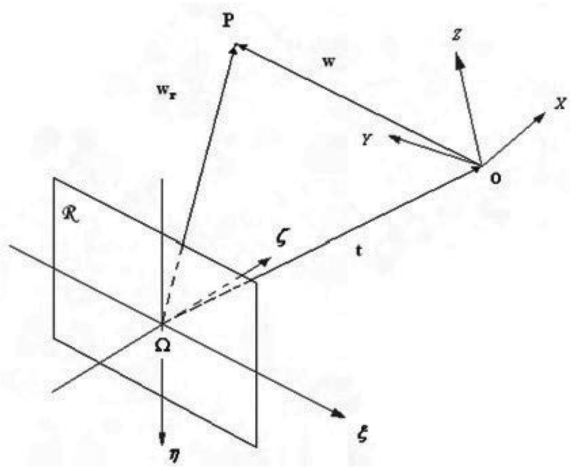


Fig. 1. Frames for the perspective transformation.

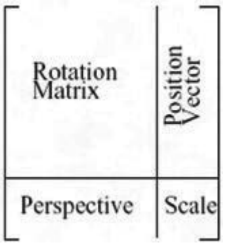
The vectors above are joined by the equation:

$$\begin{Bmatrix} w_{r,x} \\ w_{r,y} \\ w_{r,z} \\ sf \end{Bmatrix} = \begin{bmatrix} R_{11} & R_{12} & R_{13} & t_\xi \\ R_{21} & R_{22} & R_{23} & t_\eta \\ R_{31} & R_{32} & R_{33} & t_\zeta \\ 0 & 0 & 0 & sf \end{bmatrix} = \begin{Bmatrix} w_x \\ w_y \\ w_z \\ sf \end{Bmatrix} \tag{1}$$

where sf is the scale factor; more concisely equation (1) can be written as follows:

$$\tilde{w}_r = T \cdot \tilde{w} \tag{2}$$

where the tilde indicates that the vectors are expressed in homogeneous coordinates. The matrix T is a generic transformation matrix that is structured according to the following template:



The scale factor will almost always be 1 and the perspective part will be all zeros except when modelling cameras.

The fourth row of matrix [T] contains three zeros; as for these last by means of the prospectic transform three values, generally different by zero, will be determined.

Lets consider, now, fig. 2: the vector **w***, that represents the projection of vector **wr** on the plane ξ,η.

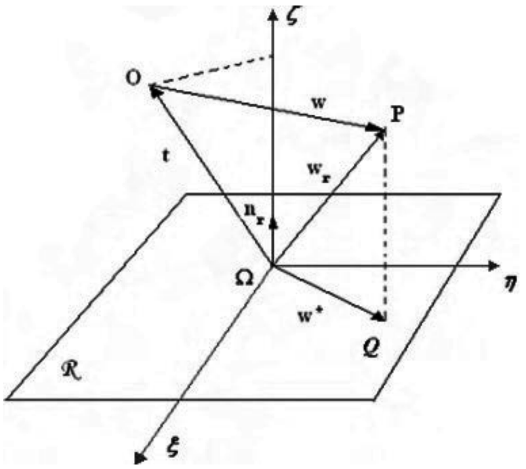


Fig. 2. Vectors for the perspective transformation.

The coordinates of point P in the image plane can be obtained from the vector $\mathbf{w_r}$, in fact, these coordinates are the coordinates of **w***, that can be obtained as follows:
Let's consider the matrix R:

$$R = \begin{bmatrix} \hat{\xi}^T \\ \hat{\eta}^T \\ \hat{\zeta}^T \end{bmatrix} \tag{3}$$

where $\hat{\xi}$ $\hat{\eta}$ $\hat{\zeta}$ are the versor of the frame {Ω,ξ,η,ζ} axes in the frame {O,x,y,z}.
In fig.2 the vector **t** indicates the origin of frame O,x,y,z in the frame Ω,ξ,η,ζ and the projection of P on the plane ξ,η is represented by point Q, which position vector is **w***. This last, in homogeneous coordinates is given by:

$$\tilde{w}^* = \begin{pmatrix} w_{r,\xi} \\ w_{r,\eta} \\ 0 \\ 1 \end{pmatrix} = \begin{pmatrix} \hat{\xi}^T w + t_\xi \\ \hat{\eta}^T w + t_\eta \\ 0 \\ 1 \end{pmatrix} \tag{4}$$

In the same figure, $\mathbf{n_r}$ is the versor normal to the image plane R , and **n** will be the same versor in the frame {O,X,Y,Z}. The perspective image of vector **w*** can be obtained by assessing a suitable scale factor. This last depends on the distance d between point P and the image plane. The distance d is given from the following scalar product:

$$d = n_r^{\ T} w_r \tag{5}$$

Let's indicate with **w**{Ω,ξ,η,ζ} the vector **w** in the frame {Ω,ξ,η,ζ}:

$$\tilde{w}_{\{\Omega,\xi,\eta,\zeta\}} = \begin{pmatrix} w_\xi \\ w_\eta \\ w_\zeta \\ 1 \end{pmatrix}$$

Because $\hat{\xi}$ $\hat{\eta}$ $\hat{\zeta}$ are the versor of the frame {Ω,ξ,η,ζ} axes in the frame {O,x,y,z}, it is possible to write the coordinates of the vector **w**{Ω,ξ,η,ζ} in the frame {Ω,ξ,η,ζ}:

$$w_\xi = \hat{\xi}^T \cdot w = \xi_x w_x + \xi_y w_y + \xi_z w_z;$$
$$w_\eta = \hat{\eta}^T \cdot w = \eta_x w_x + \eta_y w_y + \eta_z w_z;$$
$$w_\xi = \hat{\zeta}^T \cdot w = \zeta_x w_x + \zeta_y w_y + \zeta_z w_z.$$

In the frame {Ω,ξ,η,ζ}, it is possibile to write $\mathbf{w_r}$ as sum of **w**{Ω,ξ,η,ζ} and **t**:

$$\widetilde{w}_r = \widetilde{w}_{\{\Omega,\xi,\eta,\zeta\}} + \widetilde{t} = \begin{pmatrix} w_\xi + t_\xi \\ w_\eta + t_\eta \\ w_\zeta + t_\zeta \\ 1 \end{pmatrix} = \begin{pmatrix} \xi_x w_x + \xi_y w_y + \xi_z w_z + t_\xi \\ \eta_x w_x + \eta_y w_y + \eta_z w_z + t_\eta \\ \zeta_x w_x + \zeta_y w_y + \zeta_z w_z + t_\zeta \\ 1 \end{pmatrix} \tag{5'}$$

an expression of d is:

$$d = n_r^T w_r = \begin{pmatrix} n_{r,\xi} \\ n_{r,\eta} \\ n_{r,\zeta} \end{pmatrix}^T \cdot \begin{pmatrix} w_\xi + t_\xi \\ w_\eta + t_\eta \\ w_\zeta + t_\zeta \end{pmatrix} = \begin{pmatrix} n_{r,\xi} \\ n_{r,\eta} \\ n_{r,\zeta} \end{pmatrix}^T \cdot \begin{pmatrix} \xi_x w_x + \xi_y w_y + \xi_z w_z + t_\xi \\ \eta_x w_x + \eta_y w_y + \eta_z w_z + t_\eta \\ \zeta_x w_x + \zeta_y w_y + \zeta_z w_z + t_\zeta \end{pmatrix}$$

Let's introduce the expressions:

$$D_x = \frac{(\xi_x w_x + \xi_y w_y + \xi_z w_z + t_\xi) \cdot n_{r,\xi}}{w_x};$$
$$D_y = \frac{(\eta_x w_x + \eta_y w_y + \eta_z w_z + t_\eta) \cdot n_{r,\eta}}{w_y};$$
$$D_z = \frac{(\zeta_x w_x + \zeta_y w_y + \zeta_z w_z + t_\zeta) \cdot n_{r,\zeta}}{w_z};$$

it is possible to write:

$$d = n_r^T w_r = \begin{pmatrix} D_x \\ D_y \\ D_z \\ 0 \end{pmatrix}^T \cdot \begin{pmatrix} w_x \\ w_y \\ w_z \\ 1 \end{pmatrix} = D^T \cdot w \tag{5''}$$

In the expression (5″) the vector D is:

$$D=\begin{pmatrix} D_x \\ D_y \\ D_z \\ 0 \end{pmatrix}$$

As vector **w*** is given by:

$$\tilde{w}^*_p=\begin{pmatrix} \hat{\xi}^T w+t_\xi \\ \hat{\eta}^T w+t_\eta \\ 0 \\ n_r{}^T w_r \end{pmatrix} \tag{6}$$

The perspective matrix [Tp] can be obtained:

$$\tilde{w}^*_p=T_p\cdot\tilde{w} \qquad \Rightarrow \qquad T_p=\begin{bmatrix} \xi_x & \xi_y & \xi_z & t_\xi \\ \eta_x & \eta_y & \eta_z & t_\eta \\ 0 & 0 & 0 & 0 \\ D_x & D_y & D_z & 0 \end{bmatrix} \tag{7}$$

The terms Dx, Dy, Dz assume infinity values if the vector **w** has one of his coordinates null, but this does not influence on generality of the relation $\tilde{w}^*_p=T_p\cdot\tilde{w}$,in fact in$p$ this case, the term that assume infinity value, is multiplied for zero.

2.2 The perspective concept

From equation 7 some useful properties can be obtained in order to define how a geometric locus changes its representation when a perspective transform occurs. As for an example of the above said, let us consider the representation of the displacement of a point in the space: suppose that the displacement occurs, initially, in the positive direction of x axis. Say this displacement Δ w, the point moves from the position P to the position P′, that are given by the vectors:

$$w=\begin{pmatrix} w_x \\ w_y \\ w_z \end{pmatrix} \quad \text{e} \quad w'=\begin{pmatrix} w'_x \\ w_y \\ w_z \end{pmatrix} \tag{8}$$

If the perspective transforms are applied we have :

$$p=T_P\cdot w \qquad \text{and} \quad p'=T_P\cdot w'$$

so:

$$p=\begin{pmatrix} \dfrac{\xi_x w_x+\xi_y w_y+\xi_z w_z+t_\xi}{D^T w} \\ \dfrac{\eta_x w_x+\eta_y w_y+\eta_z w_z+t_\eta}{D^T w} \\ 0 \\ \end{pmatrix}$$

and

$$p' = \begin{pmatrix} \dfrac{\xi_x w'_x + \xi_y w_y + \xi_z w_z + t_\xi}{D'^T w'} \\ \dfrac{\eta_x w'_x + \eta_y w_y + \eta_z w_z + t_\eta}{D'^T w'} \\ 0 \end{pmatrix} \tag{9}$$

Hence, the displacement in the image plane is given by:

$$\Delta p \quad = p' - p$$

that is to say:

$$\Delta p = \begin{Bmatrix} \dfrac{\xi_x \cdot [w_x'(D^T w) - w_x(D'^T w')]}{(D^T w)(D'^T w')} \\ \dfrac{\eta_x \cdot [w_x'(D^T w) - w_x(D'^T w')]}{(D^T w)(D'^T w')} \\ 0 \end{Bmatrix} \tag{10}$$

In this way, a displacement Δw along the x axis corresponds to a displacement Δp in the image plane along a straight line which pitch is η_x/ξ_x. So the x axis equation in the image plane is :

$$\eta = (\eta_x/\xi_x) \cdot \xi + \frac{\xi_x t_\eta - \eta_x t_\xi}{\xi_x} \tag{11}$$

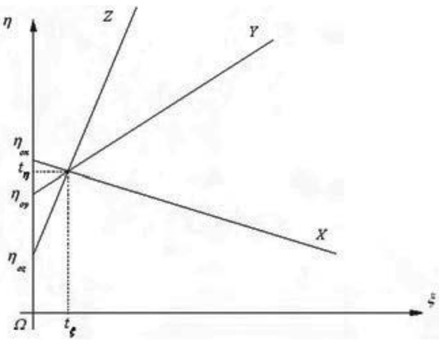


Fig. 3. Frames.

The interception was calculated by imposing that the point which cordinates are belongs to the x axis. In the same way it is possible to obtain the y axis and the z axis equations:

$$\text{y axis : } \eta = (\eta_y \big/ \xi_y) \cdot \xi + \frac{\xi_y t_\eta - \eta_y t_\xi}{\xi_y} \tag{12}$$

$$\text{z axis : } \eta = (\eta_z \big/ \xi_z) \cdot \xi + \frac{\xi_z t_\eta - \eta_z t_\xi}{\xi_z} \tag{13}$$

By means of equations (11), (12) and (13) it is possible to obtain a perspective representation of a frame belonging to the Cartesian space in the image plane; that is to say: for a given body it is possible to define it's orientation (e.g. roll, pitch and yaw) in the image plane.

An example could clarify what exposed above: let's consider a circumference which equation in the frame xyz is:

$$x^2 + y^2 = \rho^2 \tag{14}$$

It is possible to associate the geometric locus described from equation (14) to the corresponding one in the image plane; in fact by means of equations (11),(12) e (13) it is possible to write:

$$\begin{aligned} x &= [\xi, \frac{\eta_x}{\xi_x}\cdot\xi + \eta_{0x}] \\ y &= [\frac{\xi_y}{\eta_y}\cdot(\eta - \eta_{0y}), \eta] \end{aligned} \tag{15}$$

By substituting these last in the (14) we obtain :

$$\left[1+\left(\frac{\eta_x}{\xi_x}\right)^2\right]\cdot\xi^2 + \left[1+\left(\frac{\xi_y}{\eta_y}\right)^2\right]\cdot\eta^2 + \left(\frac{2\eta_x\eta_{0x}}{\xi_x}\right)\cdot$$
$$\cdot\xi + \left(\frac{2\eta_{0y}\xi_y{}^2}{\eta_y{}^2}\right)\cdot\eta = \rho^2 - \left(\frac{\eta_{0x}{}^2\eta_y{}^2 - \eta_{0y}{}^2\xi_y{}^2}{\eta_y{}^2}\right)$$

That represents the equation of a conic section. In particular a circumference the centre of which is in the origin of the xyz frame that becomes an ellipse having its foci on a generic straight line in the image plane.

2.3 Perspective transformation in D-H robotic matrix

For kinematics' purposes in robotic applications, it is possible to use the Denavit and Hartemberg transformation matrix in homogeneous coordinates in order to characterize the end-effector position in the robot base frame by means of joints variable, this matrix usually contains three zeros and a scale factor in the fourth row. The general expression of the homogenous transformation matrix that allows to transform the coordinates from the frame i to frame i-1, is:

$$A_1^{i-1} = \begin{bmatrix} C\vartheta_i & -C\alpha_i\cdot S\vartheta_i & S\alpha_i\cdot S\vartheta_i & a_i\cdot C\vartheta_i \\ S\vartheta_i & C\alpha_i\cdot C\vartheta_i & -S\alpha_i\cdot C\vartheta_i & a_i\cdot S\vartheta_i \\ 0 & S\alpha_i & C\alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

For a generic robot with *n* d.o.f., the transformation matrix from end-effector frame to base frame, has the following expression:

$$T_n^0 = A_1^0 \cdot A_2^1 \cdot A_3^2 \cdot \ldots\ldots \cdot A_n^{n-1}$$

With this matrix it is possible to solve the expression:

$$\{P\}_0 = T_n^0 \cdot \{P\}_n$$

where $\{P\}_0$ and $\{P\}_n$ are the vectors that represent a generic point P in frame 0 and *P n* frame n.
Can be useful to include the perspective concepts in this trasformation matrix, in this way it is possible to obtain a perspective representation of the robot base frame, belonging to the Cartesian space, in an image plane, like following expression shows:

$$\{P\}_p = T_p\cdot\{P\}_0 = T_p\cdot T_n^0\cdot\{P\}_n = [T_p]_n^0\cdot\{P\}_n \tag{16}$$

where $\{P\}_p$ is the perspective image of generic point P and $[T_p]_n^0$ is the perspective transformation matrix from end-effector frame to an image plane.
With this representation the fourth row of the Denavit and Hartemberg matrix will contain non-zero elements. A vision system demands an application like this.

3. The camera model

When vision systems are used for robotic applications, it is important to have a suitable model of the cameras.
A vision system essentially associates a point in the Cartesian space with a point on the image plane. A very common vision system is the television camera that is essentially composed by an optic system (one or more lenses), an image processing and managing system and an image plane; this last is composed by vision sensors. The light from a point in the space is conveyed by the lenses on the image plane and recorded by the vision sensor.
Let us confine ourselves to consider a simple vision system made up by a thin lens and an image plane composed by CCD (*Charged Coupled Device*) sensors. This kind of sensor is a device that is able to record the electric charge that is generated by a photoelectric effect when a photon impacts on the sensor's surface.
It is useful to remember some aspects of the optics in a vision system.

3.1 The thin lenses

A lens is made up by two parts of a spherical surfaces (dioptric surfaces) joined on a same plane. The axis, normal to this plane, is the optical axis. As shown in fig.4, a convergent lens conveys the parallel light rays in a focus F at distance f (focal distance) from the lens plane.

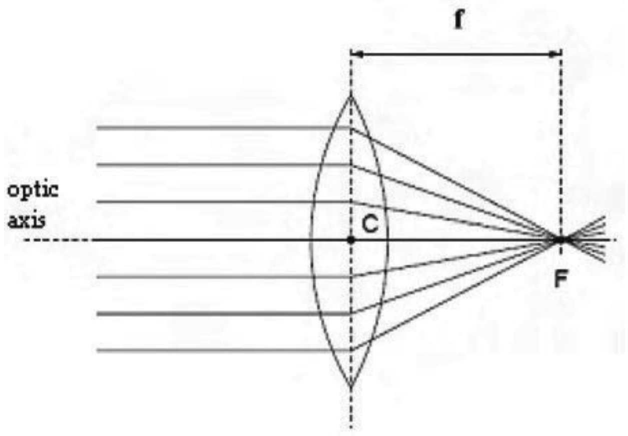


Fig. 4. Convergent lens.

The focal distance f, in air, is given by:

$$f = (n-1)\cdot\left(\frac{1}{R_1}-\frac{1}{R_2}\right)$$

where n is the refractive index of the lens and *R1* ed *R2* are the bending radius of the dioptric surfaces.
Now consider a thin lens, a point P and a plane on which the light-rays refracted from the lens are projected as shown in fig. 5. the equation for the thin lenses gives:

$$\frac{1}{d}+\frac{1}{L}=\frac{1}{f}$$

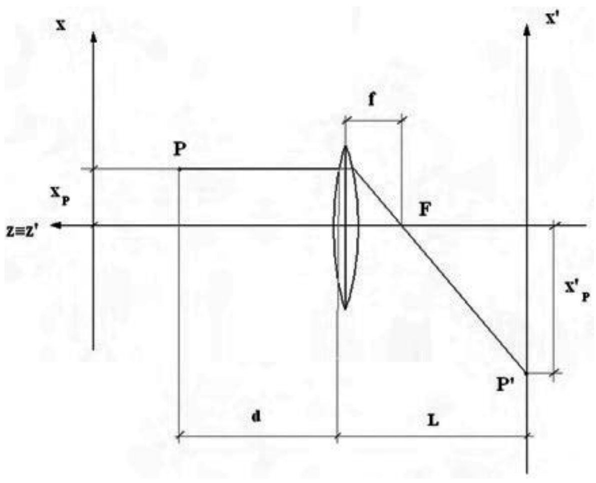


Fig. 5. Thin lens.

It is possible to determinate the connection between the position of point P in the space and it's correspondent P' in the projection's plane (fig. 5).
If two frames (xyx for the Cartesian space and x'y'z' for the image plane), having their axes parallel, are assigned and if the thickness of the lens is neglected, from the similitude of the triangles in fig.5 it comes:

$$\frac{x_P}{f} = -\frac{x_P}{L-f}$$

From the equation of the thin lenses:

$$L = \frac{f \cdot d}{d-f} \Rightarrow L - f = \frac{f^2}{d-f}$$

Hence:

$$x_P^{'} = -\frac{f}{d-f} \cdot x_P$$

If we consider that generally the distance of a point from the camera's objective is one meter or more while the focal distance is about some millimetres (*d*>>*f*), the following approximation can be accepted:

$$x_P^{'} \cong -\frac{f}{d} \cdot x_P$$

So the coordinates of the point in the image plane can be obtained by scaling the coordinates in the Cartesian space by a factor −*d/f*. The minus sign is due to the upsetting of the image.

3.2 The model of the telecamera

As already observed a telecamera can be modelled as a thin lens and an image plane with CCD sensors. The objects located in the Cartesian space emit rays of light that are refracted from the lens on the image plane. Each CCD sensor emit an electric signal that is proportional to the intensity of the ray of light on it; the image is made up by a number of pixels, each one of them records the information coming from the sensor that corresponds to that pixel.
In order to indicate the position of a point of an image it is possible to define a frame u,v (fig. 6) which axes are contained in the image plane. To a given point in the space (which

position is given by its Cartesian coordinates) it is possible to associate a point in the image plane (two coordinates) by means of the telecamera. So, the expression "model of the camera" means the transform that associates a point in the Cartesian space to a point in the image space.
It has to be said that in the Cartesian space a point position is given by three coordinates expressed in length unit while in the image plane the two coordinates are expressed in pixel; this last is the smaller length unit that ca be revealed by the camera and isn't a normalized length unit. The model of the camera must take onto account this aspect also.
In order to obtain the model of the camera the scheme reported in fig.6 can be considered.
Consider a frame xyz in the Cartesian space, the position of a generic point P in the space is given by the vector w. Then consider a frame ξηζ having the origin in the lens centre and the plane ζη coincident with the plane of the lens; hence, the plane ζη is parallel to the image plane and ζ axis is coincident with the optical axis. Finally consider a frame u,v on the image plane so that uo and vo are the coordinates of the origin of frame ξηζ, expressed in pixel.

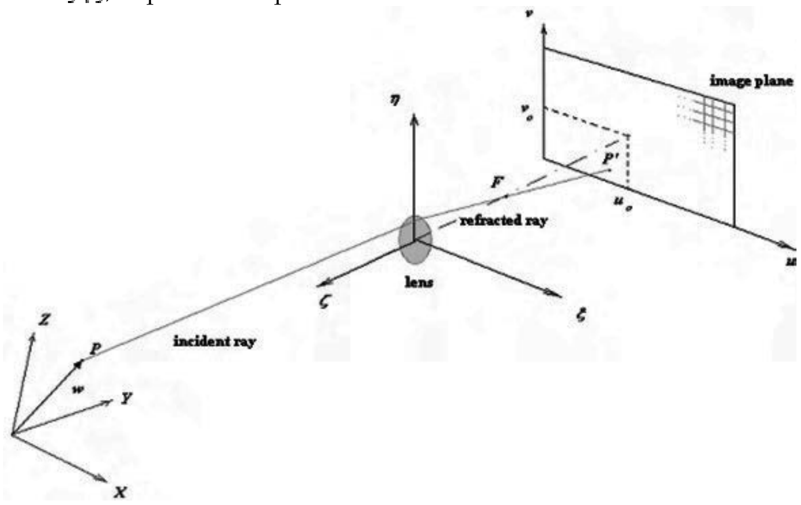


Fig.6. Camera model.

As it was already told, the lens makes a perspective transform in which the constant of proportionality is –f . If this transform is applied to vector w, a w_l vector is obtained:

$$\tilde{w}_l = T_l \cdot \tilde{w} \tag{17}$$

Were the matrix T_l is obtained dividing by –f the last row of the perspective transformation matrix Tp, (7).

$$T_l = \begin{bmatrix} \xi_x & \xi_y & \xi_z & t_\xi \\ \eta_x & \eta_y & \eta_z & t_\eta \\ 0 & 0 & 0 & 0 \\ -\frac{D_x}{f} & -\frac{D_y}{f} & -\frac{D_z}{f} & 0 \end{bmatrix} \tag{18}$$

Substantially, the above essentially consists in a changing of the reference frames and a scaling based on the rules of geometric optics previously reported.
Assumed x_l e y_l as the first two components of the vector wl, the coordinates u and v (expressed in pixel) of P' (image of P) are :

$$\begin{cases} u = \dfrac{x_1}{\delta_u} + u_o \\ v = \dfrac{x_1}{\delta_v} + v_o \end{cases} \tag{19}$$

Where δ_u e δ_v are respectively the horizontal and vertical dimensions of the pixel. So, by substituting equation (17) in equation (19) it comes:

$$\begin{cases} u = -\dfrac{f}{D^T w}\left[\left(\dfrac{1}{\delta_u}\cdot\hat{\xi} - \dfrac{u_o}{f}\cdot D\right)^T w + \dfrac{1}{\delta_u}\cdot t_\xi\right] \\ v = -\dfrac{f}{D^T w}\left[\left(\dfrac{1}{\delta_v}\cdot\hat{\eta} - \dfrac{v_o}{f}\cdot D\right)^T w + \dfrac{1}{\delta_v}\cdot t_\eta\right] \end{cases} \tag{20}$$

Finally if we define the vector $m = [u \quad v]^T$, the representation in homogeneous *w T w T*~coordinates $\tilde{m} = [m_1 \quad m_2 \quad -D^T w/f]^T = [u \quad v \quad -D^T w/f]^T$ of the previous vector can be written :

$$\tilde{m} = M \cdot \tilde{w} \tag{21}$$

Where M is the matrix :

$$M = \begin{bmatrix} \left(\dfrac{\xi_x}{\delta_u} - \dfrac{u_o D_x}{f}\right) & \left(\dfrac{\xi_y}{\delta_u} - \dfrac{u_o D_y}{f}\right) & \left(\dfrac{\xi_z}{\delta_u} - \dfrac{u_o D_z}{f}\right) & t_\xi/\delta_u \\ \left(\dfrac{\eta_x}{\delta_v} - \dfrac{v_o D_x}{f}\right) & \left(\dfrac{\eta_y}{\delta_v} - \dfrac{v_o D_y}{f}\right) & \left(\dfrac{\eta_z}{\delta_v} - \dfrac{v_o D_z}{f}\right) & t_\eta/\delta_v \\ -D_x/f & -D_y/f & -D_z/f & 0 \end{bmatrix} \tag{22}$$

that represents the requested model of the camera.

3.3 The stereoscopic vision

That above reported concurs to determine the coordinates in image plane (u,v) of a generic point of tridimensional space ($w=(w_x \; w_y \; w_z \; 1)^T$, but the situation is more complex if it is necessary to recognise the position (w) of a point starting to its camera image (u, v). In this case the expression (20) becomes a system of 2 equation with 3 unknowns, so it isn't absolutely solvable.

This obstacle can be exceeded by means of a vision system with at least two cameras. In this way, that above reported can be applied to the recording of a robot trajectory in the three dimensional space by using two cameras. This will emulate the human vision.

Let us consider two cameras and say M and M' their transform matrixes. We want to recognise the position of a point P, that in the Cartesian space is given by a vector w in a generic frame xyz. From equation (21) we have:

$$\begin{cases} \tilde{m} = M \cdot w \\ \tilde{m}' = M' \cdot w \end{cases} \tag{23}$$

Where

$$\tilde{m} = \begin{pmatrix} m_1 \\ m_2 \\ -D^T w/f \end{pmatrix}$$

and

$$\tilde{m}' = \begin{pmatrix} m_1' \\ m_2' \\ -D'^T w/f' \end{pmatrix}$$

are the position vectors in the image plane of the cameras, in homogeneous coordinates, and:

$$M=\begin{bmatrix} \mu_1^T & \mu_{14} \\ \mu_2^T & \mu_{24} \\ -D^T/f & 0 \end{bmatrix};$$

$$M'=\begin{bmatrix} \mu'^T_1 & \mu'_{14} \\ \mu'^T_2 & \mu'_{24} \\ -D'^T/f' & 0 \end{bmatrix} \tag{24}$$

are the [3x4] transformation matrixes (22) from spatial frame to image planes of two cameras.
The first equation of the system (23), in Cartesian coordinates (non-homogenous), can be written as:

$$\begin{cases} u=-f\cdot\dfrac{\mu_1^T w+\mu_{14}}{n^T w} \\ v=-f\cdot\dfrac{\mu_2^T w+\mu_{24}}{n^T w} \end{cases} \tag{25}$$

or:

$$\begin{cases} (u\cdot D+f\cdot\mu_1)^T w=\mu_{14} \\ (v\cdot D+f\cdot\mu_2)^T w=\mu_{24} \end{cases} \tag{26}$$

In the same way for the camera whose transform matrix is M′, it can be written:

$$\begin{cases} (u'\cdot D'+f'\cdot\mu'_1)^T w=\mu'_{14} \\ (v'\cdot D'+f'\cdot\mu'_2)^T w=\mu'_{24} \end{cases} \tag{27}$$

By arranging eq.(26) and eq.(27) we obtain:

$$\begin{bmatrix} (u\cdot D+f\cdot\mu_1)^T \\ (v\cdot D+f\cdot\mu_2)^T \\ (u'\cdot D'+f'\cdot\mu'_1)^T \\ (v'\cdot D'+f'\cdot\mu'_2)^T \end{bmatrix}\cdot w=\begin{bmatrix} \mu_{14} \\ \mu_{24} \\ \mu'_{14} \\ \mu'_{24} \end{bmatrix} \tag{28}$$

This last equation represents the stereoscopic problem and consist in a system of 4 equation in 3 unknown (*wx,wy,wz*). As the equations are more than the unknowns can be solved by a least square algorithm. In this way it is possible to invert the problem that is described by equations (20) and to recognise the position of a generic point starting to its camera image.

3.4 The stereoscopic problem

Relation (28) represents the stereoscopic problem, it consists in a system of 4 equations in 3 unknown, in the form:

$$A(u,u,u',v',w)\cdot w=B$$

where A is a matrix that depends by two couple of camera coordinates (u,v) and (u′,v′), and by vector w, and B is a vector with parameters of cameras configuration.
It is possible to find an explicit form of this problem.

Vectors D and w have the following expression:

$$w = \begin{pmatrix} w_x \\ w_y \\ w_z \end{pmatrix};$$

$$D = \begin{pmatrix} D_x \\ D_y \\ D_z \end{pmatrix} = \begin{pmatrix} \dfrac{\left(\xi_x w_x + \xi_y w_y + \xi_z w_z + t_\xi\right) \cdot n_{r,\xi}}{w_x} \\ \dfrac{\left(\eta_x w_x + \eta_y w_y + \eta_z w_z + t_\eta\right) \cdot n_{r,\eta}}{w_y} \\ \dfrac{\left(\zeta_x w_x + \zeta_y w_y + \zeta_z w_z + t_\zeta\right) \cdot n_{r,\zeta}}{w_z} \end{pmatrix} \tag{29}$$

Starting to first equation of (20), it is possible to write:

$$u = -\frac{f}{D^T w}\left[\left(\frac{1}{\delta_u} \cdot \hat{\xi} - \frac{u_o}{f} \cdot D\right)^T w + \frac{1}{\delta_u} \cdot t_\xi\right] \Rightarrow$$

$$\left(\frac{\xi_x}{\delta_u} - \frac{u_0 \cdot D_x}{f}\right) \cdot w_x + \left(\frac{\eta_x}{\delta_u} - \frac{u_0 \cdot D_y}{f}\right) \cdot w_y + \left(\frac{\zeta_x}{\delta_u} - \frac{u_0 \cdot D_z}{f}\right) \cdot w_z + \frac{t_\xi}{\delta_u} =$$

$$-\frac{u}{f} \cdot \left[D_x \cdot w_x + D_y \cdot w_y + D_z \cdot w_z\right] \Rightarrow$$

$$\frac{1}{\delta_u}\left(\xi_x \cdot w_x + \eta_x \cdot w_y + \zeta_x \cdot w_z\right) -$$

$$\frac{u_0}{f}\left(D_x \cdot w_x + D_x \cdot w_y + D_x \cdot w_z\right) + \frac{u}{f}\left(D_x \cdot w_x + D_x \cdot w_y + D_x \cdot w_z\right) = -\frac{t_\xi}{\delta_u} \tag{30}$$

By means of equation (29), it is possible to write:

$$D_x \cdot w_x + D_x \cdot w_y + D_x \cdot w_z =$$

$$w_x\left(\xi_x \cdot n_{r,\xi} + \xi_y \cdot n_{r,\eta} + \xi_z \cdot n_{r,\zeta}\right) + w_y\left(\eta_x \cdot n_{r,\xi} + \eta_y \cdot n_{r,\eta} + \eta_z \cdot n_{r,\zeta}\right) +$$

$$w_z\left(\zeta_x \cdot n_{r,\xi} + \zeta_y \cdot n_{r,\eta} + \zeta_z \cdot n_{r,\zeta}\right) +$$

$$\left(t_\xi \cdot n_{r,\xi} + t_\eta \cdot n_{r,\eta} + t_\zeta \cdot n_{r,\zeta}\right)$$

If we define the elements:

$$N_\xi = \left(\xi_x \cdot n_{r,\xi} + \xi_y \cdot n_{r,\eta} + \xi_z \cdot n_{r,\zeta}\right);$$

$$N_\eta = \left(\eta_x \cdot n_{r,\xi} + \eta_y \cdot n_{r,\eta} + \eta_z \cdot n_{r,\zeta}\right);$$

$$N_\zeta = \left(\zeta_x \cdot n_{r,\xi} + \zeta_y \cdot n_{r,\eta} + \zeta_z \cdot n_{r,\zeta}\right);$$

$$k = \left(t_\xi \cdot n_{r,\xi} + t_\eta \cdot n_{r,\eta} + t_\zeta \cdot n_{r,\zeta}\right),$$

equation (30) becomes:

$$\frac{1}{\delta_u}\left(\xi_x \cdot w_x + \eta_x \cdot w_y + \zeta_x \cdot w_z\right) + \frac{u - u_0}{f} \cdot \left(N_\xi \cdot w_x + N_\eta \cdot w_y + N_\zeta \cdot w_z + k\right) = -\frac{t_\xi}{\delta_u} \Rightarrow$$

$$\left(\frac{\xi_x}{\delta_u} - \frac{(u - u_0) \cdot N_\xi}{f}\right) \cdot w_x + \left(\frac{\eta_x}{\delta_u} - \frac{(u - u_0) \cdot N_\eta}{f}\right) \cdot w_y + \left(\frac{\zeta_x}{\delta_u} - \frac{(u - u_0) \cdot N_\xi}{f}\right) \cdot w_z + \frac{u - u_0}{f} \cdot k =$$

$$-\frac{t_\xi}{\delta_u} \tag{31}$$

An analogous relation can be written for second equation of (20):

$$\left(\frac{\xi_y}{\delta_v}-\frac{(v-v_0)\cdot N_\xi}{f}\right)\cdot w_x+\left(\frac{\eta_y}{\delta_v}-\frac{(v-v_0)\cdot N_\eta}{f}\right)\cdot w_y+\left(\frac{\zeta_y}{\delta_v}-\frac{(v-v_0)\cdot N_\xi}{f}\right)\cdot w_z+\frac{v-v_0}{f}\cdot k=-\frac{t_\eta}{\delta_v} \tag{32}$$

By arranging equation (31) and (32), it is possible to redefine the stereoscopic problem, expressed by equation (28):

$$P(u,u,u',v')\cdot w=S \tag{33}$$

In equation (33) P is a matrix 4x3, whose elements depend only by (u,v) and (u′,v′), and B is a vector 4x1, whose elements contain parameters of cameras configuration.

The expression of matrix P is:

$$P=\begin{bmatrix}\frac{\xi_x}{\delta_u}-\frac{(u-u_0)\cdot N_\xi}{f} & \frac{\eta_x}{\delta_u}-\frac{(u-u_0)\cdot N_\eta}{f} & \frac{\zeta_x}{\delta_u}-\frac{(u-u_0)\cdot N_\xi}{f}\\ \frac{\xi_y}{\delta_v}-\frac{(v-v_0)\cdot N_\xi}{f} & \frac{\eta_y}{\delta_v}-\frac{(v-v_0)\cdot N_\eta}{f} & \frac{\zeta_y}{\delta_v}-\frac{(v-v_0)\cdot N_\xi}{f}\\ \frac{\xi'_x}{\delta_{u'}}-\frac{(u'-u'_0)\cdot N_{\xi'}}{f'} & \frac{\eta'_x}{\delta_{u'}}-\frac{(u'-u'_0)\cdot N_{\eta'}}{f'} & \frac{\zeta'_x}{\delta_{u'}}-\frac{(u'-u'_0)\cdot N_{\xi'}}{f'}\\ \frac{\xi'_y}{\delta_{v'}}-\frac{(v'-v'_0)\cdot N_{\xi'}}{f'} & \frac{\eta'_y}{\delta_{v'}}-\frac{(v'-v'_0)\cdot N_{\eta'}}{f'} & \frac{\zeta'_y}{\delta_{v'}}-\frac{(v'-v'_0)\cdot N_{\xi'}}{f'}\end{bmatrix}$$

The expression of vector S is:

$$S=\begin{Bmatrix}-\frac{t_\xi}{\delta_u}-\frac{u-u_0}{f}\cdot k\\ -\frac{t_\eta}{\delta_v}-\frac{v-v_0}{f}\cdot k\\ -\frac{t_{\xi'}}{\delta_{u'}}-\frac{u'-u'_0}{f'}\cdot k'\\ -\frac{t_{\eta'}}{\delta_{v'}}-\frac{v'-v'_0}{f'}\cdot k'\end{Bmatrix}$$

By equation (33) it is possibile to invert the problem that is described by eqs.(20) and to recognise the position of a generic point starting to its camera image, by means of pseudoinverse matrix P+ of matrix P.

$$P\cdot w=S\Rightarrow P^T\cdot P\cdot w=P^T\cdot S\Rightarrow w=\left(P^T\cdot P\right)^{-1}\cdot P^T\cdot S\Rightarrow w=P^+\cdot S \tag{34}$$

By means of equation (34), it is possible to solve the stereoscopic problem in all configurations in which is verified the condition:

4. The camera calibration

In order to determine the coordinate transformation between the camera reference system and robot reference system, it is necessary to know the parameters that regulate such transformation. The direct measure of these parameters is a difficult operation; it is better to identify them through a procedure that utilize the camera itself.

Camera calibration in the context of three-dimensional machine vision is the process of determining the internal camera geometric and optical characteristics (intrinsic parameters) and/or the 3-D position and orientation of the camera frame relative to a certain world coordinate system (extrinsic parameters). In many cases, the overall performance of the machine vision system strongly depends on the accuracy of the camera calibration.

In order to calibrate the tele-cameras a toolbox, developed by Christopher Mei, INRIA Sophia-Antipolis, was used. By means of this toolbox it is possible to find the intrinsic and extrinsic parameters of two cameras that are necessary to solve the stereoscopic problem. In order to carry out the calibration of a camera, it is necessary to acquire any number of images of observed space in which a checkerboard pattern is placed with different positions and orientations, fig 7.
In each acquired image, after clicking on the four extreme corners of a checkerboard pattern rectangular area, a corner extraction engine includes an automatic mechanism for counting the number of squares in the grid. This points are used like calibration points, fig. 8.

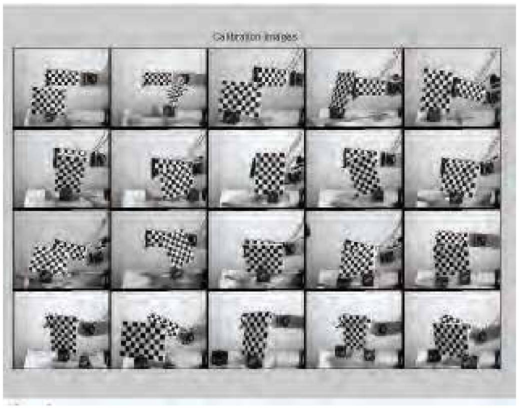


Fig. 7. Calibration images.

The dimensions **dX**, **dY** of each of squares are always kept to their original values in millimeters, and represent the parameters that put in relation the pixel dimensions with observed space dimensions (mm).

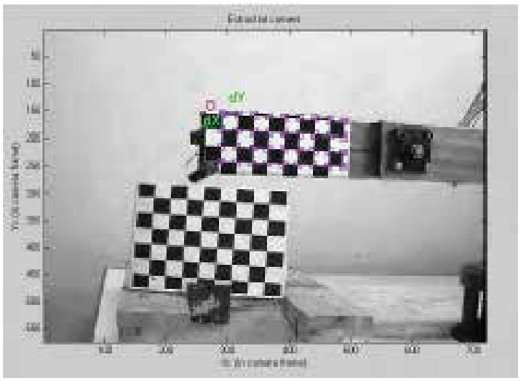


Fig. 8. Calibration image After corner extraction, calibration is done in two steps: first initialization, and then nonlinear optimization.

The initialization step computes a closed-form solution for the calibration parameters based not including any lens distortion.
The non-linear optimization step minimizes the total reprojection error (in the least squares sense) over all the calibration parameters (9 DOF for intrinsic: focal (2), principal point (2), distortion coefficients (5), and 6*n DOF extrinsic, with n = images number).
The calibration procedure allows to find the 3-D position of the grids with respect to the camera, like shown in fig. 9.

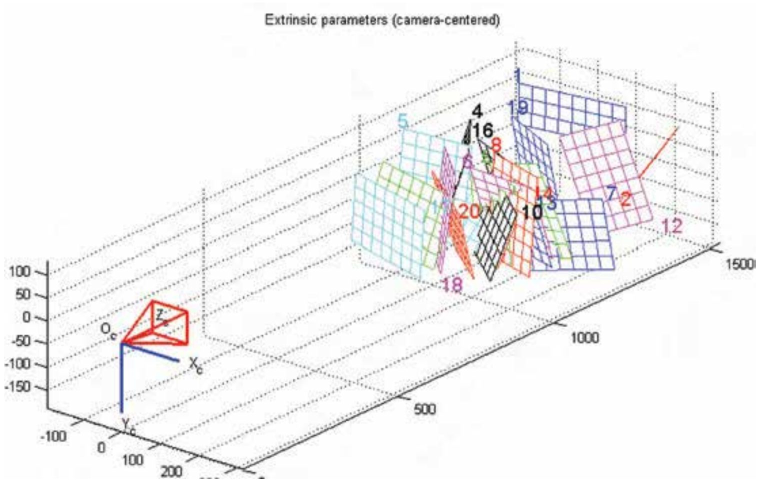


Fig. 9. Position of the grids for the calibration procedure.

With two camera calibration, it is possible to carry out a stereo optimization, by means of a toolbox option, that allows to do a stereo calibration for stereoscopic problem.
The global stereo optimization is performed over a minimal set of unknown parameters, in particular, only one pose unknown (6 DOF) is considered for the location of the calibration grid for each stereo pair. This insures global rigidity of the structure going from left view to right view. In this way the uncertainties on the intrinsic parameters (especially that of the focal values) for both cameras it becomes smaller.
After this operation, the spatial configuration of the two cameras and the calibration planes may be displayed in a form of a 3D plot, like shown in fig. 10.

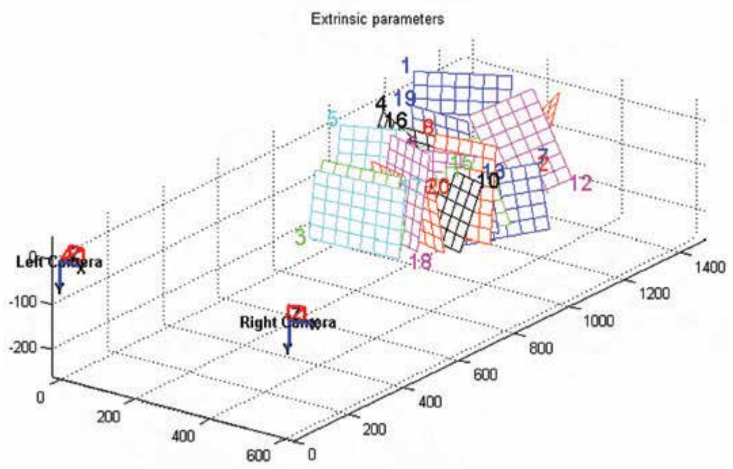


Fig.10. Calibration planes.

5. Robot cinematic calibration

Among the characteristics that define the performances of a robot the most important can be considered the repeatability and the accuracy. Generally, both these characteristics depend on factors like backlashes, load variability, positioning and zero putting errors, limits of the

transducers, dimensional errors, and so on. The last sources of error essentially depend on the correct evaluation of the Denavit and Hartemberg parameters. Hence, some of the sources of error can be limited by means of the cinematic calibration.
Basically, by the cinematic calibration it is assumed that if the error in the positioning of the robot's end effector is evaluated in some points of the working space, by means of these errors evaluation it is possible to predict the error in any other position thus offset it.
In few words, the main aim of the technique showed in this paper is to obtain precise evaluations of those Denavit-Hartenberg parameters that represent, for each of the links, the length, the torsion and the offset.

5.1 The calibration technique

This calibration technique essentially consists in the following steps:

I. The end-effector is located in an even position in the work space;
II. A vision system acquires and records the robot's image and gives the coordinates of an assigned point of the end-effector, expressed in pixels in the image plane.
III. By means of a suitable camera model, it is possible to find a relation between these coordinates expressed in pixels, and the coordinates of the assigned point of the end-effector in the world (Cartesian) frame.
IV. By means of the servomotor position transducers, the values of the joint position parameters are recorded for that end-effector position in the work space. In this way, for each of the camera images, the following arrays are obtained:

$$\begin{pmatrix} X_i \\ Y_i \\ Z_i \end{pmatrix}, \begin{pmatrix} \theta_{1,i} \\ \theta_{2,i} \\ \theta_{3,i} \end{pmatrix} \qquad (35)$$

where: $i = 1,\ldots,N$, and N is the number of acquired camera images (frames).
If the coordinates in the working space and the joint parameters are known, it's possible to write the direct kinematics equations in which the unknown are those Denavit-Hartenberg parameters that differ from the joint parameters; thus these Denavit-Hartenberg parameters represent the unknown of the kinematic calibration problem.
The expression of these equations is obtained starting from the transform matrix (homogeneous coordinates) that allows to transform the coordinates in the frame i to the coordinates in the frame i-1:

$${}^{i-1}A_i = \begin{bmatrix} C\vartheta_i & -C\alpha_i \cdot S\vartheta_i & S\alpha_i \cdot S\vartheta_i & a_i \cdot C\vartheta_i \\ S\vartheta_i & C\alpha_i \cdot C\vartheta_i & -S\alpha_i \cdot C\vartheta_i & a_i \cdot S\vartheta_i \\ 0 & S\alpha_i & C\alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad (36)$$

By means of such matrixes it is possible to obtain the transform matrix that allows to obtain the coordinates in the frame 0 (the fixed one) from those in frame n (the one of the last link) :

$${}^0T_n = {}^0A_1 \cdot {}^1A_2 \cdot \ldots \cdot {}^{n-1}A_n .$$

As for an example, if we consider a generic 3 axes revolute (anthropomorphic) robot arm, we'll obtain an equation that contains 9 constant kinematic parameters and 3 variable parameters (θ_1, θ_2, θ_3).
So, the vector:

$$\pi_{DH} = \begin{pmatrix} a_1 \\ a_2 \\ a_3 \\ d_1 \\ d_2 \\ d_3 \\ \alpha_1 \\ \alpha_2 \\ \alpha_3 \end{pmatrix} \tag{37}$$

represents the unknown of the kinematic calibration problem.
Said:

$$\Theta = \begin{pmatrix} \theta_1 \\ \theta_2 \\ \theta_3 \end{pmatrix} \tag{38}$$

the direct kinematics equation for this manipulator can be written as :

$$w = t_4(\pi_{DH}, \Theta) \tag{39}$$

where w is the position vector in the first frame and is the fourth row of the Denavit-Hartenberg transform matrix. In equation (39) it clearly appears that the position depends on the joint parameters and on the others Denavit-Hartenberg parameters. Equation (39) can be also seen as a system of 3 equations (in Cartesian coordinates) with 9 unknowns: the elements of vectorπ π_{DH} .
Obviously, it's impossible to solve this system of equations, but it's possible to use more camera images taken for different end-effector positions:

$$\begin{cases} t_4(\pi_{DH}, \Theta^1) = w_1 \\ t_4(\pi_{DH}, \Theta^2) = w_2 \\ \dots\dots\dots\dots\dots\dots \\ t_4(\pi_{DH}, \Theta^N) = w_N \end{cases} \tag{40}$$

with $N \geq 9$.
As, for each of the camera images the unknown Denavit-Hartemberg parameters are the same, equations (40) represent a system of N non linear equations in 9 unknowns. This system can be numerically solved by means of a minimum square technique.
It's known at a minimum square problem can be formulated as follows:
given the equation (39), find the solutions that minimize the expression:

$$\int_{D_\Theta} \left| t_4(\pi_{DH}, \Theta) - w \right|^2 \cdot d\Theta \tag{41}$$

This method can be simplified by substituting the integrals with summations, thus it must be computed the vector that minimize the expression:

$$\sum_{i=1}^{N} \left| t_4(\pi_{DH}, \Theta^i) - w_i \right|^2 \tag{42}$$

If we formulate the problem in this way, the higher is the number of images that have been taken (hence the more are the known parameters), the more accurate will be the solution, so it's necessary to take a number of pictures.

6. Trajectories recording

The trajectory recording, that is essential to study robot arm dynamical behaviour has been obtained by means of two digital television camera linked to a PC.
The rig, that has been developed, is based on a couple of telecameras; it allows us to obtain the velocity vector of each point of the manipulator. By means of this rig it is possible:

- to control the motion giving the instantaneous joint positions and velocities;
- to measure the motions between link and servomotor in presence of non-rigid transmissions;
- to identify the robot arm dynamical parameters.

An example of these video application for robot arm is the video acquisition of a robot arm trajectories in the work space by means of the techniques above reported.
In the figures 11 and 12 are reported a couple of frames, respectively, from the right telecamera and the left one. In fig. 13 is reported the 3-D trajectory, obtained from the frames before mentioned; in this last figure, for comparison, the trajectory obtained from the encoders signals is also reported.

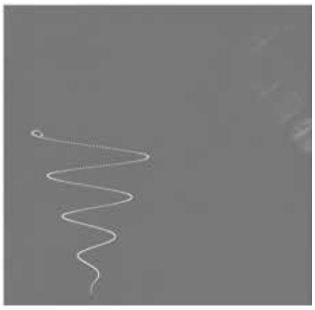


Fig. 11. Trajectories in image space: camera position 1.

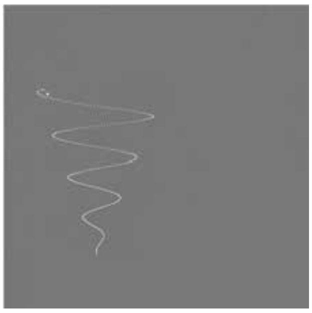


Fig. 12. Trajectories in image space: camera position 2.

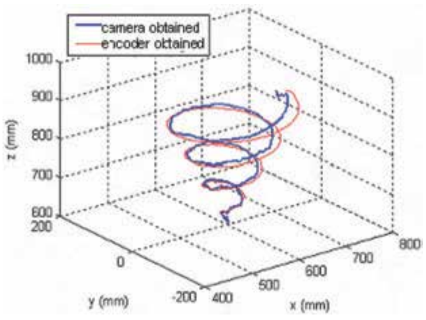


Fig. 13. comparison between trajectory recordings.

7. References

R. Brancati, C. Rossi, S. Scocca: Trajectories Recording for Investigations on Robot Dynamics and Mechanical Calibrations – *Proc. of 10th int. Workshop RAAD 2001*, Wien, May 16-19, 2001

V. Niola, C.Rossi: A method for the trajectories recording of a robot arm: early results -*Proc. 9th WSEAS International Conference on Computers*, Athens 11 – 16 July 2005.

V.Niola, C. Rossi: Video acquisition of a robot arm trajectories in the work space - *WSEAS Transactions on Computers*, Iusse 7, Vol. 4, july 2005, pagg. 830 – 836.

G.Nasti, V.Niola, S.Savino: Experimental Study on Vision Motion Measures in Robotics -*9th WSEAS International Conference on Computers*, Vouliagmeni – Athens, Greece, July 11 – 16, 2005.

V. Niola, C. Rossi, S. Savino: Modelling and Calibration of a Camera for Trajectories Recording -*Proc. 5th WSEAS Int. Conf. on Signal Processing, Robotics and Automation* – Madrid, February 15-17, 2006

V. Niola, C. Rossi, S. Savino: A Robot Kinematic Calibration Technique -*Proc. 5th WSEAS Int. Conf. on Signal Processing, Robotics and Automation* – Madrid, February 15-17, 2006

A. Fusiello: Visione Computazionale: appunti delle lezioni – Informatic Department, University of Verona, 3 March 2005.

R. Sharma, S, Hutchinson – "Motion perceptibility and its application to active vision-based servo control"-*Technical Report UIUC-BI AI RCV-94-05*, The Beckman Institute, Università dell'Illinois, 1994.

R. Sharma –"Active vision for visual servoing: a review - *IEEE Workshop on Visual Servoing: Achivement"* - Application and Open Problems, Maggio 1994.

R. Sharma, S, Hutchinson – "Optimizing hand/eye configuration for visual-servo system" - *IEEE International Confeence on Robotics and Automation* , pp. 172-177, 1995.

C. Mei: Camera Calibration Toolbox for Matlab -http://www.vision.caltech.edu/ bouguetj/calib_doc.

V. Niola, C. Rossi, S. Savino – "Modelling and calibration of a camera for robot trajectories recording" -*Proc. 5th WSEAS Int. Conf. on "Signal Processing, Robotics and Automation"* – Madrid, February 15-17, 2006.

A. Fusiello – "Visione Computazionale: appunti delle lezioni" – Informatic Department, University of Verona, 3 March 2005.

R. Brancati, C. Rossi, S. Scocca - Trajectories Recording for Investigations on Robot Dynamics and Mechanical Calibrations – *Proc. of 10th int. Workshop RAAD 2001*, Wien, May 16-19, 2001

R. Brancati, C. Rossi, S. Savino, A Method for Trajectory Planning in the Joint Space, *Proc. of 14th International Workshop on Robotics in Alpe-Adria-Danube Region,* Bucharest, Romania, May 26-28, 2005, pp.81-85.

R. Sharma, S. Hutchinson, On the observability of robot motion under active camera control, *Proc. IEEE International Conference on Robotics and Automation,* May 1999, pp. 162-167.

J. T. Feddema, C. S. George Lee, O. R. Mitchell, Weighted selection of image features for resolved rate visual feedback control, *IEEE Trans. Robot. Automat.,* Vol. 7, 1991, pp. 31-47.

V. Niola, C. Rossi, S. Savino – "Perspective Transform and Vision System for Robot Trajectories Recording" -*Proc. 5th WSEAS Int. Conf. on "Signal Processing, Robotics and Automation"* – Madrid, February 15-17, 2006.

15

Kinematic Control of A Robot-Positioner System for Arc Welding Application

Anatoly P. Pashkevich [1,2], Alexandre B. Dolgui [1]
[1]*Ecole de Mines de Saint Etienne*
France
[2]*Belarusian State University of Informatics and Radioelectronics*
Belarus

Welding is one of the most successful applications for industrial robots, which encourages intensive research and development in the area of the CAD-based and off-line programming (Pires et al., 2003, Yagi, 2004). At present, relevant software tools allow to optimise the process parameters, which directly influence the product quality and the system working cycle. Besides, they enable users to complete most of the process preparation actions in advance, without access to the workcell, and therefore, to make the robotic systems competitive for both large and small series, or even for unique products (Ranky, 2004). However, resent advances in the arc welding technology motivate rethinking of some postulates and conventions incorporated in the existing off-line programming methods. One of related problems, the kinematic control of a robot-positioner system, is addressed in this chapter.
The welding position (or, more precisely, the weld joint orientation relative to gravity) is an essential issue in both manual and robotic welding, associated with the problem of the weld puddle control. As is known from long-term experience, the highest quality and productivity are achieved for the downhand (or flat) welding position, where the workpiece is oriented so that the weld tangent line is horizontal, and the weld normal vector is the opposite of the gravity direction (Cary 1995). This orientation is preferable because gravity draws the molten metal downward into the joint allowing it to flow appropriately along the weld contour, which makes the welding faster and easier. For this reason, the downhand welding has been adopted in robotics as a de facto standard (Bolmsjo 1987, Fernandez and Cook 1988).
To ensure the desired weld orientation, a typical robotic arc welding station (Fig. 1) includes two separate moving mechanisms: (i) a five- or six-axis industrial robot (welding tool manipulator) aimed at moving the weld touch with the required speed and orientation relative to the weld joint; and (ii) a two- or three-axis positioning table (workpiece manipulator), which ensures the downhand (or close to it) orientation of the weld joint with respect to gravity.
In contrast to the robot, a skilled human-welder is capable to perform such operations in other positions, such as the horizontal, vertical, or overhead ones. To make such type of welding easier, several companies recently proposed their technological innovations, the flux-cored wires, that create a protective coating, supporting the metal against gravity. This

makes it possible to enlarge area of the non-downhand welding and employ for this industrial robots (Tolinski 2001). Besides, recent advances in computer vision have allowed enhancements in robot ability for the welding puddle control (Tarn et al. 2004). Hence, the existing kinematic control techniques for the arc welding robots must be revised in order to relax the downhand constraint.

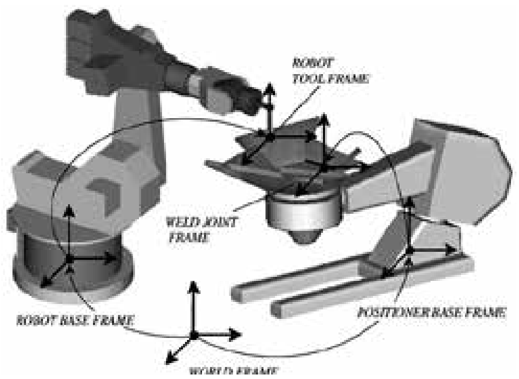


Fig. 1. Robotic arc welding station and its components.

2. Related Works

Automatic programming for robotic arc welding incorporates a number of nontrivial steps, ranging from specifying a manufacturing task to planning the robot-positioner motions (Kim et al. 1998, Bolmsjo 2002). They include, in particular, the welding feature extraction from CAD data, welding task planning and sequencing, coordinated robot-positioner control and numerous implementation issues (interfacing with the operator/controller, weldseam tracking, workcell calibration, etc.).
Since the beginning of robotic arc welding, many related studies focused on the kinematic control of the robot-positioner system. Theoretically, the arc welding requires (5+2)-axis manipulation, needed for the positioning/orienteering of the weld torch and the weld joint, respectively. However, because a standard industrial robot has 6 degrees-of-freedom, relevant papers cover topics from (6+2)- to (7+3)-axis kinematical structures. Accordingly, in all cases, the kinematic control problem is solved by imposing the following task-specific constraints:
(i) five constraints on the torch position and orientation, defined by three Cartesian coordinates of the welding tip and two Euler angles of the plasma flow-line;
(ii) two constraints on the weld joint orientation relative to gravity, defined as the verticality of the weld normal line ('downhand welding').
It is implicitly assumed here that both the torch rotation about the plasma flowline and the weld joint rotation about the vertical line are irrelevant to the welding process.
For the typical (6+2)-case, corresponding to a six-axis robot and two-axis balance or till-roll positioner, the problem has been studied by several authors (Bolmsjo 1987, Fernandez and Cook 1988, Nikoleris 1990, Kim et al. 1998). The common approach is based on the strictly downhand solution for the positioner inverse kinematics and the augmented solutions for the robot inverse kinematics, which depends on an arbitrary scalar parameter. Then, this free parameter is used for singularity or collision avoidance, optimisation of the manipulability, increase in robotic tool reach, etc. These results were extended to the (7+2)-

case by Ahmad and Luo (1989) whose study focused on a six-axis robot mounted on rail and a two-axis positioner; they also used the extra degrees of freedom to keep the robot out of singular configurations and to increase its reach. Recently, Wu et al. (2000) applied the genetic algorithm technique to solve a similar (7+3)-problem.

The positioner inverse kinematics, incorporated in the above control methods, was solved mainly for the strictly downhand weld orientation with respect to gravity. This essentially simplified the analytical expressions but was in certain disagreement with engineering practice that admits some tolerances. Hence, several authors considered more general formulations. In particular, Bolmsjo (1987) and Nikoleris (1990) stated the problem as aligning of any weld-associated and any given gravity-associated vectors. For this case, numerical and analytical solutions were obtained for both the roll-tilt and balance positioners. Later, the problem was reformulated by the authors (Pashkevich et al., 2003) and solved in terms of the weld slope-roll angles; the least-square solutions were also obtained for the case when exact ones do not exist.

In spite of common understanding, only Kim et al. (1998) have directly addressed the issue of the downhand-orientation tolerancing and its relation with the weld quality. These authors introduced the weld ‘stability’ metrics, thus allowing admissible orientations for the welding tool and welding joint (defined by work/ travel and slope/rotation angles, respectively) to be computed. The open question, however, is assigning reasonable ‘stability limits’ to obtain the required quality, which obviously needs a welding expert. Besides, no judgements on variations of the welding speed within the stability region have been proposed.

Another active area of the related research is the arc welding operations planning and sequencing. These works concentrate on specific non-trivial cases of the travelling-salesman problem (TSP) known from combinatorial optimisation (see Gutin and Punnen, 2002 for details and latest results). For arc welding, the scheduling problem with the minimum-time objective was first addressed by Rubinovitz and Wysk (1988), who suggested a heuristic algorithm based on the classical TSP nearest-neighbour method. Then, Fukuda and Yoshikawa (1990) applied to this problem a neural network technique focusing on reduction in the welding distortions. Later, the TSP-based method was enhanced by Kim et al. (1998, 2002a), who proposed several heuristics and genetic algorithms, which avoid the distortions caused by heat by imposing the problem-specific non-precedence constraints. In another paper, Kim et al. (2002b) reformulated the heat-affected zone constraint by setting the lower bound on the travel time between two consecutive welding operations (‘cooling time’) and proposed several heuristics based on the nearest-neighbour, 2-opt, 3-opt, and tabu search methods. Grenestedt (2003) applied to this problem the simulating annealing in combination with the approximate analytical distortion models. In the latest paper by Kim et al. (2005), there several enhanced welding sequencing heuristics were proposed, which also adopt the ‘cooling time’ concept.

This chapter summarises the authors’ results in the area of the robotic arc welding (Pashkevich et al., 2003; Dolgui and Pashkevich, 2006) and presents techniques for both the closed-form inverse kinematics solutions and optimal planning of the welding sequences. It contributes to the research originated from the papers of Bolmsjo (1996), Nikoleris (1990) and Kim et al. (1998), assuming that the downhand constraint is relaxed and implicitly converted into the penalty function, which increases the welding time depending on the ‘non-downhand’ degree. The objective is to minimize the overall manufacturing cycle, by finding a reasonable trade-off between the positioner motion time and the time required for the welding itself.

3. Kinematic Control Architecture

3.1 Control Hierarchy

In contrast to the early robotic manipulators, in which capabilities were limited by the servo-control of separate joint axes, the modern industrial robotic systems should implement the task-level control that essentially simplifies the manufacturing task definition for the end user. It results in including a kinematic control module as a built-in part of the hierarchical control system, where the high-level command is sequentially decomposed to the lower level ones, up to the axis drives and the process variable controllers. However, in spite of the apparent simplicity, defining of a particular content of each control level requires development of specific mathematical methods that take into account particularities of the relevant technology.

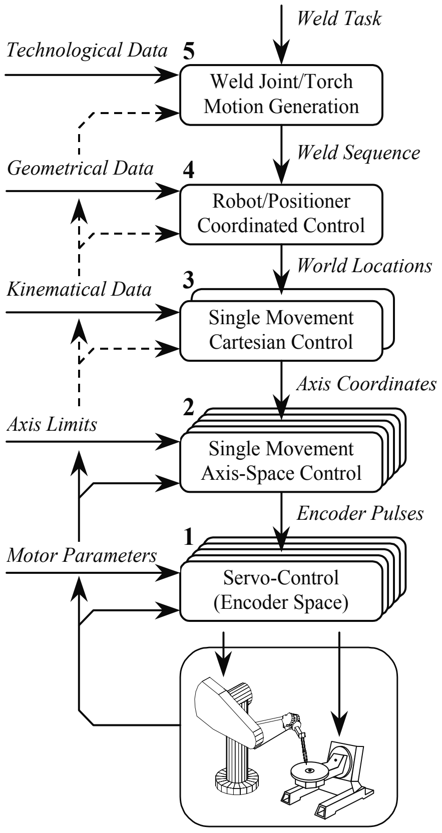

Weld Task
Technological Data
5
Weld Joint/Torch Motion Generation
Weld Sequence
Geometrical Data
4
Robot/Positioner Coordinated Control
World Locations
Kinematical Data
3
Single Movement Cartesian Control
Axis Coordinates
Axis Limits
2
Single Movement Axis-Space Control
Encoder Pulses
Motor Parameters
1
Servo-Control (Encoder Space)

Fig. 2. Multi-level control hierarchy.

For the robotic arc welding, five levels of control are typically used (Fig. 2). The highest of them highly relies on the kinematic modelling and deals with obtaining the optimal technological and geometrical parameters, such as the orientation angles of the weld joint and the welding gun, the weld sequence, the weld speed, etc. The fourth level performs appropriate coordinate transformations via the direct/inverse kinematics of all mechanical components (robot, positioner, gantry). The remaining three levels deal with the implementing of tool/workpiece movements in the Cartesian space, in the manipulator axis space and, finally, in the motor shaft space. (It should be noted that for some manipulators, the interrelation between the joint axis angle and the motor shaft angle is non-trivial.) At present, control techniques for all the abovementioned hierarchical levels are being intensively developed. For instance, advanced commercial controllers already include the forward dynamic models, which essentially improve the operational speed and accuracy. However, various aspects of the fourth and the fifth control levels are still subject of intensive research.

3.2. Kinematic Description of the Welds

The spatial location of the welding object, as a general rigid body, can be defined by a single frame that incorporates six independent parameters (three Cartesian coordinates and three Euler angles). However, defining geometry of each weld requires some additional efforts, depending on the joint profile. Since capabilities of modern industrial robotic systems allow processing two basic types of the contours (linear and circular), only these cases are considered below.

For the linear joints, a moving frame with the specific definition of the axes can describe the weld geometry. In this chapter, this frame is defined so that (Fig. 3):

- The X_w- axis is directed along the weld joint (welding direction);
- The Y_w-axis is normal to the weld joint (weld torch approaching direction);
- The Z_w-axis completes the right-hand oriented frame ("weaving" direction).

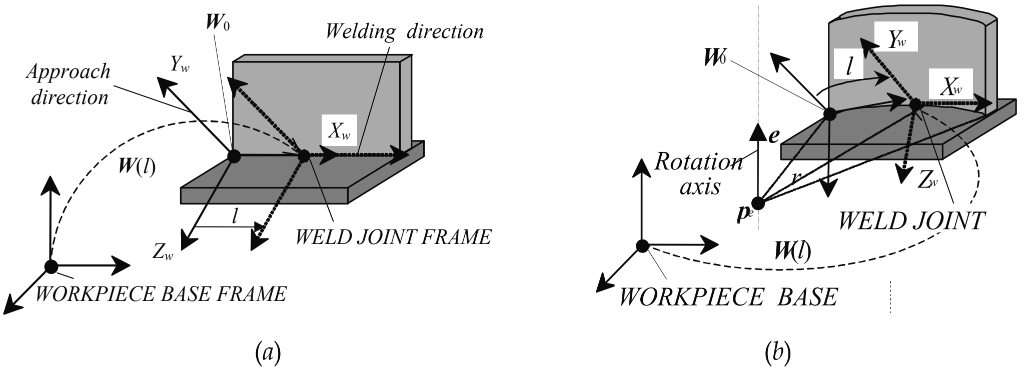


Fig. 3. Definition of the weld frames for a liner (*a*) and circular (*b*) welds.

It should be noted that, in practice, it is prudent to define the Y_w -axis as the bisectrix of the corresponding weld joint surfaces.

Taking into account the above definitions, the kinematic model of the linear weld relative to the *WB*-frame (i.e., the workpiece base frame, see Fig. 3a) can be described by the following homogenous parametric equation

$$^{WB}\boldsymbol{W}(l) = \begin{bmatrix} \boldsymbol{n}_w^s & \boldsymbol{s}_w^s & \boldsymbol{n}_w^s \times \boldsymbol{s}_w^s & \boldsymbol{p}_w^s + l \cdot \boldsymbol{n}_w^s \\ 0 & 0 & 0 & 1 \end{bmatrix}_{4\times4}, \qquad (1)$$

where the parameter *l* is the welding torch displacement, the left superscript "*WB*" refers to the workpiece base coordinate system, the right superscript "*s*" and the subscript "*w*" denote starting point of the weld, $\boldsymbol{n}_w^s$ is the unit vector of the welding direction (axis X_w), $\boldsymbol{s}_w^s$ is the unit vector of the approaching direction (axis Y_w), and $\boldsymbol{p}_w^s$ is the position vector of the

weld starting point. It should be stressed that the vectors $\boldsymbol{n}_w^s$; $\boldsymbol{s}_w^s$; $\boldsymbol{p}_w^s$ are defined relative to the *WB*-frame and, in practice, they are easily derived from the workpiece 3D CAD model.
For the circular joints, a similar approach is used, but the moving frame is computed to ensure the tangency of the welding path and the X_w-axis at every point (Fig. 3b). It is evident that the initial frame is subject to the rotational transformation and the weld kinematics is described by the following parametric equation:

$$^{WB}\boldsymbol{W}(l)=\begin{bmatrix}\boldsymbol{R}_e(l/r)\cdot\boldsymbol{R}_w^e & \boldsymbol{R}_e(l/r)\cdot(\boldsymbol{p}_w^e-\boldsymbol{p}_e)+\boldsymbol{p}_e\\ \boldsymbol{0}_{1\times3} & 1\end{bmatrix}_{4\times4}, \tag{2}$$

where the parameter *l* and the sub/superscripts "*WB*", "*w*", "*s*" have the same meaning as in (1), the orthogonal 3×3 matrix is expressed as $\boldsymbol{R}_e=[\boldsymbol{n}_w^s \;\; \boldsymbol{s}_w^s \;\; \boldsymbol{n}_w^s\times\boldsymbol{s}_w^s]$ and defines the orientation of the weld frame at the starting point, *r* is the radius of the circular welding joint, $\vartheta=l/r$ is the angle of rotation, the vector $\boldsymbol{p}_e$ defines the position of the circle centre, and $\boldsymbol{R}_e(\vartheta)$ the general rotation matrix (Fu et al., 1987) around the axis, which is determined by the unit vector $\boldsymbol{e}=[e_x,e_y,e_z]$ (see Fig. 4):

$$\boldsymbol{R}_e(\vartheta)=\begin{bmatrix}e_x^2V_\vartheta+C_\vartheta & e_xe_yV_\vartheta-e_zS_\vartheta & e_xe_zV_\vartheta+e_yS_\vartheta\\ e_xe_yV_\vartheta+e_zS_\vartheta & e_y^2V_\vartheta+C_\vartheta & e_ye_zV_\vartheta-e_xS_\vartheta\\ e_xe_zV_\vartheta-e_yS_\vartheta & e_ye_zV_\vartheta+e_xS_\vartheta & e_z^2V_\vartheta+C_\vartheta\end{bmatrix}_{3\times3} \tag{3}$$

As in the previous case, the required vectors $\boldsymbol{n}_w^s$; $\boldsymbol{s}_w^s$; $\boldsymbol{p}_w^s$ and $\boldsymbol{e}$, $\boldsymbol{p}_e$ as well as the radius *r*; may also be easily derived from the workpiece 3D model using capabilities of the modern graphical simulation systems to generate straight lines, planes and circles.
Thereby, expressions (1)–(3) completely define spatial location (i.e., the position and the orientation) of the weld joint relative to the *WB*-frame (workpiece base), which should be adjusted by the positioner to optimise the weld orientation relative to the gravity (see Fig. 1). Hence, the absolute (world) location of the weld joint is described by the product of the homogenous matrices

$$^{0}\boldsymbol{W}(l)=\left[{}^{0}\boldsymbol{T}_{PB}\cdot\boldsymbol{P}(\boldsymbol{q})\cdot{}^{PF}\boldsymbol{T}_{WB}\right]\cdot{}^{WB}\boldsymbol{W}(l) \tag{4}$$

where the left superscript "0" refers to the world coordinate system, the matrix ${}^{0}\boldsymbol{T}_{PB}$ defines the absolute (world) location of the positioner base *PB*; the matrix ${}^{PF}\boldsymbol{T}_{WB}$ describes the workpiece base *WB* location relative to the positioner mounting flange *PF*; and the matrix function $\boldsymbol{P}(\boldsymbol{q})$ is the positioner direct kinematic model, while $\boldsymbol{q}$ is the vector of the positioner joint coordinates.
To ensure good product quality and to increase the welding speed, the weld joint should be properly oriented relative to the gravity. The exact interrelations between these parameters are not sufficiently well known and require empirical study in each particular case. But practising engineers have developed a rather simple rule of thumb that is widely used for both the online and off-line programming: "*the weld should be oriented in the horizontal plane so that the welding torch is vertical, if possible*" (Bolmsjo, 1987). It is obvious that the CAD-based approach requires numerical measures of the "horizontality" and the "verticality", which are proposed below.

Let us assume that the Z_0-axis of the world coordinate system is strictly vertical (i.e. directed opposite to the gravity vector), and, consequently, the X_0Y_0-plane is horizontal. Then, the weld orientation relative to the vector of gravity can be completely defined by two angles (Fig. 4):

- The weld slope $\theta \in [-\pi/2; \pi/2]$, i.e. the angle between the vector of the welding direction ${}^0\boldsymbol{n}_w$ and the Cartesian plane X_0Y_0;
- The weld roll $\xi \in (-\pi; \pi]$, i.e. the angle between the vector of the approaching direction ${}^0\boldsymbol{s}_w$ and the vertical plane that is parallel to the vector ${}^0\boldsymbol{n}_w$ and the Cartesian axis Z_0.

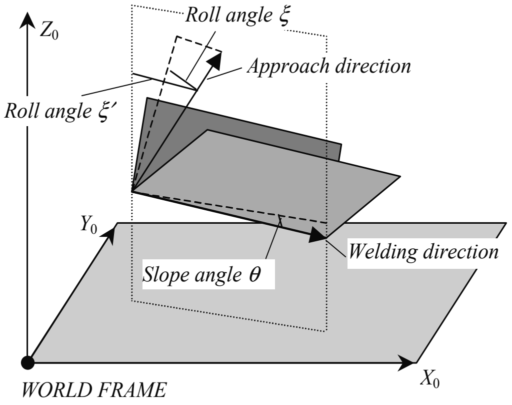


Fig. 4. Definition of the weld orientation angles.

The numerical expressions for θ, ξ can be obtained directly from the definition of the *RPY*-angles (Fu et al., 1987), taking into account that the weld orientation (θ, ξ) = (0, 0) corresponds to the horizontal direction of the axis X_w and the vertical direction of the Y_w (see Fig. 5):

$$ {}^0\boldsymbol{W}_R = \boldsymbol{R}_z(\psi)\cdot \boldsymbol{R}_y(\theta)\cdot \boldsymbol{R}_x(\pi/2-\xi) \tag{5}$$

where ${}^0\boldsymbol{W}_R$ is the 3×3 orientation submatrix of the 4×4 matrix of the weld location; $\boldsymbol{R}_x$; $\boldsymbol{R}_y$; $\boldsymbol{R}_z$ are the 3×3 rotation matrices around the axes *X*; *Y*; *Z*; respectively, and ψ is the yaw angle which is non-essential for the considered problem. Multiplication of these matrices leads to

$$ {}^0\boldsymbol{W}_R = \begin{bmatrix} C_\psi C_\theta & C_\psi S_\theta C_\xi - S_\psi S_\xi & C_\psi S_\theta S_\xi + S_\psi C_\xi \\ S_\psi C_\theta & S_\psi S_\theta C_\xi + C_\psi S_\xi & S_\psi S_\theta S_\xi - C_\psi C_\xi \\ -S_\theta & C_\theta C_\xi & C_\theta S_\xi \end{bmatrix}$$

where *C* and *S* denote respectively *cos* (.) and *sin*(.) of the corresponding angle specified at the subscript. Therefore, the weld joint orientation angles θ, ξ can be derived as follows:

$$\theta = atan2\frac{-{}^0n_w^z}{\sqrt{({}^0s_w^z)^2+({}^0a_w^z)^2}};\quad \xi = atan2\frac{{}^0a_w^z}{{}^0s_w^z} \tag{6}$$

where ${}^0\boldsymbol{n}_w$, ${}^0\boldsymbol{s}_w$, ${}^0\boldsymbol{a}_w$ are the corresponding column vectors of the orthogonal matrix ${}^0\boldsymbol{W}_R$. Taking into account interrelations between these vectors, the angles θ, ξ can be finally expressed as functions of the weld joint direction ${}^0\boldsymbol{n}_w$ and approaching direction ${}^0\boldsymbol{s}_w$

$$\theta = atan2\frac{-{}^0n_w^z}{\sqrt{({}^0n_w^x)^2+({}^0n_w^y)^2}}; \tag{7}$$

$$\xi = atan2 \frac{{}^0n_w^x \cdot {}^0s_w^y - {}^0n_w^y \cdot {}^0s_w^x}{{}^0s_w^z} \tag{8}$$

It should be noted that it is possible to introduce *alternative* definition of the *weld roll*, which is non-singular for all values of the weld slope. It is $\xi' \in [0; \pi]$, which is the angle between the approaching direction ${}^0\boldsymbol{s}_w$ and the vertical axis Z_0 (see Fig. 4):

$$\xi' = \operatorname{a\tan2} \frac{\sqrt{\left({}^0s_w^x\right)^2 + \left({}^0s_w^y\right)^2}}{{}^0s_w^z} \tag{9}$$

As in the case of angles (θ, ξ), the description (θ, ξ') also defines the 3rd row of the weld joint orientation matrix ${}^0\boldsymbol{W}_R$, but the sign of the $\boldsymbol{a}_w^z$ may be chosen arbitrary. Hence, the interrelation between both the definitions of the roll angle ξ and ξ' is given by the equation

$$cos(\theta) \cdot cos(\xi) = cos(\xi'), \tag{10}$$

and both (θ, ξ) and (θ, ξ') can be used equally.

4. Weld Joint Orienting Problems

In the robotic welding station, the desired orientation of the weld relative to the gravity is achieved by means of the positioner, which adjusts the slope and the roll angles by alternating its axis coordinates. Using the kinematic model (4) and the definitions from the previous section, the problems of the welding joint orientation can be stated as follows.

Direct Problem. *For given values of the positioner* axis *coordinates* $\boldsymbol{q}$, *as well as known homogenous transformation matrices* ${}^0\boldsymbol{T}_{PB}$, ${}^{PF}\boldsymbol{T}_{WB}$ *and the weld frame location relative to the object base* $\boldsymbol{W}$, *find the weld frame orientation in the world coordinate system* ${}^0\boldsymbol{W}$ *and the slope/roll orientation angles* (θ, ξ).

Inverse Problem 1. *For given values of* the *slope/roll orientation angles* (θ, ξ), *as well as known homogenous transformation matrices* ${}^0\boldsymbol{T}_{PB}$, ${}^{PF}\boldsymbol{T}_{WB}$ *and the weld frame location relative to the object base* $\boldsymbol{W}$, *find the values of the positioner* axis *coordinates* $\boldsymbol{q}$.

There is also another version of the inverse problem for the welding positioner (Nikoleris, 1990) that deals with a reduced version of the expression (4), which describes only a single unit vector transformation

$${}^0\mathbf{s}_w = \left[{}^0\mathbf{T}_{PB} \cdot \mathbf{P}(\mathbf{q}) \cdot {}^{PF}\mathbf{T}_{WB} \right]_{3\times3} \cdot \mathbf{s}_w{}' \tag{11}$$

Using the accepted notations, this formulation can be stated as follows:

Inverse Problem 2. *For given values of* the *world coordinates of the weld approach vector* ${}^o\boldsymbol{s}_w$, *as well as for known homogenous transformation matrices* ${}^0\boldsymbol{T}_{PB}$, ${}^{PF}\boldsymbol{T}_{WB}$ *and the normal vector orientation relative to the object base* $\boldsymbol{s}_w$, *find the values of the positioner* axis *coordinates* $\boldsymbol{q}$.

It should be stressed that both the formulations require two independent input parameters (two angels or a unit vector); however, they differ by the elements of the matrix ${}^o\boldsymbol{W}_R$ they deal with. Thus, the first formulation deals with the third row of the matrix ${}^0\boldsymbol{W}_R$, which includes only $\boldsymbol{Z}$-coordinates [n_z s_z a_z] that are not sensitive to the rotation around the gravity. In contrast, the second formulation operates with second column of this matrix $[s_x\ s_y\ s_z]^T$, which incorporates X,Y-coordinates that are sensitive to mentioned rotation. As a result, the latter approach does not

allow achieving desired weld slope and roll simultaneously. Therefore, the second formulation of the inverse problem is less reasonable from technological point of view.
The only case when the second formulation is sensible, is the "optimal weld orientation", for which the approaching vector is strictly vertical and, consecutively, the weld direction vector ${}^0\boldsymbol{n}_w$ lies in the horizontal plane. But the first formulation also successfully tackle this case, as it corresponds to the $(\theta, \xi)=(0,0)$. However, the second formulation can be successfully applied in the singular for the first approach case $(\theta=\pm\pi/2)$, when defining the roll angle does not make sense. For this reason, both formulations of the inverse problem will be considered below.
While applying the inverse formulation to real-life problems, it should also be taken into account that engineering meaning of the slope and the roll is not sensitive to the sign of this angles. For instance, the negative slope can be easily replaced by the positive one, if the weld starting and ending point are interchanged. Also, the positive and negative rolls are equivalent with respect to gravity forces. Therefore, four cases $(\pm\theta, \pm\xi)$ must be investigated while orienting the weld joint. Similar conclusion is valid for the alternative definition of the weld orientation angles (θ, ξ'), where $\xi'>0$ but two cases $(\pm\theta, \xi')$ yield four different matrices $\boldsymbol{W}_R$.

4.1 Direct Kinematic Problem

As follows from (4), successive multiplication of the corresponding homogenous matrices gives, for given axis coordinates $\boldsymbol{q}$, the full world location (position and orientation) of the weld frame. Then, the required angles (θ, ξ) or (θ, ξ') are extracted from the matrix ${}^0\boldsymbol{W}$ in accordance with the expressions (6)-(9). Therefore, the only problem is to find the matrix $\boldsymbol{P}(\boldsymbol{q})$ that describes transformation from the positioner base to the its mounting flange (or face plate).
Because the weld joint orientation relative to the gravity is completely defined by two independent parameters, a universal welding positioner has two axes. Though, the simplest robotic cells utilise a one-axis positioners (turntables and turning rolls) that are not capable to provide full weld orientation but also increase potential of the welding station. Robotic manufactures also produce five-axis positioners that are in fact combination of two two-axis machines that are moved to the robot workspace in turn (using the 5th axis), to make possible to change the workpiece while the robot is welding the other side. Therefore, a two-axis positioner can be considered as a basic orienting component of a welding station, so the reminder of this section is devoted to positioners with two d.o.f.

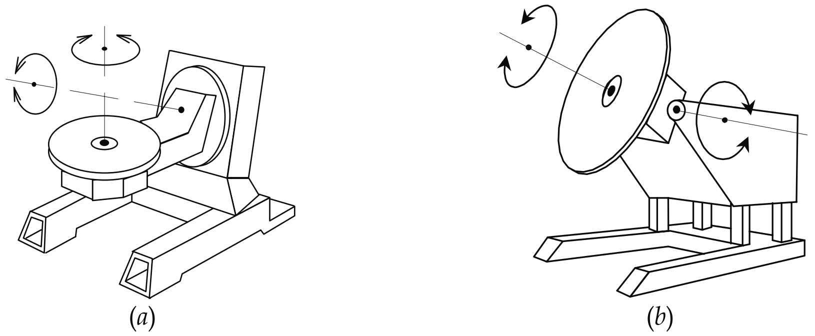


Fig. 5. The two-axis balance (*a*) and roll-tilt (*b*) positioners.

While building the positioner model, it should be taken into account that the intersection point of the axes may be located above the faceplate, to be closer to the workpiece centre of gravity (Fig. 5a).

Such design allows avoiding large payload moments specific for heavy and bulky objects. But in some cases this point may lie above the plate (Fig. 5b). For this reason, it is prudent to release the usual constraint that locates the positioner base frame at the intersection of its two axes.
The kinematic model of a general 2-axis positioner is presented in Fig. 6. It includes four linear parameters (a_1, d_1, a_2, d_2) and one angular parameter α that defines direction of the *Axis*$_1$. Without loss of generality, the *Axis*$_2$ is assumed to be normal to the faceplate and directed vertically for q_1=0. The geometrical meanings of the parameters are clear from the figure.
Similar to other manipulators, the kinematics of a positioner can be described by the Denavit-Hartenberg model (Fu et al., 1987). However, for the considered 2-axis system it is more convenient to use a product of elementary transformations that can be derived directly from the Fig. 7:

$$\boldsymbol{P}(q_1,q_2) = {}^{PB}\boldsymbol{T}_1 \cdot \boldsymbol{R}_x(q_1) \cdot {}^1\boldsymbol{T}_2 \cdot \boldsymbol{R}_z(q_2) \tag{12}$$

where ${}^{PB}\boldsymbol{T}_1 = \boldsymbol{T}_x(a_1)\cdot\boldsymbol{T}_z(d_1)\cdot\boldsymbol{R}_y(-\alpha)$; ${}^1\boldsymbol{T}_2 = \boldsymbol{R}_y(\alpha)\cdot\boldsymbol{T}_x(a_2)\cdot\boldsymbol{T}_z(d_2)$, and $\boldsymbol{T}(.)$, $\boldsymbol{R}(.)$ are the 4×4 homogenous transformation matrices that describe translation/rotation with respect to the axes specified by the subscript.

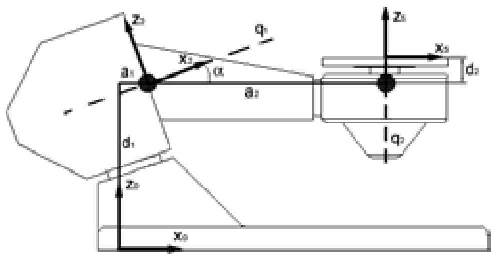


Fig. 6. The coordinate frames of the two-axis positioner.

Substituting in (12) regular expressions for translational and rotational matrices yields the final result for the not-trivial components of the positioner transformation matrix $\boldsymbol{P}(q_1,q_2)$:

$$n_x = \left(C_1 + C_\alpha^2 V_1\right)C_2 - S_\alpha S_1 S_2\,;\quad n_y = S_\alpha S_1 C_2 + C_1 S_2\,;\quad n_z = C_\alpha S_\alpha V_1 C_2 + C_\alpha S_1 S_2 \tag{13}$$

$$s_x = -\left(C_1 + C_\alpha^2 V_1\right)S_2 - S_\alpha S_1 C_2\,;\quad s_y = -S_\alpha S_1 S_2 + C_1 C_2\,;\quad s_z = -C_\alpha S_\alpha V_1 S_2 + C_\alpha S_1 C_2\,; \tag{14}$$

$$a_x = C_\alpha S_\alpha V_1\,;\quad a_y = -C_\alpha S_1\,;\quad a_z = C_1 + S_\alpha^2 V_1\,; \tag{15}$$

$$\begin{aligned} p_x &= \left(C_1 + C_\alpha^2 V_1\right)\cdot a_2 + C_\alpha S_\alpha V_1 \cdot d_2 + a_1 \\ p_y &= S_\alpha S_1 \cdot a_2 - C_\alpha S_1 \cdot d_2 \\ p_z &= C_\alpha S_\alpha V_1 \cdot a_2 + \left(C_1 + S_\alpha^2 V_1\right)\cdot d_2 + d_1 \end{aligned} \tag{16}$$

where, similarly to the section 2, vectors ***n***, ***s***, ***a***, ***p*** define the upper 3×4 block of the matrix ***P***, and *C, S, V* denote respectively *cos*(.), *sin*(.), *vers*(.) of the angle specified at a subscript. It should be noted, that compared to the model proposed by G.Bolmsjo (1987), the developed one includes less geometrical parameters while also describes the general case. Besides, the obtained expressions are less awkward and more computationally efficient than the known ones.
Therefore, expressions (13)-(16) completely define direct kinematics of the 2-axis positioner. But the obtained model can be also reduced to describe kinematics of a general 1-axis mechanism. It is achieved by fixing the *Axis*$_1$ or *Axis*$_2$ and choosing appropriate value of α. For instance, for turntables the axis variable is q_2 while q_1=0. But for turning rolls the axis variable is q_1 while q_2=0 and α=0.

5. Inverse Kinematic Problems

In accordance with Section 3, solving the inverse kinematic problem for the positioner means finding the axis angles (q_1, q_2) that ensure the desired world orientation of the weld joint, which is defined by the pair of the orientation angles (Problem 1) or by the unit vector (Problem 2). Let us consider these cases separately.

5.1. Solution of the Inverse Problem 1

Since the weld joint orientation angles (θ, ξ) or (θ, ξ′) completely define the third row of the orthogonal 3×3 matrix $^0\boldsymbol{W}_R$, the basic kinematic equation (4) can be rewritten as

$$\boldsymbol{\eta}^T \cdot {}^0\boldsymbol{W}_R = \boldsymbol{\eta}^T \cdot \left[{}^0\boldsymbol{T}_{PB} \cdot \boldsymbol{P}(\boldsymbol{q}) \cdot {}^{PF}\boldsymbol{T}_{WB}\right]_{3\times3} \cdot \boldsymbol{W}_R \,, \tag{17}$$

where the subscript "3 × 3" denotes the rotational part of the corresponding homogenous transformation matrix, and $\boldsymbol{\eta}^T$=[0 0 1]. Then, after appropriate matrix multiplications, it can be converted to the form

$$\boldsymbol{v}^T = \boldsymbol{\eta}^T \cdot \boldsymbol{P}(\boldsymbol{q})_{3\times3} \,, \tag{18}$$

where $\boldsymbol{v}^T = \left[-S_\theta \quad C_\theta C_\xi \quad C_\theta S_\xi\right] \cdot \left[{}^{PF}\boldsymbol{T}_{WB} \cdot \boldsymbol{W}\right]^T_{3\times3}$ and, without loss of generality, transformation $^0\boldsymbol{T}_{PB}$ is assumed not to include the rotational components other then $\boldsymbol{R}_z$. Further substitution in accordance with (13) yields three mutually dependent scalar equations of two unknowns (q_1, q_2):

$$C_\alpha S_\alpha V_1 C_2 + C_\alpha S_1 S_2 = v_x \,; \quad C_\alpha S_1 C_2 - C_\alpha S_\alpha V_1 S_2 = v_y \,; \quad C_1 + S_\alpha^2 V_1 = v_z \tag{19}$$

where v_x, v_y, v_z are the corresponding components of the vector $\boldsymbol{v}$. The third of these equations can be easily solved for q_1:

$$q_1 = \pm\, acos \frac{v_z - S_\alpha^2}{C_\alpha^2} \tag{20}$$

The value of q_2 can be found by solving the first and the second equations for C_2 and S_2:

$$C_2 = \left(S_1 \cdot v_y + S_\alpha V_1 \cdot v_x\right) / C_\alpha \left(S_\alpha^2 V_1^2 + S_1^2\right); \quad S_2 = \left(S_1 \cdot v_x - S_\alpha V_1 \cdot v_y\right) / C_\alpha \left(S_\alpha^2 V_1^2 + S_1^2\right).$$

that leads to the following expression for q_2:

$$q_2 = atan2 \frac{S_1 \cdot v_x - S_\alpha V_1 \cdot v_y}{S_1 \cdot v_y + S_\alpha V_1 \cdot v_x} \tag{21}$$

Therefore, Eqs. (18) and (19) represent the closed-form solution of the first inverse problem, which in the general case for given weld orientation angles (θ, ξ) or (θ, ξ′) yields two pares of the positioner axis angels (q_1, q_2).

5.2. Solution of the Inverse Problem 2

For the second formulation, the input data defines the second column of the matrix $^0\boldsymbol{W}_R$, so the basic kinematic equation (4) can be rewritten as follows:

$$ {}^{0}\boldsymbol{W}_{R}\cdot\boldsymbol{\eta} = \left[{}^{0}\boldsymbol{T}_{PB}\cdot\boldsymbol{P}(\boldsymbol{q})\cdot{}^{PF}\boldsymbol{T}_{WB}\right]_{3\times3}\cdot\boldsymbol{W}_{R}\cdot\boldsymbol{\eta}, \tag{22}$$

where $\eta=[0\;1\;0]^{T}$. Then, after appropriate matrix multiplications, this equation can be converted to the form

$$\boldsymbol{u} = \boldsymbol{P}(\boldsymbol{q})_{3\times3}\cdot\boldsymbol{w}, \tag{23}$$

where

$$\boldsymbol{w} = \left[{}^{PF}\boldsymbol{T}_{WB}\cdot\boldsymbol{W}\right]_{3\times3}\cdot\boldsymbol{\eta}; \qquad \boldsymbol{u} = \left[{}^{0}\boldsymbol{T}_{PB}\right]^{T}_{3\times3}\cdot{}^{0}\boldsymbol{W}_{R}\cdot\boldsymbol{\eta}\cdot$$

and the subscript "3×3" means the upper left 3×3 submatrix of the corresponding homogenous matrix (i.e. its orthogonal rotational part).
Further expansion of $\boldsymbol{P}(\boldsymbol{q})$ in accordance with (12) and relevant regrouping yields

$$\left[\boldsymbol{R}_y(-\alpha)\cdot\boldsymbol{R}_x(q_1)\boldsymbol{R}_y(\alpha)\right]_{3\times3}\cdot\boldsymbol{u} = \left[\boldsymbol{R}_z(q_2)\right]_{3\times3}\cdot\boldsymbol{w}, \tag{24}$$

or, in a detailed form,

$$\begin{bmatrix} (1-S_\alpha^2 V_1) & S_\alpha S_1 & S_\alpha C_\alpha V_1 \\ -S_\alpha S_1 & C_1 & C_\alpha S_1 \\ S_\alpha C_\alpha V_1 & -C_\alpha S_1 & (1-C_\alpha^2 V_1) \end{bmatrix}\cdot\begin{bmatrix} u_x \\ u_y \\ u_z \end{bmatrix} = \begin{bmatrix} C_2 & -S_2 & 0 \\ S_2 & C_2 & 0 \\ 0 & 0 & 1 \end{bmatrix}\cdot\begin{bmatrix} w_x \\ w_y \\ w_z \end{bmatrix}$$

It leads to the following scalar equations:

$$\begin{gathered}(1-S_\alpha^2 V_1)u_x + S_\alpha S_1 u_y + S_\alpha C_\alpha V_1 u_z = C_2 w_x - S_2 w_y \quad -S_\alpha S_1 u_x + C_1 u_y + C_\alpha S_1 u_z = S_2 w_x + C_2 w_y \\ S_\alpha C_\alpha V_1 u_x - C_\alpha S_1 u_y + (1-C_\alpha^2 V_1)u_z = w_z\end{gathered} \tag{25}$$

from which the third one can be transformed to the form

$$C_\alpha u_{xz}\cdot C_1 + C_\alpha u_y\cdot S_1 = (u_z - w_z) + C_\alpha u_{xz}$$

and solved for q_1:

$$q_1 = atan2\frac{u_y}{u_{xz}} \pm acos\frac{(u_z-w_z)+C_\alpha u_{xz}}{C_\alpha\cdot\sqrt{u_{xz}^2+u_y^2}}, \tag{26}$$

where $u_{xz} = S_\alpha u_x - C_\alpha u_z$. It should be noted that these two alternative solutions for q_1 correspond to different "*configurations*" of a positioner, which are strictly defined below. Besides, both the solutions must be adjusted to the feasible interval (-π,π], since the sum of *atan*2(.) and *acos*(.) can be out of the mentioned limits.
To find the value of q_2, let us consider the first two equations of system (25) and solve them for C_2 and S_2:

$$\begin{aligned} C_2 &= (w_x\cdot v_x + w_y\cdot v_y)/(w_x^2+w_y^2); \\ S_2 &= (w_x\cdot v_y - w_y\cdot v_x)/(w_x^2+w_y^2), \end{aligned} \tag{27}$$

where

$$v_x = u_x + S_\alpha(S_1 u_y - V_1 u_{xz}); \quad v_y = C_1 u_y - S_1 u_{xz}\cdot$$

It leads to the following expression for q_2:

$$q_2 = atan2\frac{w_x \cdot v_y - w_y \cdot v_x}{w_x \cdot v_x + w_y \cdot v_y}\cdot \tag{28}$$

Therefore, Eqs. (26) and (28) represent the closed-form solution of the second inverse problem, which in the general case for given unit vectors (***u***, ***w***) yields two pares of the positioner axis angels (q_1, q_2).

5.3. Solution Existence and Singularities

As follows from Eqs. (20), (21) and (26), (28), the inverse kinematic problems possess solutions for certain sets of input data that can be treated as the positioner *"orientation workspace"*. So, for some inputs, the computation may fail and a solution does not exist (if, for instance, the *cos* argument is out of the interval [-1; 1]). In other cases, the *singularities* arise if any value of q_1 or q_2 satisfies the kinematic equation (if, for example, both arguments of *atan*2 are equal to zero). For the first inverse problem, a detailed investigation of Eq. (20) shows that the value of q_1 can be definitely computed if and only if

$$-cos(2\alpha) \le v_z \le 1\cdot \tag{29}$$

Taking into account the geometrical meaning of v_z , which is the scalar product of the unit vectors extracted from the third rows of the orthogonal matrices ${}^{o}\boldsymbol{W}_R$ and $[{}^{PF}\boldsymbol{T}_{WB} \cdot \boldsymbol{W}_R]$ (see equations (17), (18)) , and denoting

$$v_z = cos(\chi); \quad \chi \in [0; \pi] \tag{30}$$

this condition can be presented as follows:

Proposition 1a. *For the Inverse Problem* 1, *the values of* q_1 *can be computed definitely from the expression* (20), *if and only if the angle* χ *between Z-axes of the conjugate frames* ${}^{o}\boldsymbol{W}_R^{T}$ *and* $[{}^{PF}\boldsymbol{T}_{WB} \cdot \boldsymbol{W}_R]^{T}$ *describing, respectively, the desired world orientation of the weld joint and its orientation relative to the positioner faceplate is less than* (π-2α) *or equal to it*:

$$0 \le \chi \le \pi - 2\alpha\cdot \tag{31}$$

For a typical industrial application case, when the *Z*-axis of the workpiece frame is parallel to the positioner *Axis*$_2$, expression (29) can be also rewritten as

$$v_z = -S_\theta \cdot n_w^z + C_\theta C_\xi \cdot s_w^z + C_\theta S_\xi \cdot a_w^z \ge -C_{2\alpha} \tag{32}$$

The corresponding value of q_2 is uniquely defined by expression (21) if either its numerator or denominator is not equal to zero. A detailed investigation of the opposite case yields v_z=1 and consequently q_1=0 (case of v_z = -1 is excluded because of inequality (29)). So, the existence and uniqueness of solutions for q_2 are subject to the following proposition:

Proposition 1b. *For Inverse Problem* 1, *the value of* q_2 (*for given* q_1) *can be computed uniquely from the Eq.* (21), *if and only if the Z-axes of the conjugate frames* ${}^{o}\boldsymbol{W}_R^{T}$ and $[{}^{PF}\boldsymbol{T}_{WB} \cdot \boldsymbol{W}_R]^{T}$ *are not coincide, i.e.* χ>0. *Otherwise,* if *these axes coincide* (i.e. χ=0), *then* q_1=0 *and any value of* q_2 *satisfy the kinematic equation*.

Therefore, for the first inverse problem, the singularity exists only with respect to the positioner *Axis*$_2$, while it is oriented strictly vertically and upward (i.e. when q_1 = 0).

However, for the *second inverse problem*, the singularity may also arise for *Axis*$_1$. As follows from the analysis of Eq. (26), the *atan*2 is indefinite if u_{xz}=0 and u_y=0. Moreover, the corresponding kinematic equations are converted to the identity, if $u_z = w_z$. So any value of q_1 is a solution for such input data. The corresponding condition can also be presented as the parallelism of the vector $\boldsymbol{u}$ and *Axis*$_1$, as well as the equality for the z-components of $\boldsymbol{u}$ and $\boldsymbol{w}$, i.e.

$$\boldsymbol{u} = [\pm C_\alpha \;\; 0 \;\; \pm S_\alpha]^T; \quad v = [* \;\; * \;\; \pm S_\alpha]^T, \tag{33}$$

To ensure definite computing of q_1, it is additionally required that the *acos* argument in Eq. (26) belongs to the interval [-1; 1]. After appropriate rearranging, this condition can be presented as

$$|S_\alpha(C_\alpha u_x + S_\alpha u_z) - w_z| \le C_\alpha \sqrt{1 - (C_\alpha u_x + S_\alpha u_z)^2}$$

After denoting the angles between the vectors $\boldsymbol{u}$, $\boldsymbol{v}$ and the *Axis*$_1$, *Axis*$_2$ as μ, η

$$C_\alpha u_x + S_\alpha u_z = cos(\mu); \qquad w_z = cos(\eta) \tag{34}$$

and assuming that $(\mu, \eta) \in [0;\pi] \times '(0;\pi]$, this inequality can be rewritten as

$$sin(\alpha - \mu) \le cos(\mu) \le sin(\alpha + \mu) \tag{35}$$

that yields the following domain for (μ, η):

$$\begin{cases} \mu + \alpha - \pi/2 \le \eta \le \mu - \alpha + \pi/2 \\ -\mu - \alpha + \pi/2 \le \eta \le -\mu + \alpha + 3\pi/2 \end{cases} \tag{36}$$

So, the results for q_1 can be summarised as follows:

Proposition 2a. *For Inverse Problem* 2, *the values of* q_1 *can be computed definitely from expression* (26) *if and only if the angles* μ, η *between the positioner Axis*$_1$, *Axis*$_2$ *and the vectors* $\boldsymbol{u}$, $\boldsymbol{w}$, *respectively, satisfy inequalities* (36) and, *additionally*, $\mu \ne 0$ and $\mu \ne \pi$. *Otherwise, if* $(\mu, \eta) = (0, \pi/2-\alpha)$ *or* $(\mu, \eta) = (\pi, \pi/2+\alpha)$, *any value of* q_1 *satisfy the kinematic equation*.

As follows from the relevant analysis, the highest "*reachability*" in the positioner $\boldsymbol{u}$-space is achieved for $\eta \in [\pi/2-\alpha; \pi/2+\alpha]$. And, in contrast, for $\eta=0$ or $\eta=\pi$, the "workspace" is reduced to a single cone with the parameter $\mu=\pi/2-\alpha$ or $\mu=\pi/2+\alpha$.

In accordance with Eq. (26), computing of q_2 can fail only in the case of $w_x = w_y = 0$, i.e. for $\eta=0$ or $\eta=\pi$. Geometrically, it corresponds to the vector $\boldsymbol{w}$, which is normal to the positioner faceplate and, consequently, cannot be alternated by rotation around the *Axis*$_2$. So, the existence and uniqueness of solutions for q_2 are subject to the following proposition:

Proposition 2b. *For Inverse Problem* 2, *the value of* q_2 (*for given* q_1) *can be computed uniquely from expression* (27), *if and only if the angle* η *between the positioner Axis*$_2$ *and the vectors* $\boldsymbol{w}$ *satisfy the conditions* $\eta \ne 0$ and $\eta \ne \pi$. *Otherwise, if* $\eta=0$ or $\eta=\pi$, *any value of* q_2 *satisfies the kinematic equation* (*provided that the solution for* q_1 *exists*).

Therefore, for the second inverse problem, the singularity may exist for both axes, when $\boldsymbol{u}$ is parallel to *Axis*$_1$ or $\boldsymbol{w}$ is parallel to *Axis*$_2$.

5.4. Positioner Configurations

Similar to other manipulating systems, the positioner inverse kinematics is non-unique because of existence of two solution branches (see ± sign in Eqs. (20) and (26)). However, both the off-line programming and the real-time control require distinguishing between them to ensure continuity of the positioner motions. For this reason, the direct kinematics must yield an

additional output, *configuration index* $M=\pm 1$ describing positioner posture, which is also used as an additional input for the inverse transformation, to produce a unique result.
For inverse problem 1, the configuration index is defined trivially (see Eq. (20)), as the sign of the coordinate q_1 :

$$M_1 = sgn\left(q_1\right). \tag{37}$$

But for inverse problem 2, such index must identify the sign of the second term only (see Eq. (26)). So, it should be defined as

$$M_2 = sgn\left(q_1 - atan2\frac{u_y}{u_{xz}}\right). \tag{38}$$

From the geometrical point of view, the index M_2 indicates relative location of two planes, passing the $Axis_1$. The first of them is obtained by rotating of the X_0Z_0-plane around the $Axis_1$ by the angle q_1. And the second plane is passed via the $Axis_1$ and the vector $\boldsymbol{u}$. It should be also noted, that the index M_2 substantially differs from the traditional for robotics orientation index M_5=sgn(q_5), which describes the wrist configuration of the typical 6 d.o.f. manipulator.

5.5. Optimal Orienting of the Weld Joint

As adopted by practising engineers, the optimal weld orientation is achieved when the approaching vector is strictly vertical and consecutively, the weld direction vector lies in the horizontal plane, i.e. $(\theta, \xi)=(0, 0)$ and ${}^0\boldsymbol{s}_w= [0\ 0\ 1]$. Let us investigate this particular case in details.
For both the inverse problems, substation of the values (θ, ξ) and the vector ${}^0\boldsymbol{s}_w$ into Eqs. (20), (21) and (26), (28) yields the similar result:

$$q_1 = \pm\ \text{acos}\frac{s_w^z - S_\alpha^2}{C_\alpha^2}; \qquad q_2 = \text{atan2}\ \frac{S_1 \cdot s_w^x - S_\alpha V_1 \cdot s_w^y}{S_1 \cdot s_w^y + S_\alpha V_1 \cdot s_w^x}. \tag{39}$$

So, the condition of the solution existence (36) is reduced to

$$s_w^z \geq -C_{2\alpha} \quad \text{or} \quad \eta \leq \pi - 2\alpha \tag{40}$$

It means, that the "*working space*" of the positioner does not include the cone with the downward directed central axis and the aperture angle 4α. And, thereby, the corresponding welds can not be optimally oriented. But it can be proved that applying the first inverse problem with the input parameter

$$\xi' = max\{0;\ 2\alpha + \eta - \pi\}, \tag{41}$$

the orientation of such welds can be essentially improved and approached to the optimal one. The corresponding "suboptimal" solution is defined by the axis angles

$$q_1 = \pi; \quad q_2 = -a\,tan2\left(s_w^y / s_w^x\right) \tag{42}$$

i.e. exact equalities are achieved for the first and the second equations of system (19), while for the third one the residual is minimised only.

6. Welding Task Planning

6.1. Components of the Cycle Time

The overall cycle time of the robotic arc welding cell is basically determined by two major components: (i) the "arc time" which is actually spent for the welding and (ii) the "motion

time" required for the torch and workpiece repositioning between the welding operations. There are also several other time components related to the workpiece cooling and downloading/uploading, torch-tip cleaning, equipment adjustment, maintenance, etc., but these are beyond the scope of the welding process model considered here.
As stressed above, the arc time is minimal in the downhand case, and the torch speed should be reduced for the "out-of-position" welding. Since the downhand location corresponds to $\theta = \xi = 0$, we propose approximating the welding speed reduction by the following expression:

$$V(\theta,\xi) = \left(1 + k_v \;\| (\theta,\xi)^T \;\|\right)^{-1} \cdot V_o \tag{43}$$

where V_o is the downhand welding speed, k_v is the scaling factor, and $\|\cdot\|$ denotes the algebraic norm of the vector $(\theta, \xi)^T$. It is obvious that definition of this norm for a particular welding task is not trivial and must rely on the expert judgements, which assign the maximum allowable tolerances ($\theta_{\max}$, $\xi_{\max}$) and corresponding reduction factor $k_{\max}$. An example of such a definition is given in our paper (Dolgui and Pashkevich, 2005).
The robot and positioner motion times highly depend on the welding task sequencing, which prevents unreasonable movements of the torch and workpiece. For a single trajectory, the motion time depends on the control mode (simultaneous or sequential), travel distance, and velocity/acceleration limits incorporated in the path-planning algorithms. It can be proved that the minimum time for a single joint movement (for both the robot and positioner) is defined by the equation

$$\tau = \begin{cases} |\Delta q| \,/\, \dot{q}_{\max} + \dot{q}_{\max} / \ddot{q}_{\max}\,, & \textit{if} \quad |\Delta q| > \dot{q}^2_{\max} / \ddot{q}_{\max} \\ 2\sqrt{|\Delta q| \,/\, \ddot{q}_{\max}}\,, & \text{otherwise} \end{cases} \tag{44}$$

where Δq is the joint displacement, and $\dot{q}_{\max}$, $\ddot{q}_{\max}$ are the velocity/acceleration limits.
Then, the *robot* motion time for a single torch displacement (simultaneous axis control) is defined by the slowest axis drive

$$t_R = \max_i \{\tau_i\} \tag{45}$$

where i=1,...6 is the axis number. It should be noted that the latter expression is valid, if the path planning is performed in the manipulator joint-space. In the alternative case (the Cartesian-space path planning), the index variable must be extended to i=0,...6, where τ_0 is computed via the Cartesian displacement and relevant velocity/acceleration constraints in a similar way.
In contrast, the *positioner* motion time for a single workpiece reconfiguration (sequential axis control) is defined as the sum of the axis motion times, the pauses $\Delta\tau_i$ between the successive axis movements and also the auxiliary time τ_R

$$t_P = \sum_i \tau_i + \sum_i \Delta\tau_i + \tau_R\,, \tag{46}$$

where τ_R is required for the welding torch removing to the safe location, to avoid possible torch-workpiece collisions. As follows from this equation, its preferable to avoid achieving the downhand location for each weld individually, since it requires extra time for the torch removing that may overcompensate the downhand welding benefits.

6.2. Welding Task Sequencing

Because of its complexity, the general problem of the robotic *welding task planning* is usually broken in several hierarchical stages (Kim et *al.*, 1998). This decomposition implements a problem-specific mechanism, which makes its possible to reduce the size of the related combinatorial optimisation problems, while maintaining the productivity/quality compromise. These stages are defined as follows:

(i) The *weld seam clustering*, which arranges the welds into the groups that can be welded without changing the positioner configuration (in the downhand position preferably, within the allowable tolerances);

(ii) The *intra-cluster sequencing*, which determines the seam start/end points and the welding order for each cluster individually (minimising the robot motions subject to the heat-distortion-related constrains);

(iii) The *inter-cluster sequencing*, which determines the cluster processing order that minimises the positioner motions subject to the downhand-related constrains.

After completing these stages, each operation is further broken down into detailed robot-positioner movements, which finally yields the workcell control programs.

For the first stage, one can be applied the well-developed general clustering techniques (Everitt et al., 2001) that group together the welds with similar $\boldsymbol{s}_w$-vectors, which indicate the seam normal line directions relative to the workpiece base. However, taking into account the welding specificity, it is prudent to perform the clustering in terms of the θ-, ξ-angles introduced above. This poses the following optimisation problem for obtaining the positioner coordinates $\boldsymbol{q}_p$

$$\max_{i\in C_i}\left\{\|\,(\theta_i(\boldsymbol{q}_p),\,\xi_i(\boldsymbol{q}_p)\,)^T\,\|\right\}\to\min_{\boldsymbol{q}_p}, \tag{47}$$

which is solved within the usual clustering algorithms while evaluating the cluster diversity. In this expression, C_i denotes the *j*th cluster index set, $\|\,.\,\|$ is the norm of the vector $(\theta,\xi)^T$, and the functions $\theta_i(\boldsymbol{q}_p)$, $\xi_i(\boldsymbol{q}_p)$ define the *i*th weld orientation relative to gravity for the given positioner coordinate vector $\boldsymbol{q}_p$.. Obviously, in order to ensure the desired welding quality, it is necessary to constraint the inter-cluster diversity by assigning the upper bound $\Delta_{\max}$ for the above norm

$$\|\,(\theta(i,\boldsymbol{q}_p^i),\,\xi(i,\boldsymbol{q}_p^i)\,)^T\,\|\le\Delta_{\max}\,,\ \ \forall i\in C_j\,, \tag{48}$$

which is also easily incorporated in the existing clustering methods. An example of slightly different weld clustering techniques, based on the "representative weld-line" concept, is given in Kim et *al.* (1998).

For the second stage, there are already exist a number of problem-specific heuristics, neural network based methods, and genetic algorithms that allow generating the minim-time inter-cluster welding sequence. These take into account the heat-related distortions by assigning the minimum cooling time, size of the heat-affected zone, etc. A comprehensive review of recent advances in this area is given by Kim et *al.* (2005).

This section focuses on the third welding task planning stage, the inter-cluster sequencing, related to the optimisation of the positioner motions. To our knowledge, the only paper that addresses this problem directly is that of Kim et *al.* (1998) devoted to the welding operations planning in a robotic workcell with a rotating-tilting positioner. However, their approach assumes that each cluster is oriented using the representative weldline, which is transformed to the strictly downhand location by the positioner. Besides, after such orienting, the cluster welding time is assumed to be constant and computed directly from the welding speed and the weld line length.

In contrast to the known approach, the proposed technique admits the *out-of-position* (i.e. not exactly downhand) weld location, which is charged by the welding speed reduction. From this, one can pose a problem of minimum-time cluster sequencing by finding a o *trade-off* between the positioner motion time and the cluster processing time. Another useful feature of the proposed approach is the *re-clustering* ability. This means that the developed algorithm is able to find the same positioner configuration for several neighbouring clusters (i.e. to merge them), if the corresponding increase of the welding time is over-compensated by the reduction of the robot-positioner motion time. This allows to modify the clustering stage, which may impose very strong constraints on the inter-cluster similarity, and partly combine the clustering stage with the cluster sequencing one.

6.3. Associated optimisation problem

In a more formal way, the considered problem may be stated as follows. Let us assume the whole set of the welds is preliminary *clustered* in *n* groups, while the positioner joint coordinate space is *sampled* and presented by the uniform grid with the set of nodes $\{Q_p\}$. For each such node and each welding cluster, let us evaluate the orientation feasibility of the all the inter-cluster welds (within the given tolerance $\Delta_{\max}$) and compute corresponding processing times required for both the welding and the time-optimal torch movements between welds. In this step, the cluster welding time is adjusted in accordance with the allowable welding speed. Using the data obtained, let us create a cluster set $\{Q_p^{(1)}, Q_p^{(2)}, \cdots Q_p^{(n)}\}$ in the sampled positioner coordinate space, where each cluster $Q_p^{(i)}$ is composed of admissible nodes $\{Q_p^{(ik)} \mid k = \overline{1, m_i}\}$ with their processing time attributes $\{T_w(i,k) \mid k = \overline{1, m_i}\}$. Besides, let us assume that the positioner inter-node motion times $\{T_m(i_1, k_1, i_2, k_2) \mid k_i = \overline{1, m_i}\}$ are also computed, and is given the auxiliary time τ_R required for the torch moving before/after positioner re-configuration. Then, the minimum-time objective for the inter-cluster sequencing may be written as:

$$\sum_{i=1}^{n} T_w(i, k_i) + \sum_{(i_1, i_2) \in I} T_m(i_1, k_{i1}, i_2, k_{i2}) + \tau_R \cdot \sum_{i=1}^{n} 1(Q_p^{(i_1 k_{i1})} \neq Q_p^{(i_2 k_{i2})}) \to \min_{I,K} \tag{49}$$

where *I, K* denote the optimal cluster sequence and corresponding cluster node numbers, k_i defines the optimal node within the *i*th cluster, the first term accumulates the cluster-processing times (welding and torch travel times) , the second term represents the positioner motion times (between the clusters), and the third term takes into account the auxiliary robot motions between/after the positioner re-configurations via the indicator function 1(.).

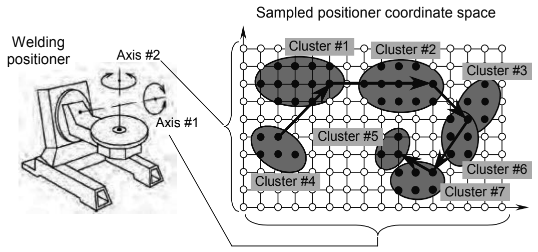


Fig 7. Representation of the inter-cluster sequencing problem in the positioner coordinate space.

A geometrical interpretation of this formulation is presented in Fig. 7 , where each welding cluster is defined by the set of feasible nodes in the positioner coordinate space. For each

node within each cluster, we assign the *node-processing time* that varies within the cluster, with the minimum located in the cluster centre. For each pair of nodes, there are also given the *inter-node travel times*. Besides, it is assumed that each non-zero travel requires some additional *pause-time*, needed for preparations of the safe movements. The goal is to find the minimum-time tour, which visits each cluster exactly ones, defined by both the cluster sequence and the set of visiting nodes within the clusters.
The hypothetical solution (see Fig. 7) shows a compromise between visiting cluster centres and their peripheries, since the node-processing times and the inter-node travel times compete inside the total sum to be minimised. Moreover, this solution utilises advantages of cluster overlapping by processing the clusters #3, #6 in the same positioner configuration (this removes the corresponding pause-time between successive positioner motions). Another interesting feature is the one-axis motion between the clusters #1 and #2, which is more time-economical than the two-axis motion requiring a pause between activating the axis drives.
It should be noted that this study focuses on generating the *non-closed* optimal tours for cluster visiting and processing, assuming that the workpiece downloading/uploading is performed at the tour start/end points interchangeably. However, the proposed technique is easily modified for the case of the closed tour, as well as for the case of a predetermined loading positioner configuration. Another essential specificity of the studied optimisation problem is related to cluster geometry, since the welding-task clusters are usually composed of several disjoint regions in the positioner coordinate space, corresponding to different inverse kinematic solutions. The next subsection proposes an optimisation algorithm that takes into account this problem specificity.

6.4. Optimisation Algorithm

As shown above, the considered problem of the inter-cluster operation planning can be converted into the *generalised TSP* (*GTSP*), in which the set of given nodes consists of several clusters (*overlapped,* in the general case) and the objective is to find the shortest route passing through each cluster exactly once. The GTSP was first mentioned in operation research literature in relations to computer files sequencing (Henry-Labordere, 1969). Further applications dealt with flexible manufacturing scheduling and material-flow systems design (Laporte and Palekar, 2002).
Since the GTSP is NP-hard and the exact algorithms usually are not able to obtain the optimal solution in a reasonable time, a variety of heuristics exist. The simplest ones are based on adaptation of the standard TSP techniques (such as nearest-neighbour, farthest-insertion, etc.). The most sophisticated to our knowledge GTSP heuristic, GI^3, was proposed by Renaud and Boctor (1998). However, it performs well for small cluster sizes only, and their simplest structure. Besides, the known technique employs some assumptions (Euclidian distances, for instance) that are not valid for the robotic welding application studied here.
To improve efficiency and computational speed in the case of the overlapping multi-region clusters and non-Euclidean distances, the the GI^3 was revised by simplifying the first two phases and replacing them with a straightforward generation of a random initial solution. Besides, the third phase is run in a slightly different manner, using two nested loops. The internal loop contains a classical TSP tour-improvement routine, which is repeated until no improvement is achieved by modifying the cluster sequence (for a fixed node subset). Then, within the external loop, the node-improvement routine is invoked to optimize the node subset, while the cluster sequence (tour) is considered not to vary. The external loop is also repeated until no improvement is achieved, converging to a local optimum. To increase the chances of attaining the global optimum, the algorithm is

repeated several times, each time starting from a new random initial solution and finishing by updating the best solution. The basic idea and outline of the proposed composite algorithm (called GI^{2+}) are given below:

```
Algorithm: Heuristic GI2+
(1) Generate random solution {Tour, Node}
(2) Set done:=0;
(3) Repeat while done=0
    (a) Repeat while done=0
        (i) Set done:=1;
        (ii) ImproveTour( );
            Set done:=0, if improved
    (b) ImproveNodes( );
            Set done:=0, if improved
(5) Update best solution and  repeat from Step 1, if
    desired.
```

In contrast to the original GI^3 technique, the proposed heuristic completely revises the subset after each tour alteration, and the tour improvement is run indefinite number of times, until no improvement is achieved (The GI^3 favours to the predetermined number of iterations for the tour improvement and for the reduced node improvement, executed for several neighbouring clusters only). The main reason for this amendment is the multi-region cluster geometry that disables the ideas implemented in the GI^3. Besides, the proposed algorithm structure (with two nested loops) simplifies the coding while keeping reasonable computational speed. The main procedures incorporated in the GI^{2+} algorithm (generation of an *initial solution*; *tour-improvement* and *subset-improvement*) are described in details in (Dolgui and Pashkevich, 2005).

6.5. Computational Results

The developed algorithm was tested on a number of randomly generated problems comprising from 50 to 1000 nodes and compared to the exact branch-and-bound technique. It was run on a PC with a 2.8 GHz Intel Pentium IV processor and 512 MB memory. To take into account specific properties of the considered robotic welding application, each cluster was build as a union of two disjoint circles, and the clusters were allowed to have essential intersection regions.

Problem Size			Reaching of Optimum (Heuristics/ Exact)				Relative Speed (Exact/Heuristics)		
n	*m*		Succ%	Mean	Max		Mean	Min	Max
5	10		87	1.005	1.202		2.73	1.82	3.87
	100		95	1.003	1.137		2.84	1.55	3.73
6	10		87	1.011	1.220		10.24	6.37	14.53
	100		89	1.003	1.166		11.20	6.92	16.70
7	10		55	1.013	1.182		41.03	22.96	56.05
	100		77	1.010	1.130		50.03	7.80	79.75

Table 1. Quality of the heuristic solutions (10 iterations)

The results presented in Table 1 indicate that problems involving up to 1000 nodes and 10 clusters were solved to optimality in 70-80% of cases, and an average solution is only 0.3…2.6 %

over the optimum, that is quite acceptable for engineering applications. In contrast, the exact algorithm (based on the branch-and-bound with the nested dynamic programming) can handle only low-dimensional problems (up to 5×100, 6×50, 7×20, 8×10). However, for smaller number of clusters ($n < 5$) the branch-and-bound method takes over the heuristic. To summarise the heuristic empirical performance, the running times were approximated by the expression $Time = c \cdot n^a m^b$ using the log-least-square technique that yielded $a \approx 1.44$ and $b \approx 1.92$. This shows that the heuristics remains rather moderate with respect to n, while for the exact algorithm the problem difficulty severely increases with the number of clusters.

7. Conclusion

Resent advances in arc welding technology motivate rethinking of some postulates and conventions incorporated in the existing robot control methods. This chapter addresses relaxing of the downhand-position assumption, which became a de-facto standard in robotic welding and requires the weld normal vector to be opposite to gravity. In contrast to the standard techniques, the developed method explicitly assumes that a weld may be processed in the non-downhand location, within given tolerances. But, to ensure the prescribed quality, the downhand deviation is charged by reduction of the welding speed. For such settings, it is proposed a novel method for the kinematic control of a robot-positioner system and related optimisation algorithm for the cluster-level planning of the welding operations. By using this technique in combination with the existing CAD tools, it is possible essentially reduce the cycle time of the robotic welding station.

8. References

Ahmad, S. & Luo, S. (1989). Coordinated motion control of multiple robotic devices for welding and redundancy coordination through constrained optimization in Cartesian space. *IEEE Transactions on Robotics and Automation*, Vol. 5, No 4, (Aug. 1989), 409–417.

Bolmsjo, G. (1987). A kinematic description of a positioner and its application in arc welding robots, In: *Proc. of the 2nd Int. Conference on Developments in Automated and Robotic Welding*, pp. 77-84., Nov. 1987, London, UK.

Bolmsjo, G.; Olsson, M. & Cederberg, P. (2002). Robotic arc welding—trends and developments for higher autonomy. *Industrial Robot: An International Journal*, Vol. 29, No 2, 98–104.

Cary, H.B. (1995). *Arc welding automation* , Marcel Dekker, New York.

Dolgui, A., & Pashkevich, A. (2006). Cluster-level operations planning for the out-of-position robotic arc welding. *International Journal of Production Research*, Vol. 44, No 4, (Febr. 2006), 675-702.

Everitt, B.S.; Landau, S. & Leese, M. (2001). *Cluster Analysis*, Oxford University Press, New York.

Fernandez, K. & Cook, G.E. (1988). A generalized method for automatic downhand and wire feed control of a welding robot and positioner, *NASA Technical Paper 2807*.

Fu, K. S.; Gonzalez, R. C. & Lee, C. S. G. (1987). *Robotics: Control, vision and intelligence*. McGraw-Hill, NewYork.

Fukuda, S. & Yoshikawa, K. (1990). Determination of welding sequence: a neural net approach, *Engineering Analysis with Boundary Elements*, Vol. 7, No 2, (June 1990), 78–82.

Grenestedt, J.L. (2003). Optimization of the weld path for overlay coatings. *Journal of Structural and Multi-Disciplinary Optimization*, Vol. 25, No 3, (Aug. 2003), 215–224.

Gutin, G. & Punnen, A.P. (Eds), (2002). *The Traveling Salesman Problem and its Variations*, Kluwer Academic Publishers, Dordrecht.

Henry-Labordere, A.L. (1969). The record balancing problem: a dynamic programming solution of a generalized traveling salesman problem. *Revue d'informatique et de recherche operationnelle* (*RIRO*), Vol. B-2, 43–49.

Kim, D.W.; Choi, J.-S. & Nnaji, B.O. (1989). Robot arc welding operations planning with a rotating/tilting positioner. *International Journal of Production Research*, Vol. 36, No 4, (Apr. 1998), 957–979.

Kim, H.J.; Kim, Y.D. & Lee, D.H. (2005). Scheduling for an arc welding robot considering heat-caused distortion. *Journal of the Operational Research Society*, Vol. 56, No 1, (Jan. 2006), 39 – 50.

Kim, K.Y.; Kim, D.W. & Nnaji, B.O. (2002a). Robot arc welding task sequencing using genetic algorithms. *IIE Transactions on Design and Manufacturing*, Vol. 34, No 10, (Oct. 2002), 865–880.

Kim, K.Y.; Norman, B. & Nnaji, B.O. (2002b). Heuristics for single-pass welding task sequencing. *International Journal of Production Research.*, Vol. 40, No 12, (Aug. 2002), 2769–2788.

Laporte, G. & Palekar, U. (2002). Some applications of the clustered travelling salesman problem. *Journal of the Operational Research Society*, Vol. 53, No9, (Sept. 2002), 972-976.

Nikoleris, G. (1990). A programming system for welding robots. *International Journal for the Joining of Materials*, Vol. 2, No 2, 55–61.

Pashkevich A. ; Dolgui A. & Semkin K.(2003). Kinematic aspects of a robot-positioner system in an arc welding application. *Control Engineering Practice*, Vol 11, No 6, (June 2003), 633–647.

Pires, J.N; Loureiro, A., Godinho, T., Ferreira, P., Fernando, B. & Morgado, J. (2003) Welding robots. *IEEE Robotics and Automation Magazine*, Vol. 10, No 2, (June 2003), 45- 55.

Ranky, P.G. (2004). Automotive robotics. *Industrial Robot: An International Journal;* Vol. 31, No 3, (June 2004), 252 – 257.

Renaud, J. & Boctor, F.F. (1998). An efficient composite heuristic for the symmetric generalized travelling salesman problem. *European Journal of Operational Research*, Vol. 108, No 3, (Aug. 1998), 571-584.

Rubinovitz, J. & Wysk, R.A. (1988). Task level off-line programming system for robotic arc welding – an overview. *Journal of Manufacturing Systems*, Vol. **7,** No 4, 293–299.

Tarn, T.J.; Chen, S.B. & Zhou, C. (Eds.), (2004). *Robotic Welding, Intelligence and Automation: Lecture Notes in Control and Information Sciences*, Vol. 299, Springer-Verlag, New York.

Tolinski, M. (2001). Getting to the Core of Welding Wire. *Forming and Fabricating*: *SME Journal*, Vol. 8, No 1, 42 –49.

Wu, L.; Cui, K. & Chen, S. B. (2000). Redundancy coordination of multiple robotic devices for welding through genetic algorithm. *Robotica*, Vol. 18, No 6, (Nov. 2000), 669 – 676.

Yagi, T. (2004). State-of-the-art welding and de-burring robots. *Industrial Robot: An International Journal*, Vol. 31, No 1, (Jan. 2004), 48-54.

16

Control and Scheduling Software for Flexible Manufacturing Cells

António Ferrolho[1], Manuel Crisóstomo[2]
[1]*Electrical Engineering Department*
Superior School of Technology of the Polytechnic Institute of Viseu
Campus Politécnico de Repeses, 3504-510 Viseu
PORTUGAL
E-mail: antferrolho@elect.estv.ipv.pt
[2]*Institute of Systems and Robotics*
University of Coimbra
Polo II, 3030-290 Coimbra
PORTUGAL
E-mail: mcris@isr.uc.pt

1. Introduction

One of the most recent developments in the area of industrial automation is the concept of the Flexible Manufacturing Cells (FMC). These are highly computerized systems composed by several types of equipment, usually connected through a Local Area Network (LAN) under some hierarchical Computer Integrated Manufacturing (CIM) structure (Kusiak, 1986) and (Waldner, 1992). FMC are capable of producing a broad variety of products and changing their characteristics quickly and frequently. This flexibility provides for more efficient use of resources, but makes the control of these systems more difficult. Developing an FMC is not an easy task. Usually, an FMC uses equipment (robots, CNC machines, ASRS, etc.) from different manufacturers, having their own programming languages and environments. CNC machines and industrial robots are still machines which are difficult to program, because it is necessary to be knowledgeable in several programming languages and environments. Each manufacturer has a particular programming language and environment (generally local). Robust and flexible control software is a necessity for developing an FMC.
We are going to present several software applications developed for industrial robots and CNC machines. The objective of this software is to allow these equipments to be integrated, in an easy and efficient way, in modern Flexible Manufacturing Systems (FMS) and FMC. For the industrial robots we are going to present the "winRS232ROBOTcontrol" and "winEthernetROBOTcontrol" software. For the CNC machines we are going to present the "winMILLcontrol", for mills, and "winTURNcontrol", for lathes. With this software, industrial robots and CNC machines can be integrated in modern FMC, in an easy and efficient way.
Genetic algorithms (GA) can provide good solutions for scheduling problems. In this chapter we also are going to present a GA to solve scheduling problems in FMC, which

are known to be *NP*-hard. First, we present a new concept of genetic operators for scheduling problems. Then, we present a developed software tool, called “HybFlexGA” (Hybrid and Flexible Genetic Algorithm), to examine the performance of various crossover and mutation operators by computing simulations of scheduling problems. Finally, the best genetic operators obtained from our computational tests are applied in the “HybFlexGA”. The computational results obtained with 40, 50 and 100 jobs show the good performance and the efficiency of the developed “HybFlexGA” to solving scheduling problems in FMC.
An FMC was developed, with industrial characteristics, with the objective of showing the potentialities of the developed software. The use of the “winRS232ROBOTcontrol”, “winEthernetROBOTcontrol”, “winMILLcontrol” and “winTURNcontrol” software allows the coordination, integration and control of FMC. With the “HybFlexGA” software we can solve scheduling problems in FMC.
With the work described in this chapter we can:

- Control and monitoring all of the functionalities of the robots and CNC machines remotely. This remote control and monitoring is very important to integrate robots and CNC machines in modern FMS and FMC.
- Develop a distributed software architecture based on personal computers using Ethernet networks.
- Develop applications that allow remote exploration of robots and CNC machines using always the same programming languages (e.g. C++).
- Use a personal computer as programming environment, taking advantage of the huge amount of programming and analysis tools available on these platforms.
- Develop Graphical User Interfaces (GUI) for robots and CNC machines, in a simple and efficient way.
- Write FMS and FMC control software easier, cheaper, and more flexible. As a result, the software expertise resides at the companies and software changes can be made by the company’s engineers. Furthermore, the developed software is easy to modify.
- Control and integrate several equipments from different manufactures in FMS and FMC (e.g. conveyors, motors, sensors, and so on).
- Data sharing. For example: share programmes, share databases and so on, among the client(s) and server(s).
- Solve scheduling problems in FMC with GA.

Suggestions for further research and development work will also be presented.

2. The State of the Art in FMS and FMC Control

An FMS or FMC consists of two major components: hardware and controlling software. The hardware, which includes computer controlled machines (or CNC machines), robots, storage and material handling systems, etc., has been around for decades and problems associated with the hardware have been well studied and are reasonably well understood (Sanjay et al., 1995). But, to write FMS and FMC control software is not a trivial task (Sanjay et al., 1995), (Johi et al., 1991), (Joshi et al., 1995) and (http#1). Nowadays, the software is typically custom written, very expensive, difficult to modify, and often the main source of inflexibility in FMC. Most FMS and FMC are sold to manufacturing

companies as turnkey systems by integration vendors. As a result, the software expertise does not reside at the companies and logic/software changes can only be made by the FMS and FMC vendor.
Several references on the use of formal models for automatic generation of software and compilers can be found in the computer literature. The use of formal models for control of flexible manufacturing cells has been discussed by (Sanjay et al., 1995). Other authors (Maimon & Fisher, 1988) have developed rapid software prototyping systems that provide only for the structural specifications of cell control software, still demanding enormous development of hand crafted code to realize effective systems. A detailed description of the implementation of a material handling system as a software/hardware component is given in (Hadj-Alonane et al., 1988).
Research at The Pennsylvania State University's Computer Integrated Manufacturing Laboratory (CIMLAB) has been focused on simulation based shop-floor control for more than a decade and a hierarchical control framework called "RapidCIM" has been developed. The "RapidCIM" architecture was initially designed for discrete manufacturing control, and has been implemented using a real-time simulation control framework. This architecture and its associated tools are capable of automatically generating much of the necessary software (up to 80 or 90% of a typical application) for automating discrete manufacturing systems. This substantially reduces the cost of developing and integrating of such systems, and allows a detailed simulation to be used for both analyses and control. Information about "RapidCIM" can be found in (Joshi et al., 1995), (http#1) and (http#2).

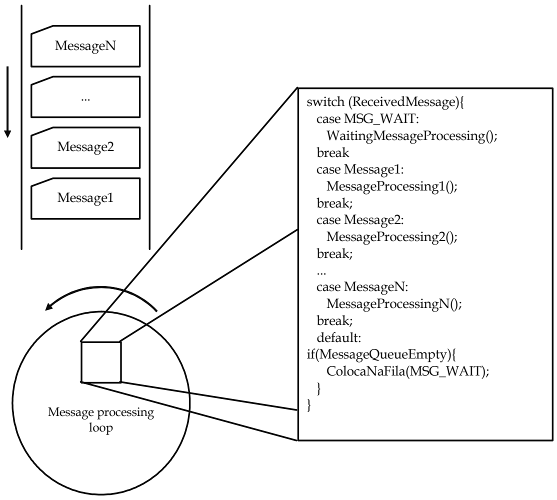
3. Industrial Robots Software

Nowadays, the robot controllers working in companies have the following important limitations:
- Some robot controllers only can use RS232 connections and others can support Ethernet networks.
- The robot manipulators have closed controllers and are essentially position-controlled devices that can receive a trajectory and run it continuously.
- Many robot controllers do not allow remote control from an external computer.
- Robot programming environments are not powerful to develop tasks requiring complex control techniques (e.g. integration and control of robots in FMC).
- Robot manipulators from different manufacturers have their own programming language and environments, making the task of integration and cooperation difficult.

To reduce several of these limitations we developed the "winRS232ROBOTcontrol" and the "winEthernetROBOTcontrol". With these developed software we can:
- Develop a distributed software architecture based on personal computers using Ethernet networks.
- Develop applications that allow remote exploration of robots manipulators using always the same programming languages (e.g. C++).
- Use a personal computer as programming environment, taking advantage of the huge amount of programming and analysis tools available on these platforms.
- Develop Graphical User Interface (GUI), in a simple and efficient way.

3.1 winRS232ROBOTcontrol

The "winRS232ROBOTcontrol" was developed to be used in robots manipulators where the communication with the controller is only possible through the RS232 serial port. With the "winRS232ROBOTcontrol" it is possible to develop robot programmes in C++, executed at the PC level, not at the robot's controller level. Nevertheless, the "winRS232ROBOTcontrol" also allows the development of programmes to be executed at a robot's controller level. Furthermore, it is also possible to have programmes running in the PC and in the robot's controller (mix solution), individually or simultaneously. The "winRS232ROBOTcontrol" was developed for industrial robot manipulators, and was implemented for the special case of Scorbot ER VII robot. Nevertheless, "winRS232ROBOTcontrol" was designed to work with all robots that support RS232. But, of course, whenever we use a different robot, it is necessary to make some upgrades in the RobotControlAPI library, according to the used robot.
The developed API, for the "winRS232ROBOTcontrol" application, is based on a thread process running simultaneously with the main programme. A thread process was created in order to poll the serial port. An event is generated each time data is presented in the serial port. In order to send data to the robot's controller, the message is placed in the serial port queue. Asynchronous processing will be used to send the message. For example, in order to open the robot's gripper, a data message is placed in the thread queue and further sent to the robot's controller trough the RS232 channel. After the delivery of the data, a waiting cycle is activated. This cycle is waiting for the robot's controller. It ends when the robot's controller sends back a prompt ('>'), a timeout error occurs or a cancellation message is sent. Fig. 1 shows the messages cycle of the thread process. The main programme communicates with the parallel process through the messages placed in the waiting queue, being this queue managed by the operating system. A message is immediately processed as soon as it arrives at the parallel process. In the case when there are no messages in the queue, the parallel process enters in a loop, waiting for data from the robot's controller. An event is generated when data arrive. The parallel process is activated in the beginning of the communication with the controller. When the communication with the robot's controller is finished, the resources allocated to the parallel process are returned to the operating system.



```
switch (ReceivedMessage){
  case MSG_WAIT:
    WaitingMessageProcessing();
  break
  case Message1:
    MessageProcessing1();
  break;
  case Message2:
    MessageProcessing2();
  break;
  ...
  case MessageN:
    MessageProcessingN();
  break;
  default:
if(MessageQueueEmpty){
    ColocaNaFila(MSG_WAIT);
  }
}
```

Fig. 1. Messages cycle of the parallel process.

The RobotControlAPI library was developed with access to the controller's functions in mind, as shown in Fig. 2. This API allows us to communicate with the robot's controller. The available functions are divided into two groups. The first contains public methods and the second contains methods representing events. Table 1 shows some of these functions.
We also have developed an Ethernet server for the computer that controls the robot. Thus, it is possible to control the robot remotely, in a distributed client/server environment, as shown in Fig. 2.

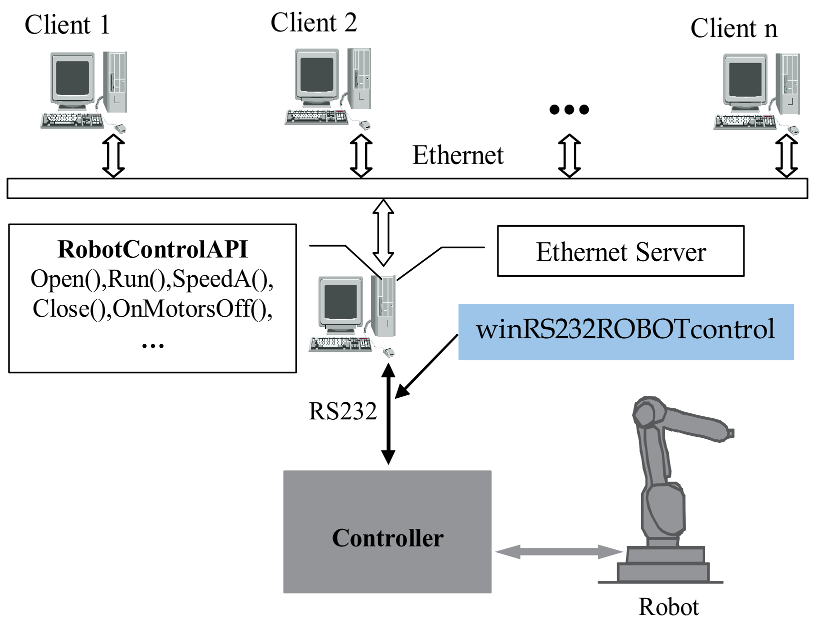


Fig. 2. RobotControlAPI library and Ethernet server.

	Function	Description
Public Methods	SpeedA()	Changes group A speed
	Home()	Homing the robot
	MotorsOff()	Switches off the robot's motors
	MotorsOn()	Switches on the robot's motors
	Close()	Closes the gripper
	Open()	Opens the gripper
Events	OnEndHoming()	This event activates after homing
	OnClose()	This event activates after the execution of the Close() function
	OnOpen()	This event activates after the execution of the Open() function
	OnMotorsOff()	This event activates after the motors are switched off
	OnMotorsOn()	This event activates after the motors are switched on

Table 1. Some available functions.

3.2 winEthernetROBOTcontrol

The "winEthernetROBOTcontrol" was developed to be used in robots manipulators where the communication with the controller is possible through an Ethernet network. The "winEthernetROBOTcontrol" is a Dynamic Link Library (DLL) and allows development of simple and fast applications for robots. The developed software uses the original capabilities of the robot's controllers, in a distributed client/server environment. Fig. 3 shows the PC1 and PC4 communicating with the robots through "winEthernetROBOTcontrol".

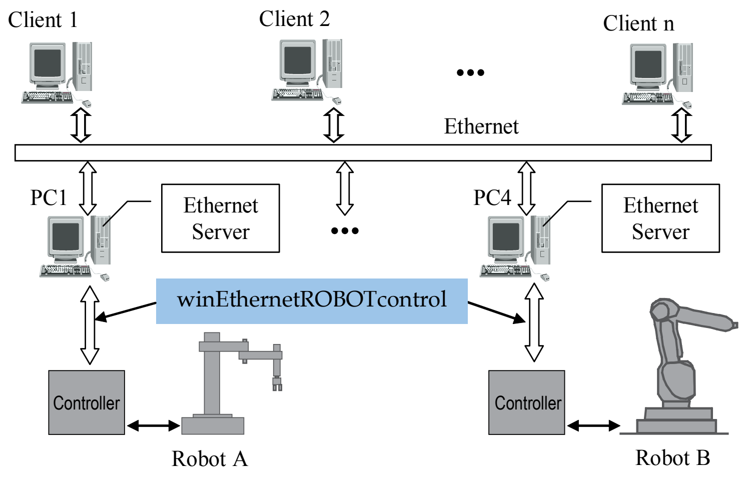


Fig. 3. winEthernetROBOTcontrol.

		Description
Procedures	Robot()	Add New Alias
	S4ProgamLoad()	Load a programme in the robot controller
	S4Run()	Switches ON the robot's motors
	S4Standby()	Switches OFF the robot's motors
	S4Start()	Start programme execution
	S4Stop()	Stop programme execution
	S4ProgramNumVarRead()	Reads a numerical variable in the robot controller
	S4ProgramNumVarWrite()	Writes a numerical variable in the robot controller
	S4ProgramStringVarWrite()	Writes a string in the robot controller
	S4ProgramStringVarRead()	Reads a string in the robot controller
Functions	Add_Variable()	Add a new variable
	Remove_Variable()	Remove one variable
	ControlID()	Gives the ID controller
	InerfaceState()	Gives the interface state.
Events	StatusChanged()	This event activates after something changed in the robot controller. For example: power off, emergency stop, executing sate, stopped state, run, and so on.
	BoolVariableChanged()	This event activates after one Boolean variable changed
	NumVariableChanged()	This event activates after one numeric variable changed

Table 2. Some available procedures, functions and events.

Table 2 presents same procedures, functions and events developed for the "winEthernetROBOTcontrol". The procedure "Robot" allows making a connection between one PC (with winEthernetROBOTcontrol) and one or more robots. For that, it is necessary to create and configure one connection between the PC and the robot controller (Add New Alias). After that it is possible to communicate and control the robot remotely through the connection (Alias). With "winEthernetROBOTcontrol" we can develop a software control, for one or more robots, in any PC. With this PC we can communicate and control the robot (or robots) through an Ethernet (see Fig. 3).
The "winEthernetROBOTcontrol" library was developed for industrial robot manipulators, and was implemented for the special case of ABB robots (with the new S4 control system). Nevertheless, this very flexible library was designed to work with all robots that support Ethernet. But, of course, if we use a different robot it is necessary to make some upgrades in this library, according to the used robot. The basic idea is to define a library of functions for any specific equipment that in each moment define the basic functionality of that equipment for remote use.

4. Industrials CNC Machines Software

Nowadays, all the manufactured CNC machines support Ethernet networks. But, these actual CNC machines have the following important limitations:

- The actual CNC machines have closed numerical control (NC) and are developed to work individually, not integrated with others equipments (e.g. others machines, robots, and so on).
- Actual CNC machine's controllers do not allow remote control from an external computer.
- CNC machine's programming environments are not powerful enough to develop tasks requiring complex control techniques (e.g. integration and control CNC machines in FMC).
- CNC machines from different manufacturers have their own programming language and environments, making the task of integration and cooperation difficult.

To reduce these limitations, we developed two programmes in C++ that allow controlling the CNC machines through Direct Numerical Control (DNC). For lathes, the "winTURNcontrol" programme was developed, and "winMILLcontrol" for mills. With the proposed software we can reduce several limitations in CNC machines:

- We can control and monitor all of the functionalities of the CNC machines remotely (e.g. open and close door, open and close chuck, open and close vice, start machine, machine ready, and so on). This control and monitoring remotely is very important to integrate CNC machines in modern FMS and FMC.
- We can develop a distributed software architecture based on personal computers using Ethernet networks.
- We can develop applications that allow remote exploration of CNC machines using always the same programming languages (e.g. C++).
- We can use a personal computer as programming environment, taking advantage of the huge amount of programming and analysis tools available on these platforms.

- We can develop Graphical User Interface (GUI) for CNC machines, in a simple and efficient way.

Some CNC machine's controllers (e.g. FANUC) only allow external connections through the RS232 serial port (Ferrolho et al., 2005). In order to surpass this limitation, a "Null Modem" cable was connected between COM1 and COM2 serial port of each PC that controls the respective machine, as shown in Fig. 4. The developed DNC programmes use the COM1 to send orders (e.g. open door, close vice, start NC programme, and so on) to the machine and the CNC machine's controllers receive the same orders by COM2. For that, we need configure the CNC machines software to receive the DNC orders by COM2. We have developed an Ethernet server for the PCs that control the machines, so as to allow remote control of the machines. Thus, it is possible to control the machines remotely, as shown in Fig. 4. Each machine's server receives the information through the Ethernet and then sends it to the machine's controller through the serial port. For example, if the client 1, in Fig. 4, needs to send an order (e.g. close door) to the machine A, the developed programme running in the machine computer receives the order through the Ethernet and sends the same order to the COM1 port. After that, the machine's controller receives this order through the COM2 (null modem cable) and then the machine closes the door.
DNC is an interface that allows a computer to control and monitor one or more CNC machines. The DNC interface creates a connection between client (e.g. client 1) and the CNC machine computer. After activating the DNC mode, client starts to control the CNC machines through the Ethernet, as shown in Fig. 4. The developed DNC interface establishes a TCP/IP connection between computer client and the CNC machine's computer. With the DNC interface developed it is possible to perform the following remotely: transfer NC programmes (files of NC codes) to the CNC machines, work with different parts' NC files, visualize alarms and information regarding the state of the CNC machines, control all the devices of the CNC machines (e.g. doors, clamping devices, blow-out, refrigeration system, position tools, auxiliary drives, and so on).

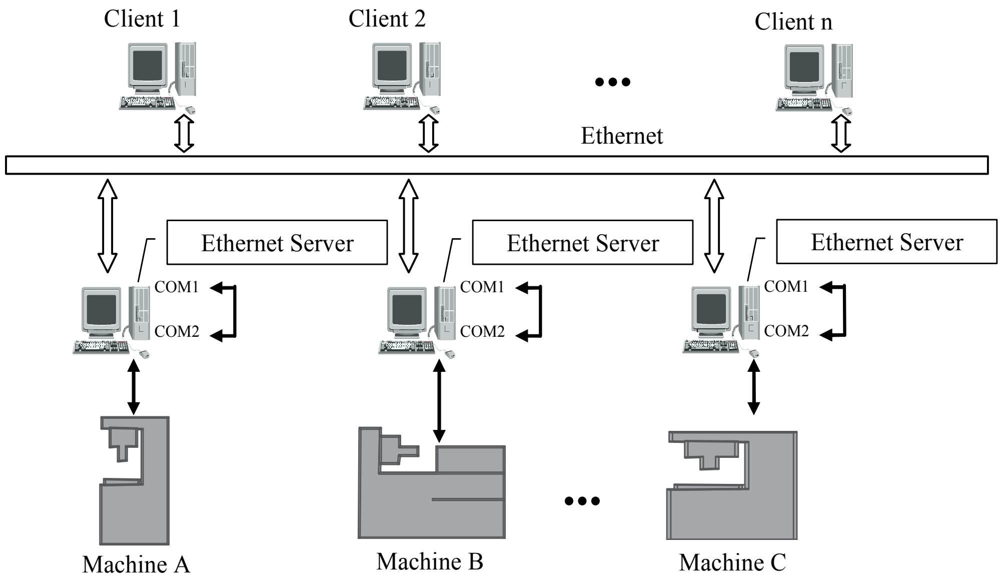


Fig. 4. Control CNC machines through Ethernet.

5. USB software and USB Kit

We have developed hardware (USB Kit) and software (USB software) for control and integration of several equipments in FMS and FMC (e.g. conveyors, motors, sensors, actuators, and so on). The USB Kit has various inputs and outputs where we can connect various sensors and actuators from different manufacturers. The communication between the PC1 and USB Kit is done by USB port, as shown in Fig. 5. The PC1 communicates with others PCs through Ethernet.

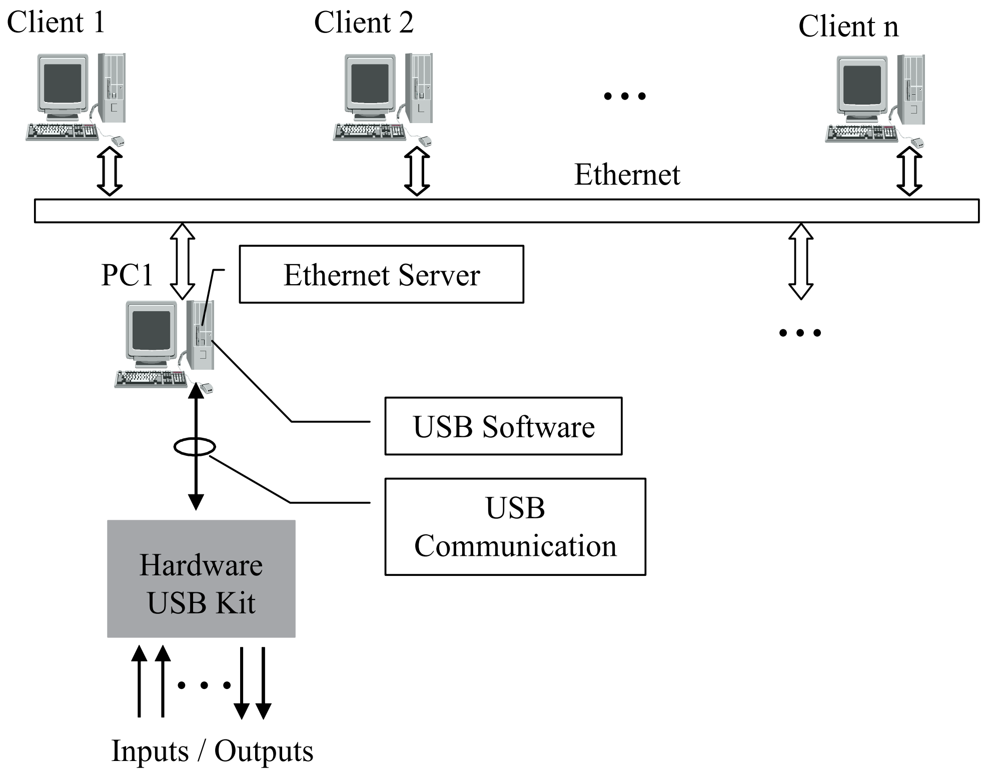


Fig. 5. USB software and USB kit.

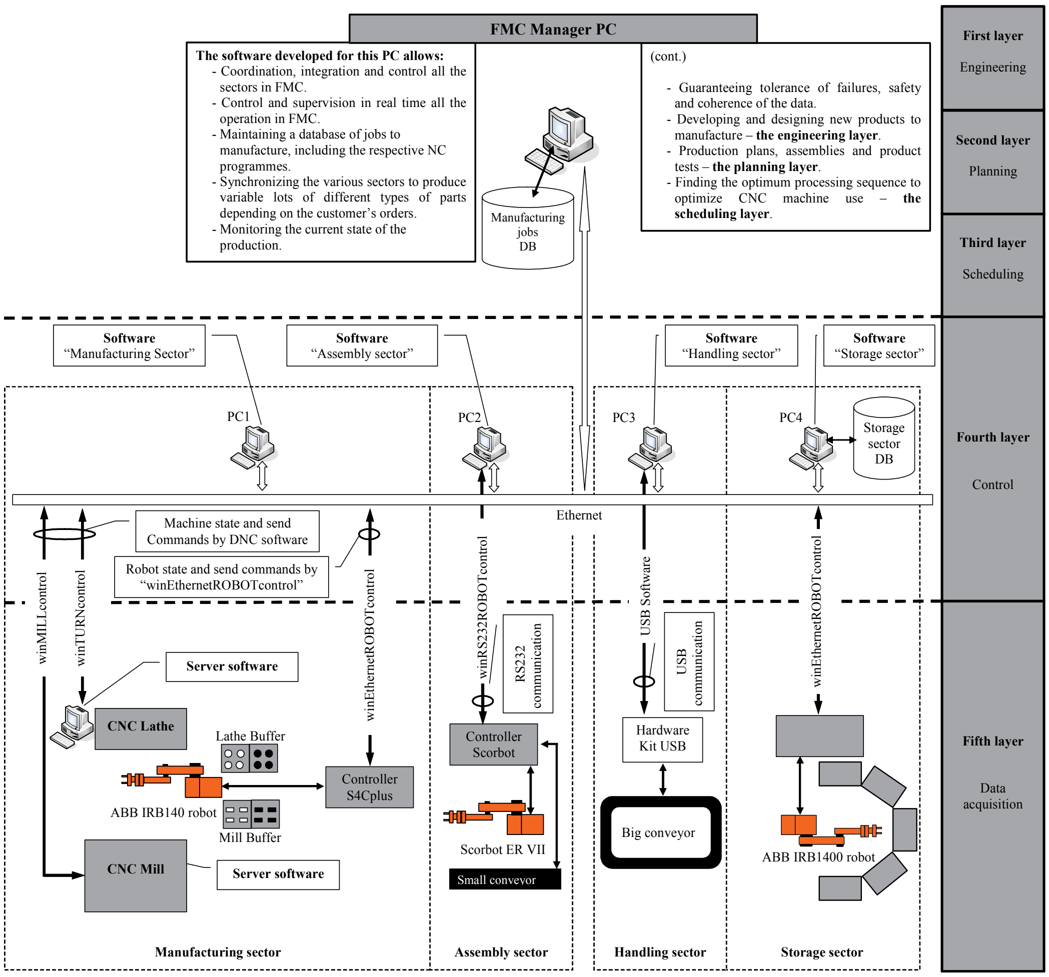
6. Developed Flexible Manufacturing Cell

The developed FMC is comprised of four sectors and the control of existing equipment in each sector is carried out by four computers: PC1 – manufacturing sector, PC2 – assembly sector, PC3 – handling sector and PC4 – storage sector. The coordination, synchronization and integration of the four sectors is carried out by the FMC manager computer. The four sectors are (Ferrolho & Crisóstomo, 2005-a) and (Ferrolho & Crisóstomo, 2005-b):

– The manufacturing sector, made up of two CNC machines (mill and lathe), one ABB IRB140 robot and one buffer.
– The assembly sector, made up of one Scorbot ER VII robot, one small conveyor and an assembly table.

- The handling sector, made up of one big conveyor.
- The storage sector, made up of five warehouses and one robot ABB IRB1400.

The hierarchical structure implemented in the FMC is shown in Fig. 6.
In the FMC three communication types were used: Ethernet in the manufacturing sector and storage sector, RS232 in the assembly sector and USB in the handling sector. PC1, PC2, PC3, PC4 and FMC manager PC are connected via Ethernet as shown in Fig. 6. If necessary it is possible to develop new applications using other communication types.


FMC Manager PC
The software developed for this PC allows:
- Coordination, integration and control all the sectors in FMC.
- Control and supervision in real time all the operation in FMC.
- Maintaining a database of jobs to manufacture, including the respective NC programmes.
- Synchronizing the various sectors to produce variable lots of different types of parts depending on the customer's orders.
- Monitoring the current state of the production.
Manufacturing jobs DB
(cont.)
- Guaranteeing tolerance of failures, safety and coherence of the data.
- Developing and designing new products to manufacture – the engineering layer.
- Production plans, assemblies and product tests – the planning layer.
- Finding the optimum processing sequence to optimize CNC machine use – the scheduling layer.
First layer
Engineering
Second layer
Planning
Third layer
Scheduling
Fourth layer
Control
Fifth layer
Data acquisition
Software "Manufacturing Sector"
Software "Assembly sector"
Software "Handling sector"
Software "Storage sector"
PC1
PC2
PC3
PC4
Storage sector DB
Ethernet
Machine state and send Commands by DNC software
Robot state and send commands by "winEthernetROBOTcontrol"
winMILLcontrol
winTURNcontrol
winEthernetROBOTcontrol
winRS232ROBOTcontrol
RS232 communication
USB Software
USB communication
winEthernetROBOTcontrol
Server software
CNC Lathe
Lathe Buffer
ABB IRB140 robot
Mill Buffer
Controller S4Cplus
CNC Mill
Server software
Controller Scorbot
Scorbot ER VII
Small conveyor
Hardware Kit USB
Big conveyor
ABB IRB1400 robot
Manufacturing sector
Assembly sector
Handling sector
Storage sector
Fig. 6. Hierarchical structure of FMC.

The layered level of the proposed architecture has inherent benefits. This hierarchical structure allows adding new sectors in the FMC, in an easy and efficient way. The new sectors can also use several equipments from different manufacturers. For this, we only need to add a computer in the fourth layer and to develop software to control the several equipments in those sectors.

Manufacturing Sector

Control of the manufacturing sector is carried out by the PC1, as shown in Fig. 6. The manufacturing sector is controlled by software, developed in C++, and has the following functions: buffer administration, CNC machine administration and ABB IRB140 robot control. Control of the IRB140 robot is carried out by "winEthernetROBOTcontrol". Control of the Mill and Lathe is carried out by DNC software – "winMILLcontrol" and "winTURNcontrol" (Ferrolho et al., 2005). This software has a server which allows remote control of the manufacturing sector.

Assembly Sector

The assembly sector is made up of a Scorbot ER VII robot, a conveyor and an assembly table, as shown in Fig. 6. Responsibility for the control and coordination of this sector falls on PC2 and in this sector we used the "winRS232ROBOTcontrol".

Handling Sector

Control of the handling sector is carried out through the PC3 and of a USB kit (hardware), as shown in Fig. 6. The USB kit acquires the signals coming from the sensors and sends them to PC3. When it is necessary to activate the stoppers, PC3 sends the control signals to the USB kit.

Storage Sector

Control of the storage sector is carried out by PC4, as shown in Fig. 6. The functions of the developed software are the administration of the database of the whole warehouse, and controlling the ABB IRB1400 robot. The database can be updated at any moment by the operator, but it is also automatically updated whenever the ABB IRB1400 robot conducts a load operation, unloads or replaces pallets. Control of the ABB 1400 robot is carried out by "winEthernetROBOTcontrol" functions, procedures and events.

FMC Manager PC

The central computer (FMC manager PC) controls all of the FMC production, connecting the various computers and data communication networks, which allows real time control and supervision of the operations, collecting and processing information flow from the various resources. In this PC the first three layers of the hierarchical structure presented in Fig. 6 are implemented: engineering, planning and scheduling.

The first layer contains the engineering and product design, where the product is designed and developed with CAD/CAM software (e.g. MasterCam). The outputs of this layer are the drawings and the bill of materials.

The second layer is process planning. Here the process plans for manufacturing, assembling and testing are made. The outputs of this layer are the NC code for the CNC machines. We can obtain these NC code also with CAD/CAM software. After that we put the product design, process plan, NC code, and so on, of each job in the products database. Whenever we have a new product (job) it is necessary to put all the information about this job in the products database.

The third layer is scheduling. The process plans together with the drawing, the bill of materials and the customer orders are the input to scheduling. The output of scheduling is the release of the order to the manufacturing floor. We used Genetic

Algorithms (GA) to solve scheduling problems in the FMC (Ferrolho & Crisóstomo, 2005-c). Some scheduling problems are very difficult to solve, but if the GA has been developed correctly, we can obtain good solutions, maybe even the optimal schedule. The scheduling problems studied and implemented in FMC were: single machine scheduling problem, flow-shop scheduling problem and job-shop scheduling problem.

Concisely, the FMC manager PC is responsible for:

- Developing and designing new products to manufacture – the engineering layer.
- Production plans, assemblies and product tests – the planning layer.
- Finding the optimum processing sequence so as to optimize CNC machine use – the scheduling layer.
- Maintaining a database of jobs to manufacture, including the respective NC programmes.
- Synchronizing the various sectors so as to produce variable lots of different types of parts depending on the customer's orders.
- Monitoring the current state of the production.
- Guaranteeing tolerance of failures, safety and coherence of the data.

7. Manufacturing Scheduling – Literature Review

Assuming we know the manufacturing products, their manufacturing schedule and the available resources for execution, the scheduling task consists in determining the schedule passage of the products (jobs) for the resources and defining the attribution of resources, considering the times at the beginning and at the end of operations, with the objective of optimizing one or more performance measures. This description is particularized for the case of the scheduling problems in production systems, as is the case of the FMC. In fact, scheduling problems appear associated to very different areas of activity, such as arranging school schedules, defining the order in which airplanes land at an airport, choosing the order of execution of different programs in a computer, and so on.

Supposed we have n jobs $\{J_1,...,J_n\}$ to be processed in m machines $\{M_1,... ,M_m\}$. Possible solutions are determined by $(n!)^m$ (French, 1982). The processing job in a machine is designated by operation and characterized by the respective processing time p_{ij} (job i, machine j). For each job technological restrictions are defined, that are the necessary operations sequence of job processing. Thus, the scheduling problem consists in determining a sequence of passage of jobs to the respective machines so that it is: 1) compatible with technological restrictions and 2) optimal, according to performance measures. Table 3 and Fig. 7 clarify the notation used.

We classify scheduling problems according to four parameters: $n/m/A/B$. The parameter n is the number of jobs, m is the number of machines, A describes the flow pattern (F for the flow-shop case and G for the general job-shop case), B describes the performance measure by which the schedule is to be evaluated. When m=1, A is left blank.

Scheduling problems for 3 jobs and m machines up to n jobs and m machines have proven very difficult to solve to completion. Heuristics have been developed to deal with these kinds of problems. Unfortunately, these are the predominant problems in industry and therefore computationally efficient techniques are very important for their solution. Some of the problems are made even more difficult when the rate of demand for products varies. This dynamic element violates some of the convenient assumptions about approaches used in static cases (Rembold et al., 1993).

Description	Symbol	Remarks
Number of jobs	n	
Job i	J_i	
Machine j	M_j	
Operation time	O_{ij}	
Processing time	p_{ij}	
Read time for J_i	r_i	
Due date for J_i	d_i	
Allowance for J_i	a_i	$a_i=d_i-r_i$
Waiting time of J_i preceding the respective k th operation	W_{ik}	
Total waiting time of J_i	W_i	$W_i = \sum_{k=1}^{m} W_{ik}$
Completion time of J_i	C_i	$C_i = r_i + \sum_{k=1}^{m} \left[W_{ik} + p_{ij(k)}\right]$
Flow time of J_i	F_i	$F_i=C_i-r_i$
Lateness of J_i	L_i	$L_i=C_i-d_i$
Tardiness of J_i	T_i	$T_i=\max\{0, L_i\}$
Earliness of J_i	E_i	$E_i=\max\{-L_i, 0\}$
Idle time on machine M_j	I_j	$I_j = C_{\max} - \sum_{i=1}^{n} p_{ij}$

Table 3. Scheduling notation.

As stated, the general n jobs and m machines job-shop scheduling problem has proved to be very difficult. The difficult is not in modeling but in computation since there is a total of $(n!)^m$ possible schedules, and each one must be examined to select the one that gives the minimum makespan or optimizes whatever measure of effectiveness is chosen. There are no known efficient, exact solution procedures. Some authors have formulated integer programming models but the computational results (computational complexity) have not been encouraging. For small problems, the branch and bound technique has been shown to be very good and better than integer programming formulation, but this has been computationally prohibitive for large problems.
In the next sections we are going to present a GA to solve scheduling jobs in FMC. First, we present a new concept of genetic operators. Then, we present a developed software tool "HybFlexGA". Finally, we show the good performance and the efficiency of the developed "HybFlexGA" to solving scheduling problems in FMC.

8. Genetic Algorithms

The strong performance of the GA depends on the choice of good genetic operators. Therefore, the selection of appropriate genetic operators is very important for

constructing a high performance GA. Various crossover and mutation operators have been examined for sequencing problems in the literature (for example, see (Goldberg, 1989), (Oliver et al., 1987), (Syswerda, 1991), (Ferrolho & Crisóstomo, 2005-d) and (Murata & Ishibuchi, 1994)). When the performance of a crossover operator is evaluated, a GA without mutation is employed and the evaluation of a mutation operator is carried out by a GA without crossover. In the literature, various crossover operators and mutation operators were examined in this manner (for example, see (Murata & Ishibuchi, 1994)).
In this section, we propose a new concept of genetic operators for scheduling problems. We evaluate each of various genetic operators with the objective of selecting the best performance crossover and mutation operators.

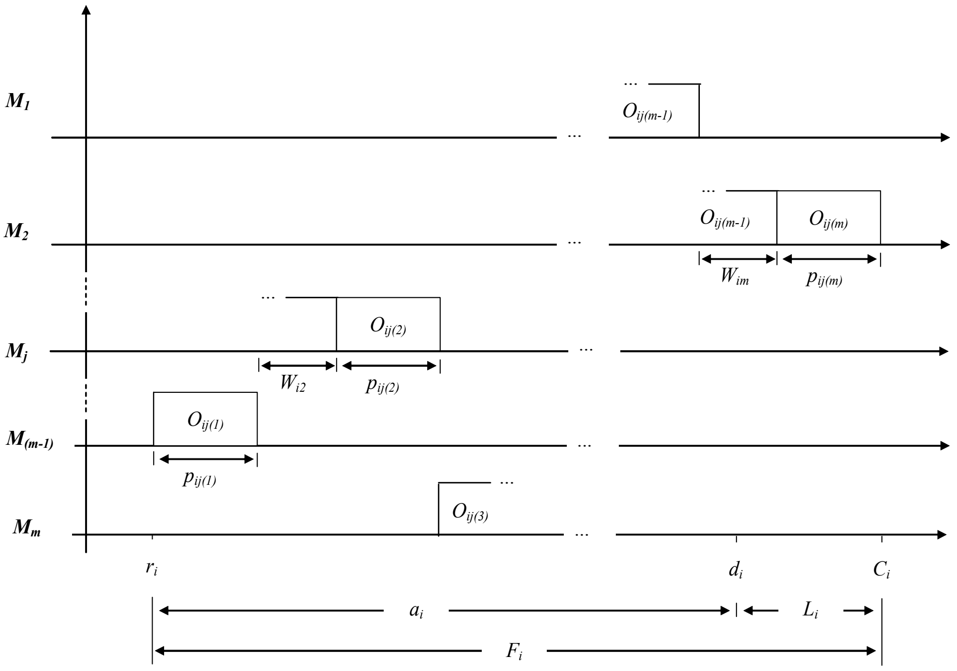


Fig. 7. Gantt diagram of a typical job J_i.

8.1 Crossover Operators

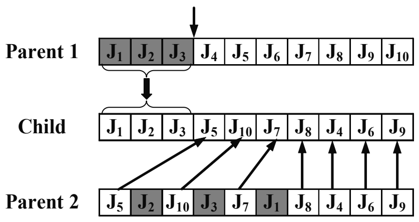
Crossover is an operation to generate a new sequence (child chromosome) from two sequences (parent chromosomes). We developed the following crossover operators:

- One-point crossover: 1 child (OPC1C) in Fig. 8 a).
- Two-point crossover: 1 child (Version I) (TPC1CV1) in Fig. 8 b).
- Two-point crossover: 1 child (Version II) (TPC1CV2) in Fig. 8 c).
- One-point crossover: 2 children (OPC2C) in Fig. 8 d).
- Two-point crossover: 2 children (Version I) (TPC2CV1) in Fig. 8 e).
- Two-point crossover: 2 children (Version II) (TPC2CV2) in Fig. 8 f).

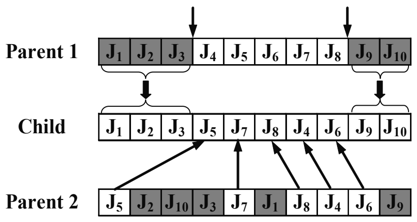
We also developed crossover operators with 3 and 4 children. The crossover operators with 3 children are called two-point crossover: 3 children (Version I) (TPC3CV1) and two-point crossover: 3 children (Version II) (TPC3CV2). TPC3CV1 is a mix of TPC1CV1 plus TPC2CV1 and TPC3CV2 is a mix of TPC1CV2 plus TPC2CV2. The crossover operator with 4 children is called two-point crossover: 4 children (TPC4C). This operator is a mix of TPC2CV1 plus TPC2CV2.

In our computational tests in section 4 we also used the following crossover operators:

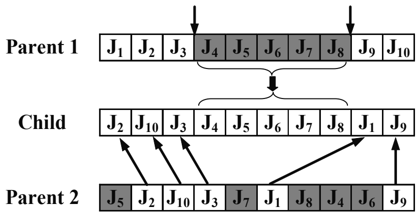
- Order crossover (OX) in Goldberg (Goldberg, 1989);
- Cycle crossover (CX) in Oliver (Oliver et al., 1987);
- Position based crossover (PBX) in Syswerda (Syswerda, 1991).



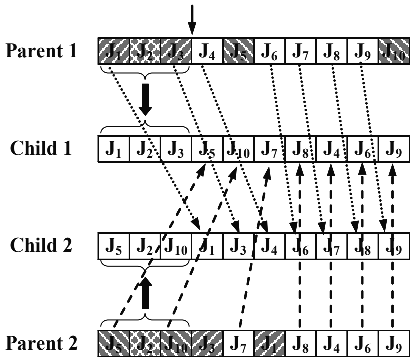
(a) One-point crossover: 1 child



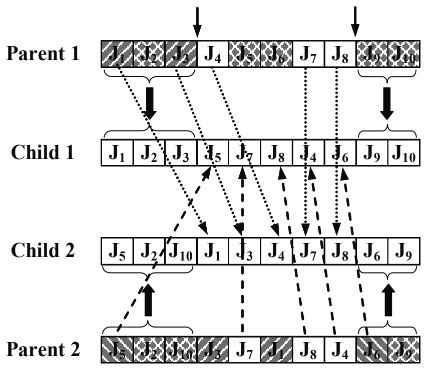
(b) Two-point crossover: 1 child (Vers. I)



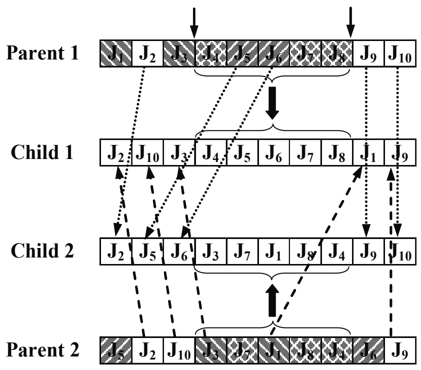
(c) Two-point crossover: 1 child (Vers. II)



(d) One-point crossover: 2 children



(e) Two-point crossover: 2 children (Vers. I)



(f) Two-point crossover: 2 children (Vers. II)

Fig. 8. Illustration of crossover operators.

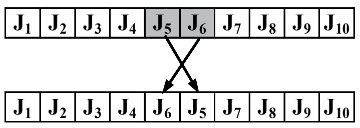
8.2 Mutation Operators

Mutation is an operation to change the order of *n* jobs in the generated child. Mutation operators are used to prevent the loss of genetic diversity. We examined the following four mutations used by Murata in (Murata & Ishibuchi, 1994).

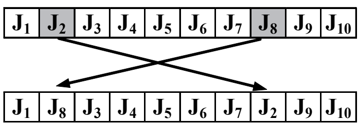
- Adjacent two-job change (Adj2JC) in Fig. 9 a).
- Arbitrary two-job change (Arb2JC) in Fig. 9 b).
- Arbitrary three-job change (Arb3JC) in Fig. 9 c).
- Shift change (SC) in Fig. 9 d).

As we can see in Fig. 9 a), b) and c) the jobs to be changed are arbitrarily and randomly selected. In Fig. 9 d) a job at one position is removed and placed at another position.

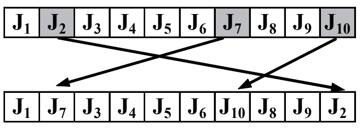
We developed a new mutation operator called the arbitrary 20%-job change (Arb20%JC), as we can see in Fig. 10. This mutation selects 20% of the jobs in the child chromosome. The 20% of the jobs to be changed are arbitrarily and randomly selected, and the order of the selected jobs after the mutation is randomly specified. The percentage in this mutation operator gives the operator some flexibility, i.e., the number of jobs to be changed depends on the size of the chromosome. For example, if we have a chromosome with 20 jobs (see Fig. 10) and another with 100 jobs, the number of jobs to be changed is 4 and 20 respectively.


J1 J2 J3 J4 J5 J6 J7 J8 J9 J10
J1 J2 J3 J4 J6 J5 J7 J8 J9 J10

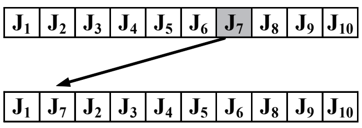
(a) Adjacent two-job change


J1 J2 J3 J4 J5 J6 J7 J8 J9 J10
J1 J8 J3 J4 J5 J6 J7 J2 J9 J10

(b) Arbitrary two-job change


J1 J2 J3 J4 J5 J6 J7 J8 J9 J10
J1 J7 J3 J4 J5 J6 J10 J8 J9 J2

(c) Arbitrary three-job change


J1 J2 J3 J4 J5 J6 J7 J8 J9 J10
J1 J7 J2 J3 J4 J5 J6 J8 J9 J10

(d) Shift change

Fig. 9. Illustration of mutation operators.

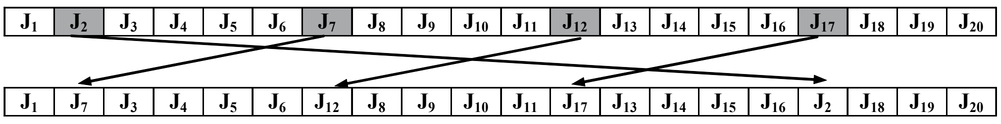

J1 J2 J3 J4 J5 J6 J7 J8 J9 J10 J11 J12 J13 J14 J15 J16 J17 J18 J19 J20
J1 J7 J3 J4 J5 J6 J12 J8 J9 J10 J11 J17 J13 J14 J15 J16 J2 J18 J19 J20

Fig. 10. Arbitrary 20%-job change.

9. Scheduling software for Flexible Manufacturing Cells

We developed a software tool, called HybFlexGA (Hybrid and Flexible Genetic Algorithm), to examine the performance of various crossover and mutation operators by computing simulations on scheduling problems. The HybFlexGA was coded in C++ language and Fig. 11 shows its architecture. Its architecture is composed of three modules: interface, pre-processing and scheduling module.

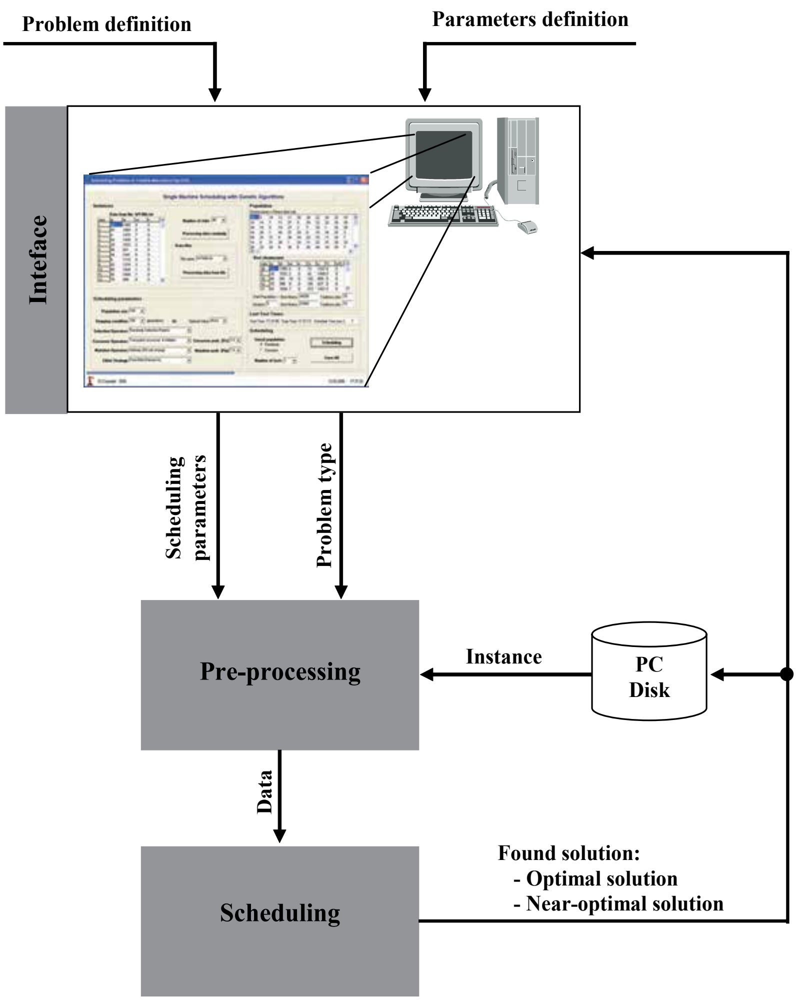


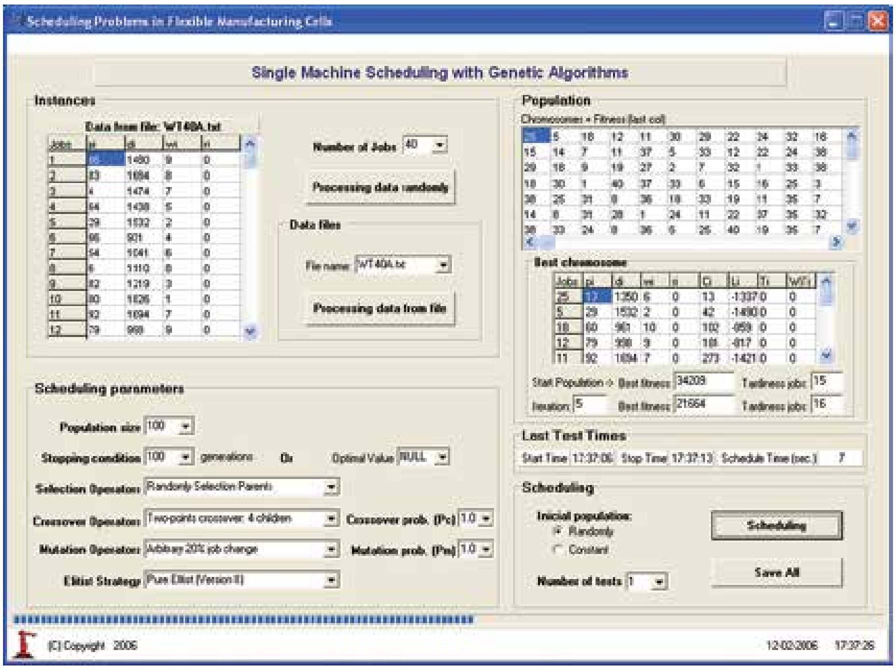
Fig. 11. Architecture of the HybFlexGA.

9.1 Interface Module

The interface module with the user is very important for the scheduling system's success. Thus, this interface should be user-friendly and dynamic so as to allow easy manipulation of the scheduling plan, jobs, and so forth. This interface allows the connection between the user and the scheduling module, facilitating data entry (for example, parameter definition and problem definition) and the visualization of the solutions for the scheduling module. Fig. 12 shows the interface window.

9.2 Pre-processing Module

The inputs of the pre-processing module are the problem type and the scheduling parameters. The instance of the scheduling problem can be randomly generated or generated by PC file, as shown in Fig. 12. This module, pre-processes the input information and then sends the data to the next module – the scheduling module.


Scheduling Problems in Flexible Manufacturing Cells

Single Machine Scheduling with Genetic Algorithms

Instances

Data from file: WT40A.txt

Jobs	pi	di	wi	ri
1	[illegible]	1480	9	0
2	83	1694	8	0
3	4	1474	7	0
4	64	1438	5	0
5	29	1532	2	0
6	96	901	4	0
7	54	1041	6	0
8	6	1110	8	0
9	82	1219	3	0
10	80	1626	1	0
11	92	1694	7	0
12	79	998	9	0

Number of Jobs: 40

Data files

File name: WT40A.txt

Population

Chromosomes + Fitness (last col)

25	5	18	12	11	30	29	22	24	32	18
15	14	7	11	37	5	33	12	22	24	38
29	18	9	19	27	2	7	32	1	33	38
10	30	1	40	37	33	6	15	16	25	3
38	25	31	8	36	18	33	19	11	35	7
14	8	31	28	1	24	11	22	37	35	32
38	33	24	8	36	5	25	40	19	35	7

Best chromosome

Jobs	pi	di	wi	ri	Ci	Li	Ti	WiTi
25	17	1350	6	0	13	-1337	0	0
5	29	1532	2	0	42	-1490	0	0
18	60	961	10	0	102	-859	0	0
12	79	998	9	0	181	-817	0	0
11	92	1694	7	0	273	-1421	0	0

Start Population -> Best fitness: 34209 Tardiness jobs: 15

Iteration: 5 Best fitness: 21664 Tardiness jobs: 16

Last Test Times

Start Time 17:37:06 Stop Time 17:37:13 Schedule Time (sec.) 7

Scheduling parameters

Population size: 100

Stopping condition: 100 generations Or Optimal Value: NULL

Selection Operator: Randomly Selection Parents

Crossover Operator: Two-points crossover: 4 children Crossover prob. (Pc): 1.0

Mutation Operator: Arbitrary 20% job change Mutation prob. (Pm): 1.0

Elitist Strategy: Pure Elitist (Version II)

Scheduling

Inicial population: [✓ Randomly] Constant

Number of tests: 1

(C) Copyright 2006 12-02-2006 17:37:26

Fig. 12. Interface window.

9.3 Scheduling Module

In this module, we implemented the GA shown in Fig. 13. The objective of the scheduling module is to give the optimal solution of any scheduling problem. If the optimal solution is not found, the GA gives the best solution found (near-optimal solution).

Step 1 - Initialization
Let t=0, where t is the generation index, and generate an initial population randomly Ψ_0 including N_{pop} solutions (N_{pop} is the number of solutions in each population, i.e., N_{pop} is the population size). The number of solutions (chromosomes) in the t generation is given by $\Psi_t = \left\{x_t^1, x_t^2, \ldots, x_t^{N_{pop}}\right\}$.
Step 2 - Selection
Select pairs of solutions (parents' chromosomes) from the current population Ψ_t. Each chromosome x_t^i is selected according to the selected operator chosen in the interface module.
Step 3 - Crossover
Apply a crossover operator, selected in the interface module, to each of the selected pairs in step 2. This way, new chromosomes will be generated according to the crossover probability (P_c) selected.
Step 4 - Mutation
Apply a mutation operator, selected in the interface module, to the generated chromosomes in step 3, according to the mutation probability (P_m) selected.

Step 5 – Elitism
Select the best N_{pop} chromosomes to generate the next population Ψ_{t+1} and the other chromosomes are eliminated. Thus, the best chromosomes, i.e. solutions, will survive into the next generation. However, duplicated solutions may occur in Ψ_{t+1}. To minimize this, new chromosomes are generated for all duplicated chromosomes.
Step 6 – Termination test
Stops the algorithm if the stopping condition, specified previously in the interface module, is satisfied. Otherwise, update t for $t:=t+1$ and return to step 2.

Fig. 13. GA implemented in the scheduling module.

10. Computer Simulations

This section presents the results of the computational tests performed to examine the crossover and mutation operators presented in section 7. The computational tests were carried out on a PC Pentium IV, 3 GHz and 1 GB of RAM.

10.1 Test Conditions

For each job i (i=1, ..., n) both the processing time p_i and the weight w_i are randomly generated integers from the closed interval [1, 100] and [1, 10] respectively. The due date d_i for each job is a randomly generated integer from the interval [P(1-TF-RDD/2), P(1-TF+RDD/2)], where RDD is the range of due dates, TF is the tardiness factor and $P = \sum_{i=1}^{n} p_i$. RDD and TF can assume the following values: 0.2, 0.4, 0.6, 0.8 and 1.0. By varying RDD and TF we can generate instance classes of different hardness.
As a test problem, we randomly generated a SMTWT scheduling problem with 40 jobs to use in computational tests.

10.2 Examination of Crossover Operators

We applied the HybFlexGA presented in section 9, with the objective of examining the twelve crossover operators in subsection 8.1. When the crossover operators were examined the mutation operator was not used. Each crossover operator was examined by using the following conditions:

- Number of tests: 20.
- Initial population Ψ_t : constant.
- Number of jobs: 40.
- Instance used: constant.
- Population size N_{pop}: 20, 40, 60, 80 and 100.
- Stopping condition: 1000 generations.
- Crossover probabilities P_c: 0.2, 0.4, 0.6, 0.8 and 1.0.
- Mutation operators and mutation probabilities: not used.

We used the following performance measure with the aim of evaluating each genetic operator:

$$Performance = f(\bar{x}_{initial}) - f(\bar{x}_{end}) , \qquad (1)$$

where $x_{initial}$ is the best chromosome in the initial population and x_{end} is the best chromosome in the last population. That is, $f(\bar{x}_{initial})$ is the fitness average (of the 20 computational tests) of the best chromosomes in the initial population and $f(\bar{x}_{end})$ is the fitness average of the best chromosomes in the end of the 1000 generations. The performance measure in (1) gives the total improvement in fitness during the execution of the genetic algorithm.

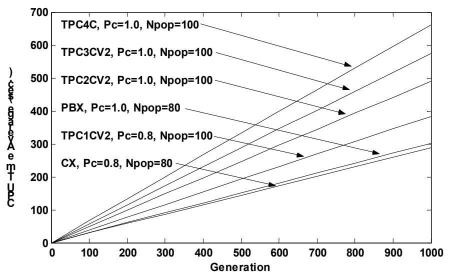
We used 20 computational tests to examine each crossover operator. The average value of the performance measure in (1) was calculated for each crossover operator with each crossover probability and each population size.

Table 4 shows the best average value of the performance measure obtained by each crossover operator with its best crossover probability and its best population size. This table shows the crossover operator by classification order.

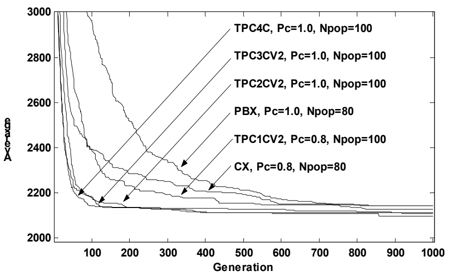
Position	Crossover	P_c	N_{pop}	Performance
1st	TPC4C	1.0	100	3834.1
2nd	TPC3CV2	1.0	100	3822.9
3rd	TPC2CV2	1.0	100	3821.8
4th	PBX	1.0	80	3805.8
5th	TPC1CV2	0.8	100	3789.3
6th	CX	0.8	80	3788.7
7th	TPC3CV1	0.8	80	3680.2
8th	TPC2CV1	1.0	80	3662.1
9th	OPC2C	0.6	100	3647.8
10th	OX	1.0	100	3635.4
11th	TPC1CV1	1.0	100	3624.7
12th	OPC1C	0.6	100	3570.5

Table 4. Classification of the crossover operators.

Fig. 14 presents the CPU time average and the fitness average along the 1000 generations. As we can see in Fig. 14 a), TPC4C, TPC3CV2 and TPC2CV2 require more CPU time than the other crossover operators. On the other hand, Fig. 14 b) shows these crossover operators present a very fast evolution at first (in the first 100 generations).



a)



b)

Fig. 14. The six best crossover operators: a) CPU time average along the 1000 generations b) Fitness average along the 1000 generations

For the same number of generations, PBX and CX do not need as much CPU time (see Fig. 14 a)) but, these operators present a worse fitness average evolution (see Fig. 14 b)). These crossover operators obtained the best performance for N_{pop}=80 (see Table 4) and this is one of the reasons both obtained a good CPU time average.

From Fig. 14 b) we can see TPC4C is the best crossover operator along the 1000 generations, but Fig. 14 a) shows that it needed more CPU time.

10.3 Examination of Mutation Operators

We applied the HybFlexGA presented in section 9, with the objective of examining the five mutation operators in subsection 8.2. When the mutation operators were examined the crossover operator was not used. Each mutation operator was examined by using the same conditions used in the examination of crossover operators.

The average value of the performance measure in (1) was calculated for each mutation operator with each mutation probability and each population size.

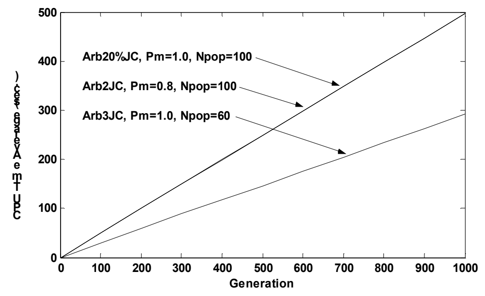
Table 5 shows the best average value of the performance measure obtained by each mutation operator with its best mutation probability and its best population size. This table shows the mutation operator by classification order.

Position	Mutation	P_m	N_{pop}	Performance
1st	Arb20%JC	1.0	100	3833.9
2nd	Arb2JC	0.8	100	3826.4
3rd	Arb3JC	1.0	60	3814.9
4th	SC	0.8	60	3673.5
5th	Adj2JC	0.4	100	3250.4

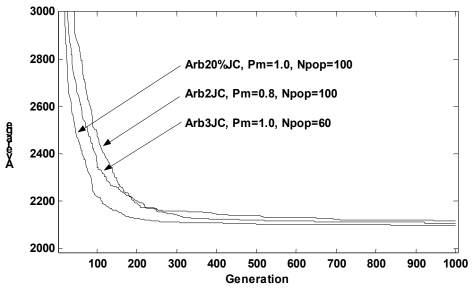
Table 5. Classification of the mutation operators

Fig. 15 presents the CPU time average and the fitness average along the 1000 generations. As we can see in Fig. 15 a), Arb20%JC and Arb2JC require more CPU time than Arb3JC. The CPU time for the Arb20%JC and for Arb2JC are very similar but, in Arb20%JC the CPU time is a little more. For this reason, in Fig. 15 a) we cannot see any difference in its lines. The Arb3JC operator obtained the best performance for N_{pop}=60 (see Table 5) and this is the reason it has a good CPU time average.

As we can see in Fig. 15 b), the Arb20%JC is the best mutation operator along the 1000 generations, but Fig. 15 a) shows that it needs more CPU time.



a)



b)

Fig. 15. The three best mutation operators: a) CPU time average along the 1000 generations b) Fitness average along the 1000 generations.

11. Computational Results

This section presents the computational results obtained in the single machine total weighted tardiness (SMTWT) problem. In the SMTWT n jobs have to be sequentially processed on a single machine. Each job i has an associated processing time p_i, a weight w_i, and an associated due date d_i associated, and the job becomes available for processing at time zero. The tardiness of a job i is defined as $T_i=\max\{0, C_i-d_i\}$, where C_i is the completion time of job i in the current job sequence. The objective is to find a job sequence which minimizes the sum of the weighted tardiness given by $\sum_{i=1}^{n} w_i.T_i$.
For arbitrary positive weights, the SMTWT problem is strongly *NP*-hard (Lenstra et al., 1977). Because the SMTWT problem is *NP*-hard, optimal algorithms for this problem would require a computational time that increases exponentially with the problem size (Morton & David, 1993), (Baker, 1974), (Lawer, 1977), (Abdul-Razaq et al., 1990) and (Potts & Wassenhove, 1991). Several algorithms have been proposed to solve this, for example: branch and bound algorithms (Shwimer, 1972), (Rinnooy et al., 1975), (Potts & Wassenhove, 1985) and dynamic programming algorithms (Schrage & Baker, 1978), have been proposed to generate exact solutions, i.e., solutions that are guaranteed to be optimal solutions. But, the branch and bound algorithms are limited by computational times and the dynamic programming algorithms are limited by computer storage requirements, especially when the number of jobs is more than 50. Thereafter, the SMTWT problem was extensively studied by heuristics. These heuristics generate good or even optimal solutions, but do not guarantee optimality. In recent years, several heuristics, such as Simulated Annealing, Tabu Search, Genetic Algorithms and Ant Colony (Huegler & Vasko, 1997), (Crauwels et al., 1998), and (Morton & David, 1993), have been proposed to solve the SMTWT problem.
We going to present, in this section, the computational results obtained with 40, 50 and 100 jobs, using Genetic Algorithms. From the OR-Library (http#3), we randomly selected some instances of SMTWT problems with 40, 50 and 100 jobs. We used 20 computational tests for each instance of SMTWT problem. We used the six best crossover operators (see Table 4) and the best mutation operator (see Table 5) in the HybFlexGA. Each instance of SMTWT problem was examined by using the following conditions:

- Number of tests: 20.
- Initial population Ψ_t: randomly generated.
- Number of jobs: 40, 50 and 100.
- Instance used: from the OR-Library (http#3).
- Population size N_{pop}: 80 and 100 (see Table 4 and Table 5).
- Stopping condition: 1000 generations for the instances with 40 and 50 jobs or the optimal solution, and 5000 generations for the instances with 100 jobs or the optimal solution.
- Crossover operators: the six best crossover operators in Table 4.
- Crossover probabilities P_c: 0.8 and 1.0 (see Table 4).
- Mutation operators: the best mutation operator in Table 5.
- Mutation probabilities P_m: 1.0 (see Table 5).

	Instance	40A	40B	40C	50A	50B	50C	100A	100B	100C
	Optimal solution	**6575**	**1225**	**6324**	**2134**	**22**	**2583**	**5988**	**8**	**4267**
TPC4C + Arb20% JC	Tests with optimal solution	16	20	19	20	20	15	16	20	19
	CPU time average (sec.)	362.4	190.0	319.7	88.3	45.5	214.1	2405.1	523.9	3213.5
	Generations average	593	284	475	107	54	256	1611	323	1971
TPC3C V2 + Arb20% JC	Tests with optimal solution	13	15	15	18	20	12	15	20	15
	CPU time average (sec.)	382.9	231.3	260.5	112.3	50.4	207.9	3851.0	1012.4	4453.7
	Generations average	725	402	448	158	70	290	3042	727	3181
TPC2C V2 + Arb20% JC	Tests with optimal solution	8	16	15	17	20	12	9	20	5
	CPU time average (sec.)	369.1	216.8	305.8	146.2	92.0	157.8	3921.3	1339.8	4036.2
	Generations average	380	449	627	246	152	263	3685	1142	3423
PBX + Arb20% JC	Tests with optimal solution	0	8	1	18	20	10	1	20	3
	CPU time average (sec.)	---	236.1	312.0	144.6	86.3	151.6	2067.0	1413.6	3237.7
	Generations average	---	747	976	371	218	385	2957	1828	4182
TPC1C V2 + Arb20% JC	Tests with optimal solution	0	3	2	13	20	5	1	19	2
	CPU time average (sec.)	---	230.0	363.5	224.7	118.2	180.0	4104.0	2190.7	3847.0
	Generations average	---	609	956	487	252	385	4948	2405	4192
CX + Arb20% JC	Tests with optimal solution	0	4	4	17	20	11	3	20	5
	CPU time average (sec.)	---	165.3	270.5	107.8	51.0	140.0	2594.0	736.4	3224.2
	Generations average	---	551	892	292	136	375	3912	1011	4390

Table 6. Computational results obtained for the SMTWT problems with 40, 50 and 100 jobs.

Table 6 shows the computational results obtained for the SMTWT problems with 40, 50 and 100 jobs. In this table, we have the number of tests with optimal solution, the CPU time average (in seconds) and the generations average for each instance problem. For example, we chose the TPC4C with P_c=1.0, Arb20%JC with P_m=1.0 and instance 40A (SMTWT problem with 40 jobs, from the OR-Library (http#3)) in the HybFlexGA. We did 20 computational tests for this instance. In these tests we obtained the optimal solutions in 16 tests. In these 16 tests, the CPU time average was 362.4 seconds and the generation average was 593.

As we can see in Table 6, we obtained good results with the TPC4C+Arb20%JC, TPC3CV2+Arb20%JC and TPC2CV2+Arb20%JC combination, for all the instances with 40, 50 and 100 jobs. But, as we can see in this table, the best results are obtained for the TPC4C+Arb20%JC combination. When we used the TPC4C+Arb20%JC combination, the HybFlexGA is very efficient. For example, in the three instances with 40 jobs (see Table 6 – 40A, 40B and 40C) the HybFlexGA found 20 tests with optimal solution in one instance (40B), 19 tests with optimal solutions in one instance (40C) and 16 tests with optimal solutions in one instance (40A).

In subsection 10.2 and 10.3, we demonstrated that TPC4C and Arb20%JC need more CPU time (for the same generations) than the other genetic operators. But, we also demonstrated that TPC4C and Arb20%JC need fewer generations to find good solutions, probably the optimal solutions. The computational results obtained in Table 6 show the HybFlexGA with the TPC4C+Arb20%JC combination requires less CPU time than the one with the other combinations. For example, in the 50B instance the HybFlexGA always found the optimal solution for all combinations (e.g., TPC3CV2+Arb20%JC, TPC2CV2+Arb20%JC, and so on). For this instance, as we can see in Table 6, the combination that needs less CPU time average was the TPC4C+Arb20%JC (45.5 seconds). All the other combinations need more CPU time average.

12. Conclusion

Robust control software is a necessity for an automated FMS and FMC, and plays an important role in the attainable flexibility. Often FMS and FMC are built with very flexible machines (robots, CNC machines, ASRS, etc) but the available control software is unable to exploit the flexibility of adding new machines, parts, changing control algorithms, etc. The present robot control and CNC machines control systems are closed and with deficient software interfaces. Nowadays, the software is typically custom written, very expensive, difficult to modify, and often the main source of inflexibility in FMS. We have developed a collection of software tools: “winRS232ROBOTcontrol”, “winEthernetROBOTcontrol”, “winMILLcontrol”, “winTURNcontrol” and USB software. These software tools allow the development of industrial applications of monitoring and controlling robotic networks and CNC machines inserted in FMS or FMC. All the software developed has operation potentialities in Ethernet. The basic idea is to define for any specific equipment a library of functions that in each moment define the basic functionality of that equipment for remote use.

In this chapter we also propose a new concept of genetic operators for scheduling problems. We developed a software tool called HybFlexGA, to examine the performance of various crossover and mutation operators by computing simulations on job scheduling problems. The HybFlexGA obtained good computational results in the instances of SMTWT problems with 40, 50 and 100 jobs (see Table 6). As we demonstrated, the HybFlexGA is very efficient with the TPC4C+Arb20%JC combination. With this combination, the HybFlexGA always found more optimal solutions than with the other combinations: TPC3CV2+Arb20%JC, TPC2CV2+Arb20%JC, and so on. When we used this combination (TPC4C+Arb20%JC) in the HybFlexGA, the genetic algorithm required fewer generations and less CPU time to find the optimal solutions. These results show the good performance and efficiency of the HybFlexGA with the TPC4C and Arb20%JC genetic operators.

Application of the HybFlexGA, with these same genetic operator combinations, in other scheduling problems (e.g. job-shop and flow-shop) is left for future work.

13. References

Abdul-Razaq, T.; Potts C. & Wassenhove L. (1990). A survey for the Single-Machine Scheduling Total WT Scheduling Problem. *Discrete Applied Mathematics*, No 26, (1990), pp. 235–253.

Baker, R. (1974). *Introduction to Sequencing and Scheduling*, Wiley, New York.

Crauwels H.; Potts C. & Wassenhove L. (1998). Local search heuristics for the single machine total weighted tardiness scheduling problem. *INFORMS Journal on Computing*, Vol. 10, No 3, (1998), pp. 341–350.

Ferrolho A.; Crisóstomo M. & Lima M. (2005). Intelligent Control Software for Industrial CNC Machines, *Proceedings of the IEEE 9th International Conference on Intelligent Engineering Systems*, Cruising on Mediterranean Sea, September 16-19, 2005, in CD-ROM.

Ferrolho A. & Crisóstomo M. (2005-a). Scheduling and Control of Flexible Manufacturing Cells Using Genetic Algorithms. *WSEAS Transactions on Computers*, Vol. 4, 2005, pp. 502–510.

Ferrolho A. & Crisóstomo M. (2005-b). Flexible Manufacturing Cell: Development, Coordination, Integration and Control, in *Proceedings of the IEEE 5th Int. Conference on Control and Automation*, pp. 1050-1055, Hungary, June 26-29, 2005, in CD-ROM.

Ferrolho A. & Crisóstomo M. (2005-c). Genetic Algorithms for Solving Scheduling Problems in Flexible Manufacturing Cells, in *Proceedings of the 4th WSEAS International Conference on Electronics, Signal Processing and Control (ESPOCO2005)*, Brazil, April 25-27, 2005, in CD-ROM.

Ferrolho A. & Crisóstomo M. (2005-d). Genetic Algorithms: concepts, techniques and applications. *WSEAS Transactions on Advances in Engineering Education*, Vol. 2, 2005, pp. 12–19.

French S. (1982). *Sequencing and Scheduling: An Introduction to the Mathematics of the Job-Shop*, John Wiley & Sons, New York.

Goldberg D. (1989). *Genetic Algorithms in Search, Optimization and Machine Learning*, Addison-Wesley.

Hadj-Alouane N.; Chaar J.; Naylor A. & Volz R. (1988). Material handling systems as software components: An implementation. *Tech. Paper, Center for Res. on Integrated Manufacturing*. The Univ. Michigan, May 1988.

Huegler P. & Vasko F. (1997). A performance comparison of heuristics for the total weighted tardiness problem. *Computers & Industrial Engineering*, Vol. 32, No 4, (1997), pp. 753–767.

Joshi S.; R. A. Wysk & E. G. Mettala (1991). Automatic Generation of Control System Software for Flexible Manufacturing Systems. *IEEE Trans. on Robotics and Automation*, Vol. 7, No. 6, (December 1991) pp. 283–291.

Joshi, S.; J. S. Smith; R. A. Wysk; B. Peters & C. Pegden (1995). Rapid-CIM: An approach to rapid development of control software for FMS control. *27th CIRP International Seminar on Manufacturing Systems*, Ann Arbor, MI, 1995.

Kusiak A., (1986). *Modelling and Design of Flexible Manufacturing Systems*, Elsevier Science Publishers.

Lawer, E. (1977). A pseudopolinomial algorithm for Sequencing Jobs to Minimize Total Tardiness. *Annals of Discrete Mathematics*, (1977), pp. 331–342.

Lenstra J.; Kinnooy Kan H. & Brucker P. (1977). Complexity of machine scheduling problems. *Annals of Discrete Mathematics*, Vol. 1 (1977), pp. 343–362.

Maimon O. & Fisher E. (1988). An object-based representation method for a manufacturing cell controller. *Artificial Intelligence in Engineering*, Vol. 3, No. 1.

Morton, E. & David W. (1993). *Heuristic Scheduling Systems*, John Wiley & Sons.

Murata T. & Ishibuchi H. (1994). Performance Evaluation of Genetic Algorithms for Flowshop Scheduling Problems, *Proceedings of the 1st IEEE Int. Conference on Evolutionary Computation*, pp. 812–817, Orlando, USA.

Oliver J.; Smith D. & Holland J. (1987). A study of permutation crossover operators on the traveling salesman problem, *Proceedings of the Second ICGA* , pp. 224–230.

Potts C. & Wassenhove L.(1985). A branch and bound algorithm for the total weighted tardiness problems. *Operations research*, Vol. 33, No 2, (1985), pp. 363–377.

Potts, C. & Wassenhove L. (1991). Single Machine Tardiness Sequencing Heuristics. *IIE Transactions*, Vol. 23, No 4, (1991), pp. 346–354.

Rembold U.; Nnaji B. & Storr A. (1993). *Computer Integrated Manufacturing and Engineering*, Addison-Wesley.

Rinnooy H.; Lageweg B. & Lenstra J. (1975). Minimizing total costs in one-machine scheduling. *Operations research*, Vol. 23, No 5, (1975), pp. 908–927.

Sanjay B. Joshi; Erik G. Mettala; Jeffrey S. Smith & Richard A. Wysk (1995). Formal Models for Control of Flexible Manufacturing Cells: Physical and System Model. *IEEE Trans. on Robotics and Automation*, Vol. 11, No. 4, (August 1995) pp. 558–570.

Schrage L. & Baker K. (1978). Dynamic programming solution of sequencing problem with precedence constraints. *Operations research*, Vol. 26, (1978), pp. 444–449.

Shwimer J. (1972). On the n-job, one-machine, sequence-independent scheduling problem with tardiness penalties: a branch-bound solution. *Management Science*, Vol. 18, No 6, (1972), pp. 301–313.

Syswerda G. (1991). *Scheduling optimization using genetic algorithms*. L. Davis (Ed.) Handbook of Genetic Algorithms, Van Nostrand Reinhold, New York, pp. 332–349.

Waldner JB., (1992). *CIM, Principles of Computer Integrated Manufacturing*, John Wiley & Sons.

Sites WWW

http#1 - Rapid Prototyping and Development of FMS Control Software for Computer Integrated Manufacturing. Available: http://tamcam.tamu.edu/rapidcim/rc.htm

http#2 - RapidCIM. Available: http://www.engr.psu.edu/cim/

http#3 - http://people.brunel.ac.uk/~mastjjb/jeb/orlib/wtinfo.html

Adaptive Robot Based Reworking System

Bernd Kuhlenkötter, Carsten Krewet, Thorsten Schüppstuhl
Robotics Research Institute, Dortmund University,
Otto-Hahn-Straße 8, Dortmund 44227, Germany

1. Introduction

Surface quality plays an important role for both components and products. Either are an exact form and a dimensional accuracy required to fulfil a certain function, e.g. in the case of turbine blades, or the visual impression of a surface must meet high demands, e.g. in sanitary fittings. In order to meet these high demands in material processing, processes such as grinding and polishing are often used (VDMA, 2000).
Nowadays the processes of grinding and polishing are automised by industrial robots to release the operator and to grant an economical manufacturing. Presently known systems however face the problem that they can be adjusted to other part geometries and processing cycles only at high costs and therefore are used only in large batches (Schraft & Kaun, 1999). One reason are the work-intensive and time-consuming programming and optimisation of the movement paths, and another is that the handling systems do not recognize the existing defects on the surface resulting from grinding and polishing. As a defect-related adaptation to the subsequent processing steps is not possible, the reprocessing of the surfaces is done manually.
In this chapter a concept is presented which makes it possible to automatise the reworking of design surfaces of high-quality sanitary fittings with the help of industrial robots.

2. Analysis of the initial situation in the reworking of sanitary fittings

Design surfaces of sanitary fittings must meet high optical demands whereas a millimetre accuracy to size of the fittings is not required. The slightest defects in the surface, such as pores or cracks, and also slightly visible processing traces lead to fittings not being sold anymore. For this reason defective fittings must be reworked at high costs. Since current handling and robot systems are not flexible enough to react, the reworking is usually done manually.
The course of the manual reworking is the following:
First the operator checks the ground and subsequently high-polished fitting for possibly occurring defects in the surface. Frequently occurring defects in the manufacturing process are, for example pores, cracks or gas pockets.
When the operator thinks that the defects are small and can be removed by reprocessing, he marks the area, and passes the fitting on to the reworking station. If the defects are too big, he will reject the fitting. In the reworking station the defective area is first ground and then polished.

When analysing the manual reworking more closely, one sees that it is an iterative process. The operator first grinds a little area in the centre of which the defect is to be removed and then he checks the result. If the defect has not been removed completely, he repeats grinding it, but this time he will work on a bigger area. This is repeated until the defect is removed.
Upon removal of the defect the operator tries to clear the traces of the processing by further grinding paths, finally he polishes the reworked surface largely.
The most important aspects of the manual reworking can be summarized as follows:

- Every high-polished fitting is checked for surface defects.
- Big defects are scrapped, little ones are reworked.
- If possibly, only little areas are reworked.
- There are two types of grinding paths:
  1. Paths to remove the defect.
  2. Paths to remove the processing traces.
- The result of the previous processing step is directly checked.
- The processing strategy is adapted to the result of the previous grinding step.

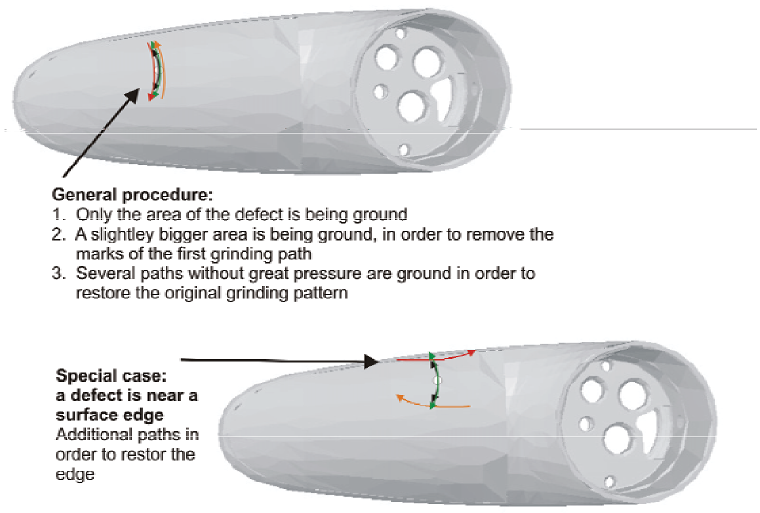

General procedure:
1. Only the area of the defect is being ground
2. A slightley bigger area is being ground, in order to remove the marks of the first grinding path
3. Several paths without great pressure are ground in order to restore the original grinding pattern
Special case:
a defect is near a surface edge
Additional paths in order to restor the edge

Fig. 1. Example for manual rework strategies.

3. Analysis of the transferability of manual rework to a robot-aided solution

If you try to automatise the above mentioned reworking process with the help of an industrial robot system, you will face various problems which make an adaptation to the abilities of a robot necessary.
When reworking, the robot system must take over the following tasks:

- To handle the sanitary fitting during the checking and reworking process.
- To check the sanitary fitting for surface defects.
- To make a decision if a defective fitting is to be reworked or scrapped.
- To select an appropriate strategy for the reprocessing and generating of the robot paths necessary for reworking.

- To process the defective fittings with the means of surface processing, such as grinding and polishing.

An image processing system is used for checking the surface quality (see also Kuhlenkötter, et al, 2004). This image processing system has an illumination especially developed for checking high-polished components to detect defects up to a minimum size of 0.3 mm and to classify them into different defect and polluting groups. For further information about the classification system see (Zhang, et al., 2006).

Due to the time needed for the checking of the complete surface (up to 25 seconds) it is economically not advisable to recheck the fitting each time after a processing step what makes a steady adaptation of the processing strategy - as it is done in manual rework - impossible.

Therefore the robot system must select a suited strategy for the whole reworking process and generate processing paths as to the strategy and adapt them to the previous case of defect already before the reworking process starts.

This reason entails another change in comparison to the manual process. The fact that it is not possible to check after each grinding or polishing process in which condition the surface is in and which defects and processing traces are still visible or have occurred newly, leads in most cases to an enlargement of the area to be processed. By that, it is granted that both the defect and the processing traces are actually removed and no further reworking will become necessary.

The division of the processing paths into groups of defect removal and groups of processing trace removal has been modified in comparison to the manual reworking. In the robot-aided reworking the paths for the removal of processing traces are subdivided into paths which border the paths to the outside and paths which are in the vicinity of the defects to be removed. This additional division helps to better adapt the paths to be generated to the respective location of the defect and the occurring type of defect.

4. Concept of a robot based reworking system

The robot based reworking system requires an intelligent link-up of the vision system for surface inspection, the selection of the right processing strategy and the following generation of the processing paths, which is based on the CAD-model of the handled sanitary fitting. In order to realise this link-up, some important questions need to be answered. One question is, how to draw a conclusion from the 2D information about the location of the defect, which is given by the picture of the vision system, to the 3D location on the CAD-model in a sufficiently accurate way. Another question is, how to generate the paths before each reworking process considering the requirements according to the requested station times. Or, if it is more practical to implement a database solution with stored processing strategies and appropriate paths and use a kernel function to select the elements out of the database which have the biggest similarity with the present case and interpolate between these elements.

Answers to these questions must be found to develop a reworking system which can be used under real production conditions.

In the application introduced here the 3D-position of the defect on the surface of the sanitary fitting and the CAD-model, respectively is determined by a projection of the defect-position on the 2D-picture of the vision system onto the surface of the CAD-model (see also Fig. 2).

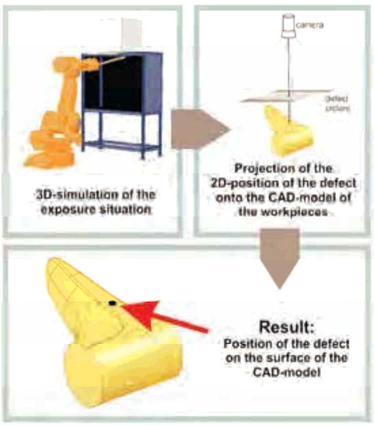

camera
defect picture
3D-simulation of the exposure situation
Projection of the 2D-position of the defect onto the CAD-model of the workpieces
Result:
Position of the defect on the surface of the CAD-model

Fig. 2. Projection of the defect position onto the 3D CAD model.

Such a projection is possible because all needed information like the focal point of the camera lens and the position of the inspected fitting is known, or can be determined. With this method it is possible to get the position of the defect with an accuracy of about two millimetres. A more accurate determination of the position without an additional complex sensor system is not possible. In nearly all cases the reached accuracy is sufficient, so that an additional sensor system is not needed.
The knowledge about the 3D defect position makes it possible to use the information about the position and the shape of the surface around the defect itself to generate geometry based processing strategies.
The shape of a workpiece and the required processing quality lead to high demands on the path planning process for industrial robots. The analysis of the manual rework and the transferability of the strategies of the manual rework to a robot based solution have shown that the loss of degrees of freedom leads to the necessity of developing a category of geometrical strategies for the defect removal that are suitable for industrial robots. Therefore the approach of a geometry value modification was used.

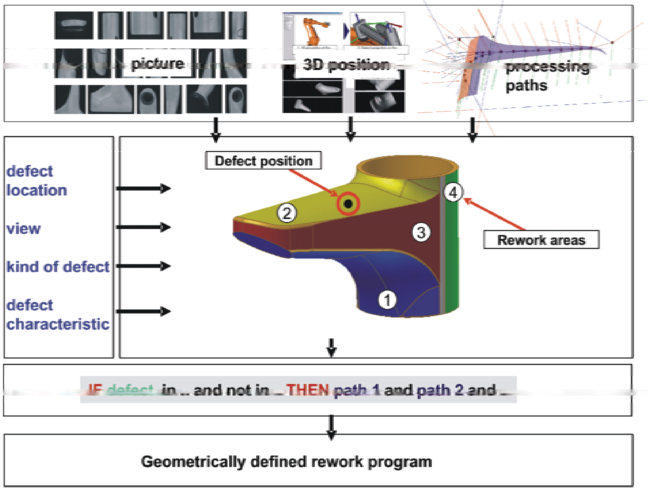


Fig. 3. Generation of a geometry orientated rework program (Kuhlenkötter, et al, 2005).

The first step in the generation of a complete rework program is the definition of the movement paths. This definition is subject to different restrictions that have to be considered, e.g. the geometric coverage of the defect area, the reachability, the retractability and the collision freeness of the generated robot paths.

To develop a suitable method it had become necessary to do a large number of grinding and polishing tests. At first, the robot program for the normal manufacturing process had to be created. Here it was very important that the dependency between the workpiece surface and the processing paths was defined accurately, to make the identification of special process correlations possible. To use teach-in robot programs for reaching the needed accuracy was not possible, but instead programs that had been created and optimised totally newly and mathematically accurately in an offline-programming system.

During the testing of different rework strategies it turned out that besides the mentioned constraints the perpetuation of the same grinding pattern of reworked and not reworked areas is another important requirement. In order to get a smooth invisible transition the approach of an area based reworking was selected. That means, the workpiece surface is divided once into different rework areas in the setup process. In the rework process the complete affected area is reworked. If e.g. a defect is found in area 3, the whole area 3 is reworked, and if defects appear in more than one area, all affected areas are reworked. The most significant criterion for the predefinition of the rework areas is the realising of a transition free processing at the area edges.

As far as possible, the paths of the original processing program should be used also for rework, because they are already defined to a technologically und geometrically very accurate extent fulfilling the requirements mentioned above. If this is not possible, new paths will have to be generated.

According to the current state of the art and research an automated path generation cannot meet all requirements in a satisfactory way. Especially the required accuracy of the automatically generated paths is not good enough due to a lack of appropriate process models. Therefore a partly automated approach for the generation of the rework program was selected (see Fig. 3). The realised system uses a manual predefinition of rework areas and a set of predefined rework strategies for different defect situations and defect areas. This predefinition is done in the setup process of the rework cell.

In the automated part of the developed process controller the defect data of the vision system is pre-processed in a first step. After that, the data is used to select the suitable processing paths automatically and to transfer them to the robot control in a structured way including appropriate start and end points. This adaptation to the current defect situation takes place directly before the rework process and is carried out whenever the vision system detects a defect on the inspected fitting.

Besides the geometrical constraints the requirements the processes themselves define are taken into account. Thus it is necessary to develop a group of technological strategies for the rework process that enables the robot system to gain an optimum rework result from the technological point of view. In order to realise a fully automated rework in the presented application, a class of qualitative defect removal strategies was developed based on technology value modifications. The developed solution for the generation of a rework program uses a two-stage method. In a first stage, a geometric movement program for the rework of the whole affected defect area is generated. This program consists of geometric robot paths that cover the whole area to be processed as well as standard values for the technology parameters. In a second stage, the standard values of the technology parameters of the program such as feed speed and infeed are

adapted automatically to the existing defect situation in a way that the defects are optimally removed and the edges to the not reworked areas satisfy the quality requirements.
This means, that in the defect vicinity the robot program must be adapted to the shape of the defect, while at the edge of the rework area it must be adjusted to the neighbouring areas. In the realized system a classification of the path points is used based on their function inside the rework program (see Fig. 4). This classification is done once during the programming process and according to the current state of the art and research it only can be done manually. To shorten this time-consuming work, the classification into function groups was integrated into an off-line programming system that was developed in the same project by a participating company. By this, a graphically interactive, very efficient and descriptive execution of these operations becomes feasible.

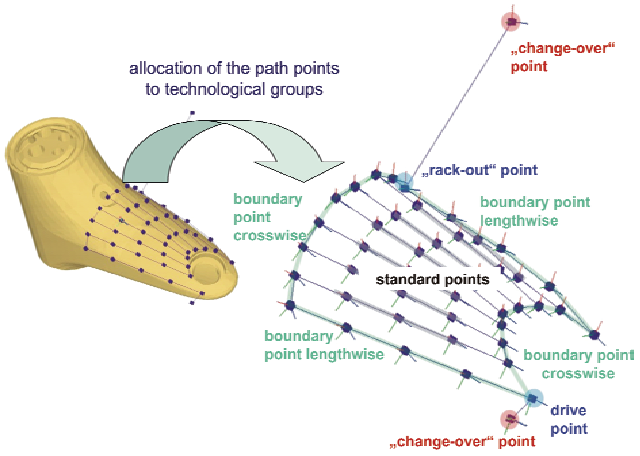


Fig. 4. Classification of the path point into function groups (Kuhlenkötter, et al, 2005).

In order to realise a flexible solution for the technology value modification scale factors were integrated into the control structure of the system.
These scale factors help to adapt predefined default values to different defect situations. Based on the classification of points different scale factors can be defined both for several rework areas and function groups.
These scale factors represent the relationship between the technology parameters within the defect area and the remaining areas so that the technology parameters can be adapted to the current defect situation easily and effectively.
In order to meet the demands on the adaptability of the system an additional layer was added to the described control structure. Besides the adaptation of the technology parameters an adjustment of the rework intensity was implemented into the control structure, by which a multi-level approach could be realised. One out of four different levels of rework intensity can be selected depending on shape, size and class of the defect.
The following levels are available:

- Level 1: polishing and high-gloss polishing
- Level 2: fine grinding, polishing and high-gloss polishing
- Level 3: light rough grinding, fine grinding, polishing and high gloss polishing
- Level 4: intensive rough grinding, fine grinding, polishing and high gloss polishing

An analysis during the realisation of this level based approach showed that in the majority of the occurring cases level 2 is sufficient for the rework intensity. The use of a rough grinding process step is normally not necessary.

5. Realisation of the reworking system

For the realisation of the presented concept for a robot based reworking system the following control and software structure was implemented. The system consists of three software modules. One module is a robot-aided software that performs the communication processes and data exchange to the external control computer and the control of the vision system. Another module is the software system that was developed for the external control computer. This software system is able to perform the described procedure of generating the rework program. The basis of this software is the software framework "DirectControl" being developed at the Robotics Research Institute of Dortmund University. The framework "DirectControl" provides several base functionalities for the development of complex robot applications like communication protocols and simulation tools.
The third software module is the off-line programming system FAMOS robotic V7 that was developed by the company Carat Robotic Innovation. This software system provides the possibility of predefining the rework areas and programming the strategies for processing the rework areas. Compared to conventional systems this software system reduces the effort of programming a rework robot cell significantly.

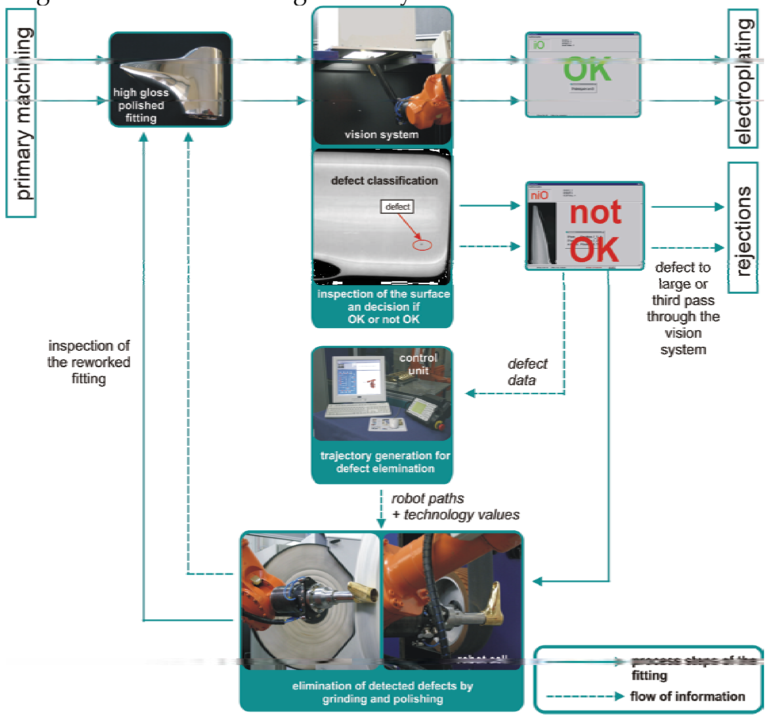

primary machining
high gloss polished fitting
vision system
OK
electroplating
defect classification
defect
inspection of the surface an decision if OK or not OK
not OK
rejections
defect to large or third pass through the vision system
inspection of the reworked fitting
control unit
defect data
trajectory generation for defect elemination
robot paths + technology values
robot cell
elimination of detected defects by grinding and polishing
process steps of the fitting
flow of information

Fig. 5. Flow chart of the fully automated rework process chain.

Using the described software structure, the following sequence for a robot based rework was realised (see Fig. 5). A high-glossed polished fitting is inspected by the vision system. If no defect is found on the surface, the fitting is transferred to the next manufacturing process, the electroplating without a reworking process. If one or more defects are found on the surface of the fitting, the size of the defect is analyzed. If the defect is too large, the fitting is scrapped. Otherwise the defect data and the fitting are transferred to the control unit and rework cell. By analysing the defect data the control unit generates the rework program that consists of the robot paths and the adapted technology values and transfers it to the robot control. Then the defect affected areas of the fitting are reworked. After rework the fitting is inspected again by the vision system and the whole process starts again. If after a third inspection a fitting still shows a defect on its surface, it is scrapped regardless of the size of the defect.

6. Conclusion and prospects

In this article the realisation of a robot based rework of design surfaces is presented taking sanitary fittings as an example. Because of known circumstances it is not possible to include all strategies used in the manual reworking process in the robot based solution. Some adjustments are necessary to make according to the station times and abilities of the robot system itself.
Nevertheless it is already now possible to automatise such complex processes as the reworking of design surfaces.
At the Robotics Research Institute of Dortmund University such a robot system was built up and tested successfully. Presently, parts of the system are used by a manufacturer of sanitary products with a promising future. This manufacturer is planning to put a whole robot cell for automated reworking into operation in the near future.

7. Acknowledgements

This research and development project is funded by the German Federal Ministry for Education and Research (BMBF) within the national programme "Research in Production for Tomorrow" and supervised by the Project Agency for Production and Manufacturing Technologies, Forschungszentrum Karlsruhe (PTKA-PFT), Germany.

8. References

Krewet, C.; Kuhlenkötter, B.; Schüppstuhl, T. (2005): Optimierung robotergestützter Fertigungsprozesse – Effiziente Optimierung von Bearbeitungsprogrammen für komplexe Werkstückgeometrien. wt Werkstattstechnik online, Jahrgang 95 (2005) H. 3, Springer-VDI-Verlag, Düsseldorf, 2005

Kuhlenkötter, B., Schüppstuhl, T. (2005): Vollautomatisierung durch innovative Robotersysteme. VDI-Berichte 1892.2, Mechatronik 2005, Innovative Produktentwicklung, VDI Verlag, Düsseldorf 2005

Kuhlenkötter, B.; Steib, S.; Schüppstuhl, T.; Raedt, H.-W. (2004) : Möglichkeiten zur automatisierten Fehlererkennung, -markierung und -nachbearbeitung auf Oberflächen geschmiedeter und gegossener Bauteile. Schmiedejournal, September 2004, ISSN 0933-8330

Schraft, R.D.; Kaun, R.: Automatisierung, Stand der Technik, Defizite und Trends in der Automatisierungstechnik. Verlagsgruppe Handelsblatt GmbH, Wirtschaftwoche, Düsseldorf, 1999

VDMA; Fachverband Robotik + Automation: Protrait der Branche 1999/2000, Frankfurt/Main, 2000

Zhang, X.; Krewet, C.; Kuhlenkötter, B. (2006): Automatic classification of defects on the product surface in grinding and polishing. International Journal of Machine Tools & Manufacture, Volume 46, Issue 1, pp. 59-69, Elsevier Science, January 2006

18

Off-line Programming Industrial Robots Based in the Information Extracted From Neutral Files Generated by the Commercial CAD Tools

Vitor Bottazzi, Jaime Fonseca
Department of Industrial Electronics - University of Minho
Portugal

1. Introduction

In order for a robotic manipulator to perform useful work, it must be programmed to accomplish the desired task or motion cycle. Nowadays industrial robots generally require a tremendous amount of programming to make them useful. Their controllers are very sophisticated, the commercial robot programming environments are typically closed systems and the programming languages varies from manufacturer to manufacturer. Despite the great evolution of the industrial robots controllers, in the majority of the industrial applications, the robot programming is made, using one of the following ways:

- Manual on-line programming;
- Off-line programming;

Manual on-line programming refers to physically teaching a robot the required trajectory, through interaction with teach pendant or other similar device (Lee & ElMaraghy, 1990). This programming kind presents the following disadvantages: very slow, it needs that the robot is available, difficulty in the handling of equipments, need some practice in the language used by the robot, and technical knowledge to understand the operation of the equipment. These disadvantages are very expensive in the industry because the productive process needs to stop for a long time.
One simple approach to solve some disadvantages described above is the Off-line programming environments. These environments are based in graphical simulation platforms, in which the programming and execution process are shown using models of the real objects. Consequently, the robot programmer has to learn only the simulation language and not any of the robot programming languages. Other benefits in off-line programming environments include libraries of pre-defined high-level commands for certain types of applications, such as painting or welding, and the possibility to assess the kinematics feasibility of a move, thus enabling the user to plan collision-free paths. The simulation may also be used to determine the cycle time for a sequence of movements. These environments usually provide a set of primitives commonly used by various robots, and produce a sequence of robot manipulator language primitives such as ″move″ or ″open gripper″ that are then downloaded in the respective robot controllers.

However, the off-line programming tools based in graphically 3D representation presents several problems in many industry applications, particularly, when the robot task or the robot trajectory needs frequent changes, for example: in welding applications where the configuration of the pieces to weld change frequently (the size, the shape, etc.); the robot painting and gluing applications can have similar problems.
Nowadays, the CAD tools are often used in the industry to develop and to document the products and its manufacture. There are a lot of commercial CAD tools, like, AutoCAD, SolidWorks, Ideas and Cimatron, having each tool its own file format. However, it is possible to export the information of these pieces, in a neutral file format, namely: STL, IGES, STEP and SET formats.
This work presents one solution for programming different robots based in the relevant information extracted from neutral files. The solution implemented was tested in the industrial robots Mitsubishi (Mitsubishi Move Master Industrial Robot) and ABB (model IRB 140 with IRC5 controller). This chapter is organized as follows: section 2 presents an overview about the format of neutral files (STL, IGS, STEP and SET); in the section 3, the algorithms for extraction of the relevant information from the neutral files are described; in the section 4, the developed tool for code generation for different industrial robots is presented; section 5 and 6 present the results and conclusions; section 7 presents future work.

2. Neutral file formats

Computer Aided Design (CAD) technology for engineering, and manufacturing is now playing an increasingly important role in production industry. The importance of this technology to increase productivity in engineering design has been widely recognised. These technologies make it possible to shorten the time and lower the cost of development. Additionally, the reliability and the quality of the product can be improved. CAD systems have therefore been used in various fields of industry including automobile and aircraft manufacture, architecture and shipbuilding, and there are currently many commercial systems available. SolidWorks, Catia, Inventor and Cimatron are examples of available systems using CAD technology.
With the existence of a great diversity of CAD tools emerge the demand to import/export files between different CAD software. The emergence of neutral format files and neutral format file interfaces in order to exchange product data between CAD systems solve this problem. The most widely accepted formats have been the Initial Graphics Exchange Standard (IGES), the *Standard d'Echange et de Transfert* (SET), the STandard for the Exchange of Product model data (STEP) and the Standard Transform Language (STL).

2.1 Initial Graphics Exchange Standard (IGES)

IGES (Smith et al., 1988) was the first specification for CAD data exchange published in 1980 as a NBS (National Bureau of Standards) report (IGES 1, 1980) in USA.
The version IGES 5.2, provide the following capabilities:

- Geometry : 2D/3D wireframes, curves and surfaces; CSG (Constructive Solid Geometry) and B-Rep (boundary Representation) are supported;
- Presentation : Drafting entities for technical drawings;
- Application dependent elements : Piping and electronic schematics, AEC elements;
- Finite Element Modelling : Elements for FEM (Finite Element Method) systems

An IGES file consists of six sections: Flag, Start, Global, Directory Entry, Parameter Data, and Terminate. Each entity instance consists of a directory entry and parameter data entry. The directory entry provides an index and includes attributes to describe the data. The parameter data defines the specific entity. All the parameter data are defined by fixed length records according to the corresponding entity. Each entity instance has bi-directional pointers between the directory entry and the parameter data section (Vuoskoski, 1996).

2.2 Standard d'Echange et de Transfert (SET)

SET is a French standard for the exchange and archiving of CAE data and is supported by several CAD systems. It was developed as a neutral file format for exchanging data between different CAD systems at Aerospatiale in 1983. The aim was to develop a more reliable alternative to IGES. It supports wireframe, surface and solid models, including CSG and B-Rep. Entities for drafting and connectivity applications, as well as scientific data and FEM (Finite Element Method) modelling are also included. It was considered to be important to have an unambiguously defined format that is compact in size, and is flexible enough to handle future demands from the CAD/CAM industry (Vuoskoski, 1996).
The structure of SET is based on a three-level hierarchy of data assemblies, data blocks, and data sub-blocks. Information that is common to several blocks or assemblies is stored in a so-called dictionary.

2.3 Standard for the Exchange of Product model data (STEP)

STEP (Owen, 1994) is a new International Standard (ISO 10303) for representing and exchanging product model information. It includes an object-flavoured data specification language, EXPRESS (Schenck& Wilson, 1994) to describe the representation of the data. STEP defines also implementation methods, for instance, a physical transfer file, and offers different resources, e.g. geometric and topological representation.
The objective of STEP is to offer system-independent mechanism to describe the product information in computer aided systems throughout its lifetime. It separates the representation of product information from the implementation methods. Implementation methods are used for data exchange.
The representation offers a definition of product information to many applications. STEP provides also a basis for archiving product information and a methodology for the conformance testing of implementations.
EXPRESS is a formal data specification language used to specify the representation of product information. The use of a formal data specification language facilitates development of implementation, and also enables consistency of representation. STEP specifies the implementation methods used for data exchange that support the representation of product information.
STEP does not only define the geometric shape of a product: it also includes topology, features, tolerance specifications, material properties, etc. necessary to completely define a product for the purposes of design, analysis, manufacture, test, inspection and product support. The use of STEP is still very modest but it is growing all the time. The majority of CAD system vendors have implemented or are implementing STEP pre- and post-processors for their CAD systems. STEP is

an evolving standard which will cover the whole product life cycle in terms of data sharing, storage and exchange. It is the most important and largest effort ever established in engineering domain and will replace current CAD exchange standards.

2.4 Standard Transform Language (STL)

STL is a file format native to the stereolithography CAD software created by 3D Systems of Valencia, CA, USA and is perhaps the main standard for rapid prototyping systems. STL files may be ASCII or binary data, although binary is far more common due to the resulting size of the CAD data when saved to the ASCII format.

a) ASCII format

The first line is a description line that must start with the word "solid" in lower case; it then normally contains the file name, author, date etc. The last line should be the keyword "endsolid". The lines between the above contain descriptions of 3 vertex facets including their normals, the ordering of the vertices should comply with the right hand rule.
The syntax for an ASCII STL file is as follows:

```
solid [name_of_object]
  facet normal x y z
    outer loop
      vertex x y z
      vertex x y z
      vertex x y z
    endloop
  endfacet
  facet normal x y z
    outer loop
      vertex x y z
      vertex x y z
      vertex x y z
    endloop
  endfacet
...
endsolid name_of_object
```

Normal vector components and vertex coordinate data are written in scientific notation (+-d.ddddddE+-ee). Often the normal's need not be provided and they will be generated by the parsing software/system. The main restriction placed upon the facets in STL files is that all adjacent facets must share two common vertices (figure 1).

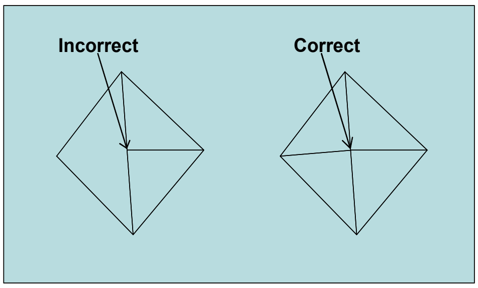


Fig. 1. STL representation restriction.

As an example consider the following except from a STL file

```
solid
 facet normal  0.000000e+000 9.971213e-001 -7.582376e-002
   outer loop
         vertex 7.293332e+002 2.200000e+002 1.183396e+003
         vertex 7.295713e+002 2.200000e+002 1.183396e+003
         vertex 7.295713e+002 2.190000e+002 1.170246e+003
       endloop
 endfacet
 facet normal  5.202612e-003 9.971148e-001 -7.572907e-002
       outer loop
         vertex 7.295713e+002 2.190000e+002 1.170246e+003
         vertex 7.293332e+002 2.190000e+002 1.170229e+003
         vertex 7.293332e+002 2.200000e+002 1.183396e+003
       endloop
 endfacet
 facet normal  0.000000e+000 9.971076e-001 -7.600333e-002
   outer loop
     vertex 7.295713e+002 2.200000e+002 1.183396e+003
     vertex 7.298094e+002 2.200000e+002 1.183396e+003
     vertex 7.298094e+002 2.190000e+002 1.170277e+003
   endloop
 endfacet
endsolid
```

b) BINARY format

Binary STL files consist of an 80 byte header line that can be interpreted as a comment string. The following 4 bytes interpreted as a long integer give the total number of facets. What follows is a normal and 3 vertices for each facet, each coordinate represented as a 4 byte floating point number (12 bytes in all). There is a 2 byte spacer between each facet. The result is that each facet is represented by 50 bytes, 12 for the normal, 36 for the 3 vertices, and 2 for the spacer.

```
<STL file> := <name> <facet number> <facet 1> <facet 2> ... <facet n>
<name> := 80 bytes file name, filled with blank
<facet number> := 4 bytes long int integer
<facet> := <normal> <vertex 1> <vertex 2> <vertex 3> <fill-bytes>
<normal> := Nx, Ny, Nz
<vertex> := X Y Z
<fill-bytes> := 2 fill bytes
```

3. Extracting relevant information from STL file

After compare the different types of neutral file formats in this work, the STL format was chosen because it presents the following advantages:

- The information about the 3D coordinates of the points that compose the object it is easily extracted;

- The dimensions of the object for using in the Off-line programming it is easily imported from the STL files;
- The loss of information about the layers, colours, and other attributes in the export process for the STL format don't bring significant losses for this specific application;
- The main commercially CAD tools export the information in STL format.

3.1 Implemented Algorithms for STL file reading

Extract 3D coordinates data from ASCII STL files is very easy like was described at the section 2.4 in this chapter. The easy handling syntax used to access its files stimulates it choice. Read binary file information is also usual, attempting to the correct size of byte arrays used in blocks data reading (section 2.4 – b)).
Two algorithms will be shown exhibiting a suggestion to reading data implementation using pseudo language.

Reading Binary STL file:

```
Open File
To reserve 80 bytes array to header reading
To reserve 4 bytes array to facets number reading
To read header
To read facets number
While not EOF
    Read normal vector float xyz coordinates
    While not end of Triangle
        Read triangle vertexes float xyz coordinates
    EndW
EndW
Close File
```

Reading ASCII STL file:

```
Open File
To read line
While read line is not "endsolid"
    If read line is equal “vertex”
        While not End of Triangle
            Read triangle vertexes
        EndW
    EndIf
To read line
EndW
Close File
```

4. Used Project Techniques

Abstraction is a very important feature when thinking about industrial machine programming. Code abstraction makes possible generalize implementation reaching high code re-use and less memory allocation. This abstract programming way makes feasible and easy, merge code from isolated software packages to new and specialized applications fields. With an abstract modelling of basic mechanisms structures like: joints, axes, programs and points for example. It can facilitate the routines implementation and strings manipulation interaction. To compose motion commands through many languages that will run over different robot controllers. At this section will be explained how programming techniques helps to extrapolate these structures and organize sequentially machine control commands.

4.1 Design Patterns

Every object oriented effective architecture is full of patterns. Because to use design patterns during the development will provide a smaller, simplest and maintainable architecture, when compared with other paradigms (Gamma et al., 1998).
Experience is an evaluated characteristic in every kind of business. Also in oriented object programming (OOP) is useful adopt renamed programmers successful experiments and relates, called Design Patterns to solve common implementation problems.
Code reusability is one of the benefits brought by OOP. But, projecting reusable object oriented software is a hard work. For that reason specific development problems are targeted by object oriented design patterns bringing flexibility, elegance and code recycle. Resuming, design patterns (DPs) is a high level mechanism that relates the best programmers' practices (successful experiences) to solve software projects common problems. The DPs used at present work was: Factory, Singleton, Facade, Model/View/Controller, Memento, Command and Template Method (Gamma et al., 1998). Next we will discuss this OOP techniques applied in the robot code generator development, his labels, it micro-architecture, common problems solved by this practices, and implantation consequences.

4.1.1 Factory

It can be understood like an objects factory. This pattern centralizes the objects creation with high changing probability in the project. The demand of a class that centralize the objects creation was the motivation to idealize it. Prohibiting object directly instance inside of business classes. The source problem is: if a signature method changes in a specific class, it will be necessary change all of direct instances to it class, becoming hard and complex work, considering that this object is being instanced by many classes. The proposed solution was developing a centralized solution class, responsible to create and return instance references from all called objects. If some object charger method changes, in the factory class, it will be centralized in the getInstanceX() method, where "X" is the name of a class that is moulding the object identity.
The strategy was analyze the list of classes that has an big probability to change, and implements its getInstance() methods. All GetInstance() methods implements the objects creation inside of a centralized factory class. Figure 2 shows the class Program requesting through Factory a reference to new Point object instance.

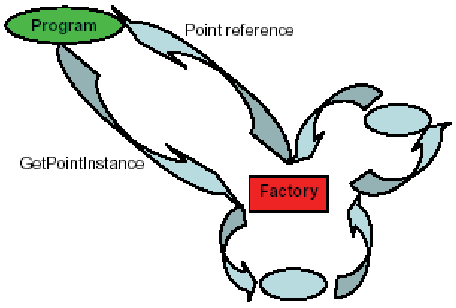


Fig. 2. Point instance request.

4.1.2 Singleton

It is a pattern that prohibits the duplication of objects in RAM memory. Your meaning is "unique instance". It was idealized because during the program execution many objects are recreated without real requirement, reflecting in memory and processing overhead. The main problem attacked by it DP is; the memory wastefulness caused by the sub-utilization of created objects in memory. It will decrease the overhead caused by java virtual machine garbage collector service, that monitories and cleans the unreferenced objects in memory.
The proposed solution was, to verify during object creation if it still exists in memory. The direct consequences to it practice: decreases process overhead caused by garbage collector and increase the objects life cycle, reflecting in a better memory resources allocation. The follow strategy was represented in figure 3.

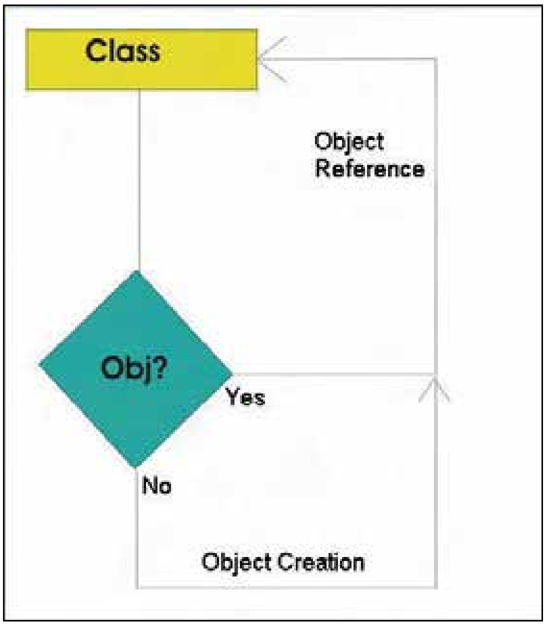


Fig. 3. If the object still exists in memory is returned a reference to him, else the object is created and also is returned a pointer to it new object.

4.1.3 Facade

The facade pattern decouple user interface and business classes, through an acknowledged default data input point. The meaning of facade can be understood like "consensus". It was implemented to make possible connect different interfaces with the same business classes, through a centralized data input, decoupling interface layer from business layer. That problem is: if the interface changes, also will be necessary changes the businesses classes to compatible it. This practicum suggests the creation of a class that hide interface layer system complexity and allows interaction with any kind of input, like command prompts, database queries, applets or encoded data input applications. The direct consequences will be decrease interface interaction complexity and decouple interface from business layer.

Thus all dependencies between involved entities in the use cases will be transparent to the user. Another visible gain is, the software maintenance becomes easier. Reflect in fast interface modifying and new interface input creation.
The strategy was implements a charger class, that knows how to talk with the business layer and known by interfaces that want to use the service. This way will make possible change the presentation layer without big software set adjustments. Figure 4 shows how charger class, talks with business layer, and it data inputs centralized model.

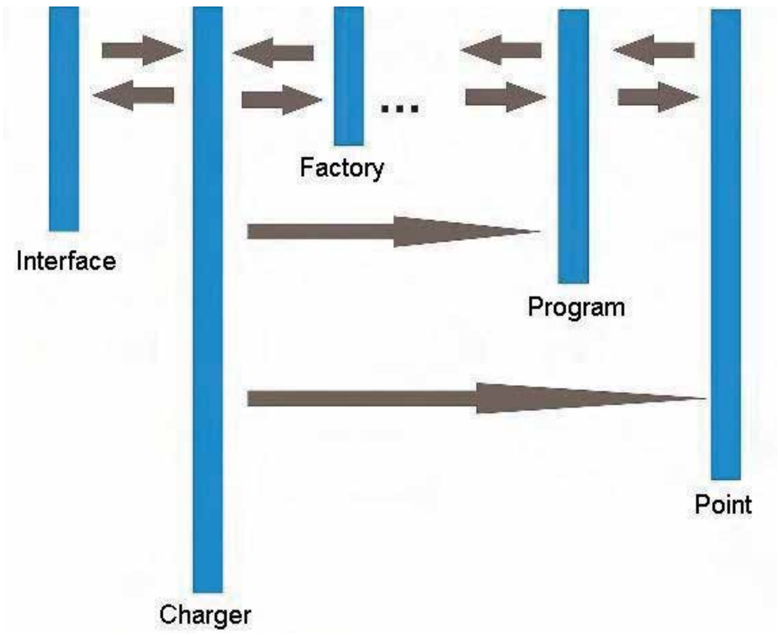


Fig. 4. The charger intermediation.

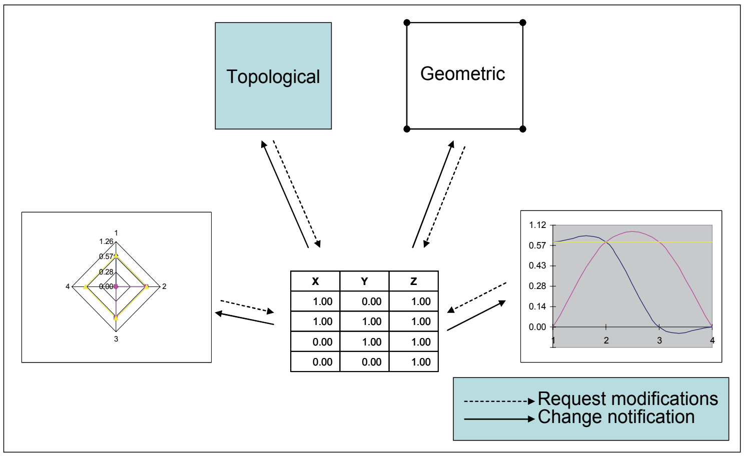
4.1.4 MVC

The triple MVC is an acronym to Model/View/Controller. It is used to develop object oriented interface keeping consistence between model and views through the controller.
This pattern is composed by 3 object types basically:

- The model that is the application object.
- The View that is the graphic representation.
- The Controller, who defines the interface behaviour reacting a data input.

Before MVC pattern creation this functionalities were grouped in a singular object. This pattern separates views from model establishing a subscription/notification protocol between them. The view has to reflect the model state, and if the model changes else all dependent views have to be notified. The main target of this pattern is link an unrestricted number of views to a model, bringing capability of many different data representations. It able inserts new views, without huge code modifications.
The aim of this pattern is separate data (Model) from user interface (View) and application flow (Controller). The gain of this practice is share the same business logic to be accessed through different interfaces, like is pointed at figure 5.
In MVC architecture, the Model does not know how much Views are showing its state. This pattern possibly append easily new human interface technologies to a specific solution, like

holograms for example, only attaching the needed hardware and adding the minimum code required to access the model behaviour.
Interactive applications require dynamic interfaces. Therefore reducing the coupling between interface and business classes will reflect in less development effort and easiest interface maintenance to new versions with new functionalities.

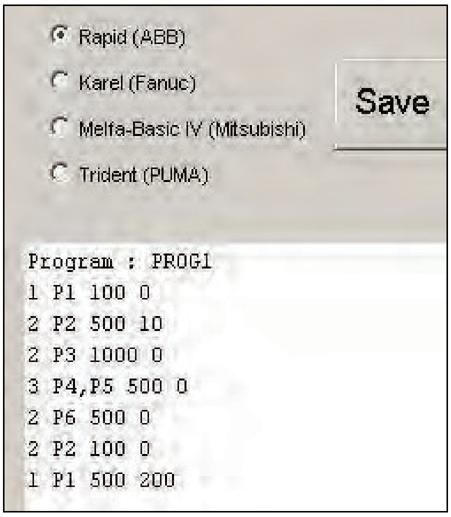


X	Y	Z
1.00	0.00	1.00
1.00	1.00	1.00
0.00	1.00	1.00
0.00	0.00	1.00

Fig. 5. Model View Controler Iteraction.

4.1.5 Memento

It is a pattern that stores the last recent memory interactions with the software, making possible recoup effected operations. Can be understood like "Undo". This demand was perceived when the user, seeking a program creation though line needs to undo some interactions in the code. The problem detected is: how to store the working structures? The proposed solution was store the most recently objects states in a limited size list, to restore it if necessary. Thus the user can undo a fix number of executed steps. The main objective of this pattern is creating a user-friendly interface.



[✓ Rapid (ABB)] Karel (Fanuc) Melfa-Basic IV (Mitsubishi) Trident (PUMA)

```
Program : PROG1
1 P1 100 0
2 P2 500 10
2 P3 1000 0
3 P4,P5 500 0
2 P6 500 0
2 P2 100 0
1 P1 500 200
```

Fig. 6. Generic list of inserted motion commands.

The strategy is to create a collection that memorize the user steps, defining a structure that stores and restore the objects attributes, making it persistent in memory. This collection clones the list showed in figure 6, but keeping persistent the inserted/deleted motion commands to undo procedure.

4.1.6 Command

This pattern generates interface command decoupling actions of it causing events. The name of this pattern can give an idea of his functionality. Command pattern is necessary because the user is forced to execute system functionalities through data input interface objects(Menu, Button, CheckBox,...). The user interface generation tool has menus and buttons objects responsible for "commands" entered by the user. Action is responsibility of business classes. The user interface objects should not implement the explicit action, because only the business layer should know how to do it.
The solution adopted is showed in figure 7. All current interface instanced objects know only a reference to the method responsible to execute the user desired action. It decouples interface and business layer functions. The chosen strategy is; the interface objects only will know the system default input point (see Facade pattern, section 4.1.3). Thus interface objects invoked by the user, have to reference the method located in the business layer to execute actions, using the common input point, the charger class.

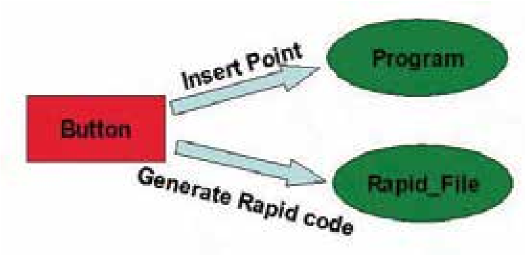


Fig. 7. Button commanding objects action.

4.1.7 Template Method

This pattern helps to define a skeleton algorithm. This skeleton will be used specializing subclasses that inherit the object abstract common model.

It allows a father class refinement from child classes realizing it reality showed in figure 8.

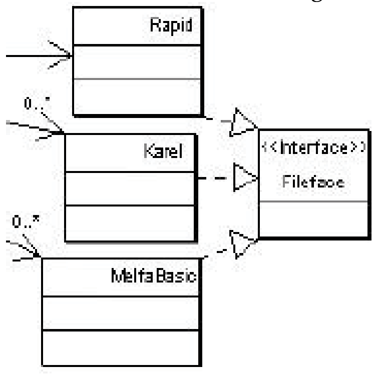


Fig. 8. Template method pattern used to persistence layer.

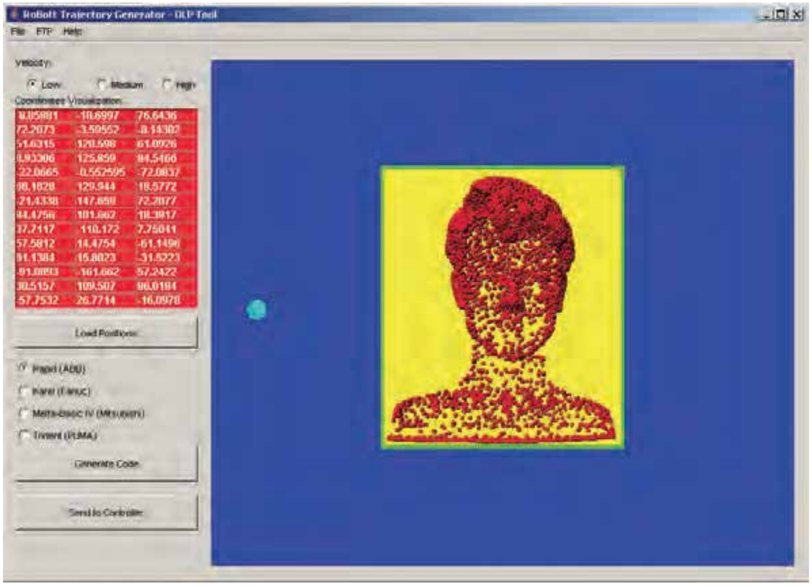
This technique consists in create a template to be specialized by child classes. It classes will materialize its behaviour overcharging father class methods through polymorphism. The TM direct consequence is increase code reuse. Although reflects in couple, caused by the implemented inheritance between the abstract class and child classes. This practice allied to dynamic binding is the bases to a framework building. The foundation of frameworks will be introduced in the next section.

4.2 Frameworks

It is a work layer where all knowledge about the predefined activity is encapsulated. Some characteristics of OO like dynamic binding, polymorphism, and inheritance make easy structures modelling it. These techniques were widely used to reusable, abstract, and specialized object creation. Make objects interact and communicate dynamically is the most important thing in the framework building. The framework concept was crucial to develop the robot code generator presented in this research, because it has to generate many different specific robot languages. Sometimes to implant the framework model is necessary redesign all the application to support its powerful interaction. Wherefore, the OODP and framework documentation is so important, prevents completely remodel the projected system if appear some located change demand, like insert a new brand robot language formulation.

4.3 RoBott

The RoBott Trajectory Generator (figure 9) was projected using OO concepts allied with UML specification to project analysis phase. Beyond it, all showed patterns concepts were considered in the software development phase, and applied using iterative incremental development process (Jacobson et al., 1998). The language used to develop this Off-line programming tool is Java Enterprise Edition (JEE) version 1.4.2_03, and the development environment is Oracle JDeveloper version 9.0.5.1(Build 1605).



Velocity: [✓ Low] Medium High

Coordinates Visualization:

8.85881	-10.6997	76.6436
72.2073	-3.59552	-8.14302
51.6315	120.598	61.0926
0.93306	125.859	84.5466
-22.0665	-0.552595	-72.0837
08.1828	129.944	18.5772
-21.4338	147.659	72.2077
44.4756	101.662	18.3817
37.7117	110.172	7.75041
57.5812	14.4754	-61.1496
91.1384	15.8023	-31.5223
-91.0893	-161.662	57.2422
30.5157	109.507	86.0184
-57.7532	26.7714	-16.0978

[✓ Rapid (ABB)]
Karel (Fanuc)
Melfa-Basic IV (Mitsubishi)
Trident (PUMA)

Fig. 9. RoBott OLP Tool.

5. Results

The tests were implemented using two different robot constructors, ABB and Mitsubishi industrial robots. The complete Off-line programming test was done using the ABB robot. First, it was necessary to setup references of ABB IRB140 working area (Murray, 1994), represented in figure 10 as a green rectangle. To get the robot real working area references is required to position the piece that will be worked. The blue point represents the robot base were is connected to the axis 1. After that, the neutral file has to be read to extract the outline of the piece, and it can be positioned as well using rotation and translation matrixes, respecting the working area bounds. After piece placement, the select layer (yellow rectangle) can be moved to touch the interesting points selecting the respective coordinates that will be used to generate the specific robot program. The file transfer to the ABB Robot Controller (IRC5) was done using FTP RoBott capability.

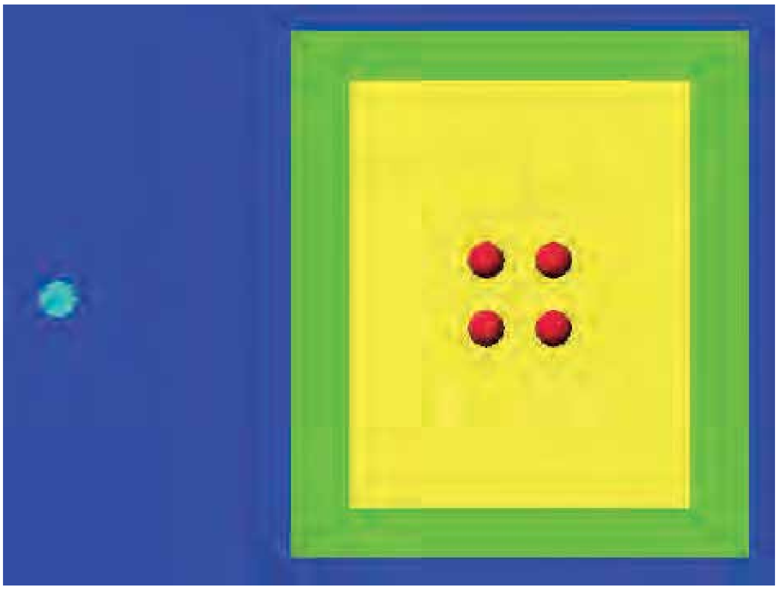


Fig. 10. Selected points from a top of a Cube placed in the ABB IRB140 work area.

After working area reference setup, cube boundary extraction from a STL file, and point cloud selection, it is possible generate a robot program (ABB, 2000) like the following example:

```
%%%
    VERSION:1
    LANGUAGE:ENGLISH
%%%
MODULE cubo
PERS robtarget Point0:=[[505.5,200.0,450.5], [0.043,0.384,0.921,0.021],…];
PERS robtarget Point1:=[[505.5,210.0,450.5], [0.043,0.384,0.921,0.021],…];
PERS robtarget Point2:=[[515.5,210.0,450.5], [0.043,0.384,0.921,0.021],…];
PERS robtarget Point3:=[[515.5,200.0,450.5],[0.043,0.384,0.921,0.021],…];
    PROC main()
     ConfL \Off;
```

```
        MoveL Point0,v10,fine,Tool0;
        MoveL Point1,v10,fine,Tool0;
        MoveL Point2,v10,fine,Tool0;
        MoveL Point3,v10,fine,Tool0;
        MoveL Point0,v10,fine,Tool0;
        ConfL \On;
   ENDPROC
  ENDMODULE
```

Some tests were done also with Mitsubishi Move Master Industrial Robot, but restricted only to Melfa Basic IV(Mitsubishi, 2000) motion commands generation. The 3D coordinates input to Mitsubishi platform was done by teaching.

6. Conclusion

Actually, the robot programming is still a hard work (Wrn, 1998). Some causes are: difficulty of available equipment reserve to improve it, complex handling, and technologic approach demanded to learn it. This research demonstrates that the manufacturing cell integration can be accelerated, the communication between different platforms of robots can be optimized and costs with specialized people can be reduced.
The off-line programming method was created to minimise the integration cell time. But the contemporary off-line programming has not brought significative gains to the manufacture cell integration, also to reduce the robot programmer working hours. The contemporary programming tools to manufacture cells were projected without the necessary abstraction, to generalize the robot programming problem. The available tools present in robot kits, can program and interact only with its platform, files and libraries.
The demand grows for a unified tool that interact between different manufacturers solutions, turning easy robot programming to the companies. The development focus creates portable software, capable to write robot programs to different manufacturer's languages. Hence actually, it research is able to extract reference coordinates from a neutral project CAD file, generate motion commands to different robot platforms, through different input interfaces, running over different operating systems.
Summarizing the process:

- The points cloud is extracted from a STL file
- The coordinates references are transformed by scaling, translations and rotations matrixes
- The interesting points are selected according the planned task
- The proprietary robot programs are generated
- The program is sent to the robot controller and runs.

So, the effective Off-line programming was tested successfully.

7. Future Research

The abstraction used through robot programming interface development allows it application to program many robot tasks. Practicing a right point selection in the points cloud, it is possible program tasks to welding, polishing, painting, assembly, inspection, every else manufacture area or robotized services. It is also possible insert new robot platforms with his respective proprietary languages.

The off-line robot programming can be used without restrictions to reduce manufacture cell integration time. It is also useful in areas where the contingent of workers was reduced by repeatable and insalubrious features tasks.

To future works we suggest specialized algorithms implementation using:

- Graphs theory to points selection refinement, task and path planning to:
  - Painting and Polishing using Smoothing paths
  - Welding using Waving paths
  - Or assembly and inspection using Template Matching
- Finite Elements theory (FEM) to calculate the force applied in the tool, working over a piece composed by some known material, like the sculpture process for example.
- Real time module receiving feedback from an extensometer network mounted between the tool and the robot flange, correcting in real time the tool attack angle and the tension applied to robot servo motors.

Acknowledgements. This research was carried out in the context of a MSc Degree in Industrial Electronic Engineering. Project Supported by the Programme ALBAN, the European Union Programme of High level Scholarships for Latin America, scholarship no. E04M033540BR. Thanks to CIMATEC - Technology and Manufactory Integrated Centre of SENAI-BA, Brazil, and Electronic Engineering Department (DEI), University of Minho, Portugal.

8. References

Gamma, E., Helm, R., Johnson, R. & Vlissides, J. (1995). *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison Wesley Longman, ISBN 0-201-63361-2, 1st edition.

Jacobson, I., Booch, G. & Rumbaugh, J.(1998). *The Unified Software Development Process*, Addison Wesley Longman, 1998.

Lee, D. M. A. & ElMaraghy, W. H. (1990). ROBOSIM: a CAD-based off-line programming and analysis system for robotic manipulators. *Computer-Aided Engineering Journal,* Vol. 7, No. 5, (October 1990) page numbers (141-148), ISSN: 0263-9327

Murray, R. M. (1994). A Mathemetival Introduction to Robotic Manipulation. *CRC Press, Florida*, 1st edition.

Owen, J. (1994). STEP : An Introduction. *Information Geometers Ltd,* (April 1994).

Schenck, D. A. & Wilson, P. R. (1994). Information Modeling: The EXPRESS way. *Oxford University Press,* (1994)

Smith, B. M., Rinaudot, G. R., Reed, K. A. & Wright, T. (1988). Initial Graphics Exchange Specification (IGES). *Version 4.0. IGES/PDES Organization,* (June 1988), Gaithersburg, MD.

Vuoskoski, J. (1996). Exchange of Product Data between CAD systems and a Physics Simulation Program. *Tampere University of Technology, Pori Unit,* (April 1996) page numbers (19-23).

Wrn, H. (1998). Automatic off-line programming and motion planning for industrial robots. *In ISR98, 29th International Symposium on Robotics 1998*. ISR Press.

ABB (2000). ABB Robotics AB, *RAPID Reference Manual* – Características Gerais, Västeras.2000, 172p (3HAC 5780-1).

IGES 1 (1980), NBS, US. Department of Commerce, Washington DC 20234, 1980.

Mitsubishi (2000). Mitsubishi Electronics Corporation. *Mitsubishi Industrial Robot – Instruction Manual, Detailed explanations of functions and operations*. Nagoya, 2000. 152p (BFP-A5992-C).

19

A Teaching-Free Robot System Utilizing Three-Dimensional CAD Product Data

Yoichi Nagao, Hideaki Ohta, Fumihiro Honda
Kawasaki Heavy Industries, Ltd.
Japan

1. Introduction

The recent shortage of experienced workers and the trend toward better working environments have accelerated the introduction of robots into manufacturing sites. This in turn has highlighted the need for more efficient teaching methods of robots. Conventionally, industrial robots are taught online. With this method, the production line had to be halted at every teaching sequence, which adversely affects production efficiency. More recently, off-line teaching has become common using computer-based robot models (Kobayashi et al., 2001). While off-line teaching does not stop the operation of the production line, teaching demand is intensive for the production lines where many different products are manufactured in small lots, leading to high operator workload and pushing up production costs (Nagao & Onoue, 2000).
The authors have been engaged in research into an automatic NC data generator named Kawasaki Common Off-line NC data Generator (KCONG) (Nagao et al., 2000a). KCONG generates action data for a robot based on the design information from a three-dimensional CAD unit without the need for any human intervention or teaching. Recently, the authors have developed some teaching-free robot systems by combining KCONG with welding, cutting, painting and other industrial robots (Nagao et al., 2000b).
This chapter introduces one such system, a teaching-free robot system for welding large box structures using KCONG, which was delivered to a client and is now operating successfully (Nagao et al., 2001).

2. System Overview

2.1 Applicable operations

The system is designed for large box structures more than 4 meters in length and to which many stiffeners are attached (see Fig. 1). Horizontal and vertical fillet welding and flat, vertical and horizontal butt-welding can be performed on square, single-bevel and flared grooves. Weld lines may be straight, curved or a combination. Boxing at the ends, connecting beads, applying multi-layer beads and continuous / intermittent welding are also possible.

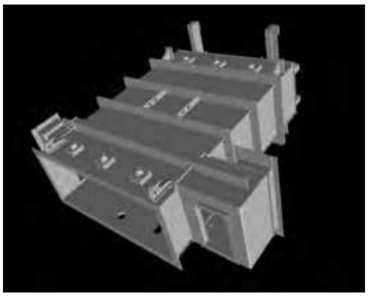


Fig. 1. A typical workpiece.

2.2 System configuration

The system consists of a three-dimensional CAD unit to design workpieces, the NC data generator KCONG, a line controller and a welding robot equipment. As shown in Fig. 2, the robot equipment consists of a three-axis moving unit — longitudinal, lateral and vertical — and a six-axis robot suspended perpendicularly from the moving unit. Since weld joints are located on both external and internal surfaces of the box, each workpiece is positioned in different orientations during welding: sideways, sideways but reversed, and upright. The robot enters the box to weld the internal surfaces.

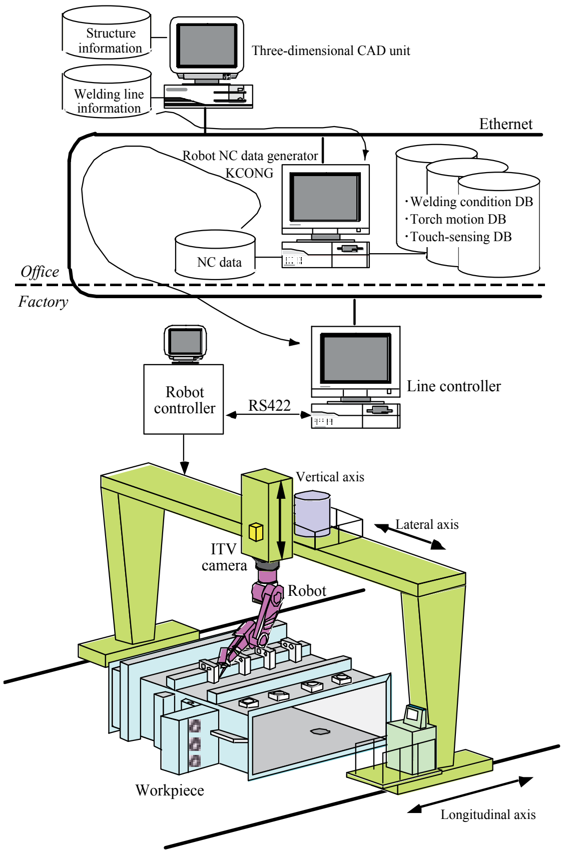


Fig. 2. Configuration of a teaching-free robot system.

Data flow through the system is as follows:

(1) Three-dimensional configuration data and welding-related data for the workpiece are output from the three-dimensional CAD unit. Configuration data is output in a general-purpose CAD format (VRML). Welding-related data includes weld line positions, types of grooves, welding conditions, torch orientations and the positions of workpiece reference points.

(2) KCONG reads the configuration data and welding-related data from the CAD unit and generates NC data by referring to its databases. Calculations are all performed automatically to determine the robot orientation, check for interference between the robot and the workpiece, and plan the welding sequence.

(3) The NC data generated by KCONG is sent to the line controller via a LAN.

(4) The line controller measures the position of the workpiece using an ITV camera installed on the moving unit. The NC data is corrected based on the measurement results and the corrected data is then sent to the robot controller, which generates a weld execution command to the robot.

3. NC Data Generator: KCONG

The NC data generator KCONG generates NC data for the robot, as shown in Fig. 3.

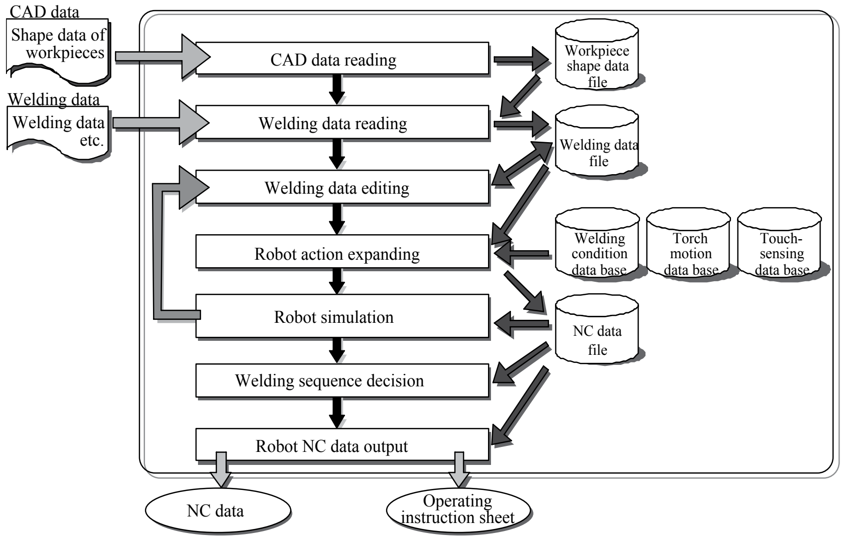


Fig. 3. The procedure for generating NC data with KCONG.

3.1 Reading CAD data

After reading the configuration and welding-related information for the workpiece from the three-dimensional CAD unit, the generator KCONG performs the following tasks:

(1) ***Shortening weld lines*:** If a weld line is detected to be out of the range in relation to the weld area due to the shapes of the parts to be welded, it is shortened.

(2) ***Dividing weld lines*:** If there is a point on a weld line where welding has to be stopped due to the presence of one or more crossing members, the weld line is divided into shorter lines as required.

(3) ***Concatenating weld lines*:** To minimize the air cut time, weld lines are concatenated where possible.

When the above tasks are completed, the workpiece configuration and weld lines are displayed on the screen three-dimensionally. It is possible to zoom in and out of the image on the display, and change the viewpoint as needed.

3.2 Generating NC data

When creation of NC data is requested, KCONG generates it as follows:

(1) ***Determining the torch orientation*:** The torch orientation is determined for each weld line based on the conditions surrounding the line, and the position of the base (root) of the robot and its configuration are determined from the torch motion database.

(2) ***Determining welding areas*:** Simulation is carried out at small intervals along each weld line to check if there is interference between the robot and the workpiece. If there is interference, KCONG returns to step (1) to select a different torch orientation; this process is repeated until the interference is resolved. If interference persists, the torch orientation that minimizes the unwelded area is chosen. In Fig. 4, part B near weld line PQ and part C that is attached to part A interfere with the welding torch, making welding impossible in two areas.

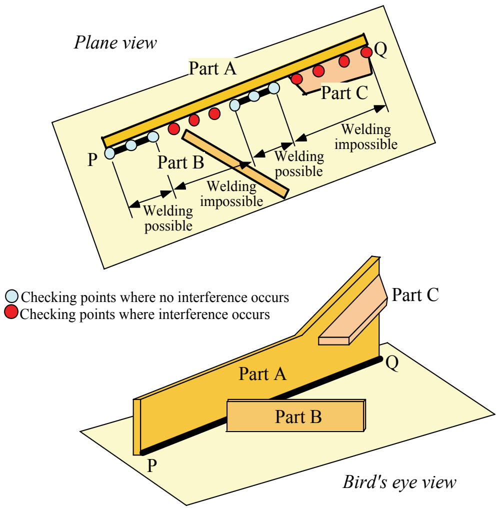


Fig. 4. Determination of welding performable area.

(3) ***Determining the touch-sensing path*:** Touch-sensing paths that would assure accurate detection of parts without interference are determined from the touch-sensing path database (see Fig. 5) and the surrounding conditions at the ends of the weld lines. For example, in Fig. 6, there is a hole in the area where the welding wire should touch. KCONG acquires this information and modifies the sensing path, from solid lines (1) to (3) according to (i, j) = (0, 0), to broken lines (1) to (3) according to (i, j) = (20, 0) in the touch-sensing database.

(4) ***Determining approach / departure paths*:** KCONG determines the interference-free approach and departure paths for the chosen torch orientation by referring to its databases and by conducting simulations.

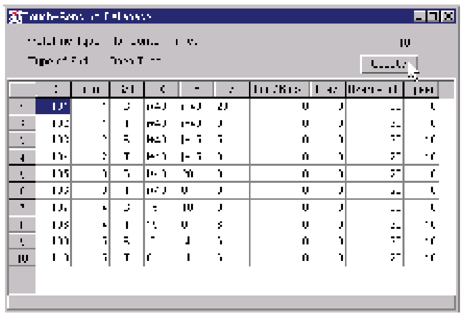


Fig. 5. An editing window of the touch-sensing database.

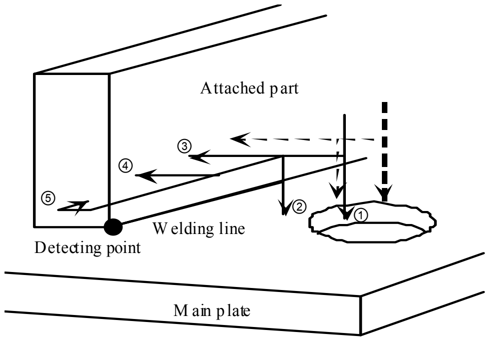


Fig. 6. Automatic selection of a touch-sensing path.

(5) ***Simulating entire welding procedure*:** After acquiring from the databases the welding conditions and torch motion at the ends of the weld lines, KCONG simulates the approach, touch-sensing, torch motion at the ends, regular welding, and departure in order to verify the NC data that has been just created. See Fig. 7 for an example simulation display.

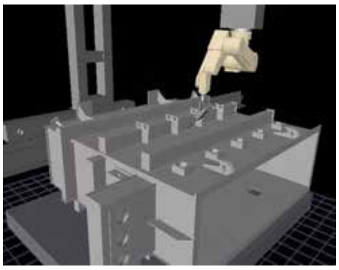
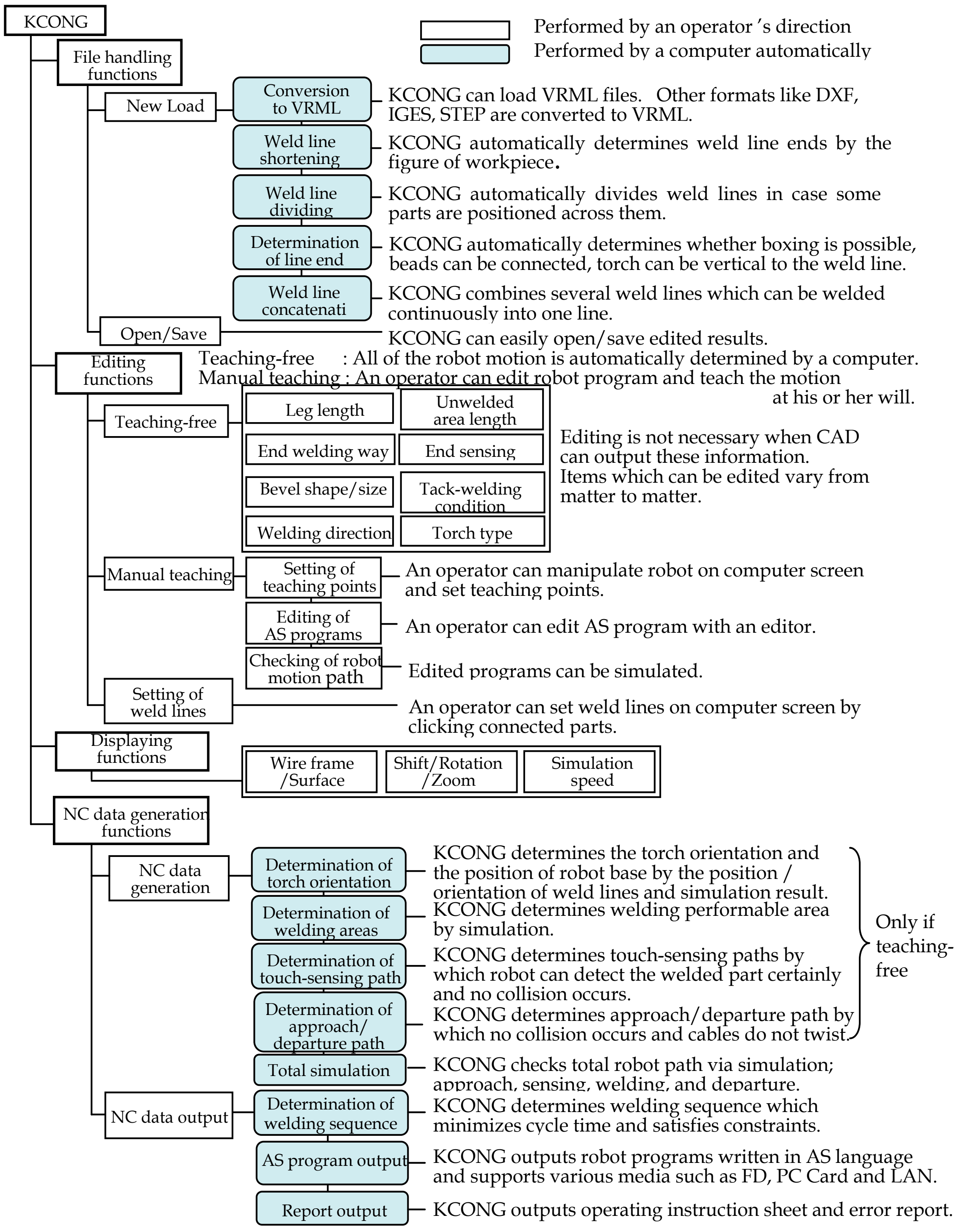


Fig. 7. A graphical display for the robot simulator.

At this point, the robot NC data has been generated. The operator can modify the data in the databases as necessary: welding conditions, touch-sensing paths, and torch motion at the ends of the weld lines. Even if automatic NC data generation is not possible, an operator can create a robot action program by directly teaching the computed robot model.

3.3 Outputting NC data

When NC data output is requested, KCONG outputs the NC data in the following manner:



KCONG
Performed by an operator 's direction
Performed by a computer automatically
File handling functions
New Load
Conversion to VRML
KCONG can load VRML files. Other formats like DXF, IGES, STEP are converted to VRML.
Weld line shortening
KCONG automatically determines weld line ends by the figure of workpiece.
Weld line dividing
KCONG automatically divides weld lines in case some parts are positioned across them.
Determination of line end
KCONG automatically determines whether boxing is possible, beads can be connected, torch can be vertical to the weld line.
Weld line concatenati
KCONG combines several weld lines which can be welded continuously into one line.
Open/Save
KCONG can easily open/save edited results.
Editing functions
Teaching-free : All of the robot motion is automatically determined by a computer.
Manual teaching : An operator can edit robot program and teach the motion at his or her will.
Teaching-free
Leg length
Unwelded area length
End welding way
End sensing
Bevel shape/size
Tack-welding condition
Welding direction
Torch type
Editing is not necessary when CAD can output these information. Items which can be edited vary from matter to matter.
Manual teaching
Setting of teaching points
An operator can manipulate robot on computer screen and set teaching points.
Editing of AS programs
An operator can edit AS program with an editor.
Checking of robot motion path
Edited programs can be simulated.
Setting of weld lines
An operator can set weld lines on computer screen by clicking connected parts.
Displaying functions
Wire frame /Surface
Shift/Rotation /Zoom
Simulation speed
NC data generation functions
NC data generation
Determination of torch orientation
KCONG determines the torch orientation and the position of robot base by the position / orientation of weld lines and simulation result.
Determination of welding areas
KCONG determines welding performable area by simulation.
Determination of touch-sensing path
KCONG determines touch-sensing paths by which robot can detect the welded part certainly and no collision occurs.
Determination of approach/ departure path
KCONG determines approach/departure path by which no collision occurs and cables do not twist.
Only if teaching-free
Total simulation
KCONG checks total robot path via simulation; approach, sensing, welding, and departure.
NC data output
Determination of welding sequence
KCONG determines welding sequence which minimizes cycle time and satisfies constraints.
AS program output
KCONG outputs robot programs written in AS language and supports various media such as FD, PC Card and LAN.
Report output
KCONG outputs operating instruction sheet and error report.

Fig. 8. Function tree of KCONG.

(1) ***Determining the welding sequence*:** The welding sequence is determined so that all the preset conditions are fulfilled and the cycle time is minimized.
(2) ***Outputting the NC data file*:** The NC data is output as a file in the robot language format. The file is then transferred to the line controller via the LAN.
(3) ***Printing the operating instruction sheet and error report*:** Instructions for the operator, error reports and operating time estimates are printed.

The function tree of KCONG is shown in Fig. 8.

4. Line Controller

The line controller controls the entire system. It has the following five functions:

(1) ***Selecting the correct data for the workpiece*:** When a workpiece is fed onto the operation stage, the line controller selects and loads the appropriate data for that workpiece from the NC data created by KCONG.
(2) ***Teaching the reference points*:** This function teaches the robot to measure the three reference points on the workpiece set by the three-dimensional CAD unit. The process starts by aligning the crosshairs of the ITV camera installed on the moving unit with the reference points, thus measuring the workpiece placement in the longitudinal and lateral directions. The touch-sensing function then enables automatic measurement of the reference points in the vertical direction.
(3) ***Controlling sequence*:** This is the function that sequentially executes welding for each weld line on the workpiece. After correcting the NC data for the first weld line based on the measured reference point positions, the line controller sends the data to the robot controller with the welding start command. While the first weld line is being worked, the NC data for the next weld line is sent to the robot controller to eliminate any wait.
(4) ***Troubleshooting*:** If a problem arises while a weld line is being worked, the sequence for that weld line or for the entire workpiece is skipped.
(5) ***Monitoring*:** The line controller monitors the operation of the robot for any errors that might occur; the status of the robot is displayed in real-time on the screen as it operates.

5. Welding Robot Equipment

The robot used is the Kawasaki Arc Welding Robot JA-10. The gantry-type moving unit has three axes: longitudinal (stroke: 8.5 meters), lateral (4.3 meters) and vertical (2.4 meters). The six-axis welding robot is hung from the elevating section of the unit. Fig. 9 shows a general view of the welding robot equipment.

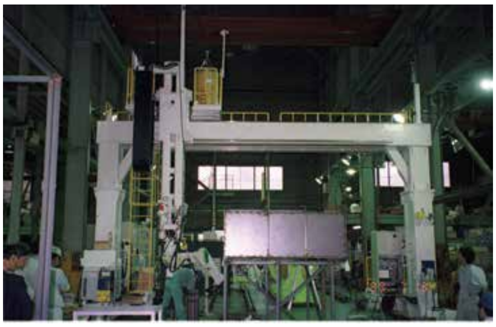


Fig. 9. General view of the welding robot equipment.

To increase the rate of automated welding for complex workpieces, the robot has three types of welding torches: straight, angled at 22°, and angled at 45°. They can be changed remotely from the line controller. The robot is also equipped with a high-voltage touch-sensing unit, a nozzle cleaner and a wire cutter. A wire reel is mounted on the lateral table of the moving unit.

6. Conclusion

A welding robot system for large box structures has been developed and put to practical use to automate welding and to reduce labor requirements. The system automatically generates the robot operation data based on the design information from a three-dimensional CAD unit without the need for teaching by humans. This system was delivered to a client in September, 1999, since which time the system has been operating smoothly, allowing the factory to automate more than 80% of its welding work.

In order to save the effort of preparation of welding information on a three-dimensional CAD unit, the authors have already implemented KCONG on Solidworks, a mechanical three-dimensional CAD. This new version has a function of definition of welding paths with which operators can prepare welding information in a simple operation. The authors are now developing redundancy control function which automatically configures the position and orientation of a robot base and a positioning unit, so as to move in a favorable posture.

7. References

Kobayashi, M.; Ogawa, S. & Koike, M. (2001). Off-Line Teaching System of a Robot Cell for Steel Pipe Processing, *Advanced Robotics*, Vol.15, No.3, (2001) (327-331).

Nagao, Y. & Onoue, K. (2000). Teaching Method and Open Architecture Controller for Industrial Robots, *Journal of the Robotics Society of Japan*, Vol.18, No.4, (May 2000) (24-28), ISSN0289-1824.

Nagao, Y.; Urabe, H.; Honda, F. & Kawabata, J. (2000a). Benifits of the Autoamted Robot Off-line Teaching System: KCONG, *The Structural Technology*, Vol.13, No.145, (June 2000) (44-48), ISSN0916-0426.

Nagao, Y.; Urabe, H.; Honda, F. & Kawabata, J. (2000b). Development of a Panel Welding Robot System for Subassembly in Shipbuilding Utilizing a Two-Dimensional CAD System, *Advanced Robotics*, Vol.14, No.5, (2000) (333-336).

Nagao, Y. ; Urabe, H. ; Honda, F. & Kawabata, J. (2001). Development of a Teaching-free Robot System for Welding Large Box Structures, *Proceedings of the 32nd International Symposium on Robotics*, pp. 971-976, Seoul, April 2001.

A Non-Contact Method for Detecting On-Line Industrial Robot Position Errors

Gregory C. Smith
AI Sensors
USA

1. Introduction

To be useful, industrial robots must meet positioning accuracy requirements for their given applications. Most industrial robots can return repeatedly to the same location in space quite precisely; they typically meet published repeatability specifications on the order of 0.5 mm. On the other hand, most industrial robots cannot move as precisely to a specified (x, y, z) position in space; they typically meet published accuracy specifications roughly an order of magnitude higher (typically 10 mm or worse) (Owens, 1994).

In many cases, published repeatability specifications meet positioning accuracy needs in industrial robot applications such as spot welding, spray painting, and assembly. However, published positioning accuracy specifications often do not meet industry needs, when using off-line programming rather than manual teaching methods. As a result, to meet application positioning accuracy requirements, most robot users turn to off-line calibration to bring positioning accuracy close to robot repeatability levels (Owens, 1994). Off-line calibration generally consists of the following five steps:

1. Move the robot into several poses (positions and orientations).
2. Measure and record the precise 3D workspace coordinates of the robot tool center point (TCP) at each pose.
3. Read and record the corresponding position of the robot, from the robot controller, at each pose.
4. Use the differences between measured 3D workspace coordinates and corresponding positions read from the robot controller to correct the parameters in the kinematic model used by the controller to position the robot.
5. During robot operation, use the corrected kinematic model to compute adjusted positions in space and then command the robot to move to the adjusted positions (which causes the robot to move to the actual desired positions).

The number of calibration poses used and the corresponding link positions for each pose must be selected to provide the information needed to accurately compute the kinematic model parameters (Robinson, Orzechowski, James, & Smith, 1997). For example, Owens (1994) used 25 different poses, while Rocadas and McMaster (1997) used 50 different poses.

To measure pose positions precisely enough to complete off-line calibration, robot manufacturers generally use expensive measurement devices, such as theodolites, coordinate measurement machines, or laser tracking systems (Mayer & Parker, 1994; Nakamura, Itaya, Yamamoto, &

Koyama, 1995; Owens, 1994). Such systems generally cannot be used for recalibrating robots in factory environments, due to cost and space limitations. However, recalibration may be needed after robot repair, collisions with the workpiece or other objects in the workspace environment, or over time as encoders or servo systems drift (Owens, 1994).

As a result, prior research offers many low-cost systems for calibrating robots off-line within factory environments. Low-cost methods for measuring robot position during calibration include cables (Owens, 1994), cameras (van Albada, Lagerberg, & Visser, 1994), dial gauges (Xu & Mills, 1999), and trigger probes with constraint planes (Zhong & Lewis, 1995).

After off-line calibration, industrial robots, run open-loop without additional control or intervention, have met the in-process accuracy needs of most current industrial applications (spot welding, material handling, workpiece handling, and assembly).

When open-loop use of robots has not met a given industrial application's needs, closed-loop control or passive compliance has been used. For example, for arc welding, laser-based vision systems have been used to locate and track welding seams (Agapakis, Katz, Friedman, & Epstein, 1990). For assembly, passive compliance devices, such as remote center compliance (RCC) devices, have been used to align components for mating (Bruyninckx et al., 2001; Boubekri & Sherif, 1990).

However, on-line sources of robot position error have been largely ignored. Collisions with the workpiece or other objects in the workplace environment, encoder errors, or servo drift can cause robot position to drift out of specification, leading to product faults, scrap, machine damage, and additional costs. Without in-process monitoring, in-process robot position errors are generally not detected until product faults are detected during product inspection.

Generally, sensors and methods used for calibrating robots cannot be used for in-process monitoring, because the mechanisms interfere with in-process robot operation (e.g., cable measuring systems, pointers, and calibration plates) or do not work well during in-process operations. Thus, typically, the only counter measures currently used to prevent in-process errors are regularly scheduled robot recalibration or production line stops when product faults are detected in inspection. However, detecting product faults after they occur is costly. Regular calibration, when not needed, is also expensive. Shop-floor recalibration of a single robot can take up to six hours or more (Owens, 1994). The wasted manpower time spent is an unnecessary excess cost.

To achieve greater operational efficiencies, new non-invasive, non-contact methods for monitoring in-process robot position are needed (Caccavale & Walker, 1997). Early work focused on using model-based methods, with existing built-in robot joint position and velocity sensors, to detect errors with respect to ideal dynamic observer or parameter estimation models (Caccavale & Walker, 1997). However, modeling accuracy, sensor characteristics, computational loads, and response times can limit the sensitivity and usefulness of model-based methods (Visinsky et al., 1994).

As an alternative, optical methods, with external sensors, such as laser systems, cameras, or machine vision systems, have also been proposed (e.g., Dungern & Schmidt, 1986). However, optical systems can also suffer from severe limitations. Laser and optical sensors can be difficult to place, since their optical paths to robot end effectors are easily blocked by workpieces or parts of the robot. In addition, smoke or sparks from welding, or fluids in other manufacturing processes, can interfere with proper laser and optical sensor operation.

2. Purpose

To overcome the limitations of optical position monitoring methods, the investigator developed a simple, low-cost method for detecting in-process robot position errors, which uses a low-cost Doppler motion detector unit placed at one or more critical robot work positions. The small detector can be easily located near critical work positions. Detector position and line of sight are not critical, since the detector measures and generates electrical signals due to large-scale robot motion, rather than precise robot end effector position.

Motion signals from the motion detector unit (MDU) are monitored as a time series, and statistical quality control methods indicate when robot position drift or other process faults occur. When faults are detected, signals can be generated to halt the robot and trigger alarms. Alarms signal the need for robot service or recalibration. Halting the robot at the earliest sign of possible position errors can help prevent product faults, scrap, machine damage, and additional costs.

The method may be more robust, in industrial environments, than optical condition monitoring methods, since radar signals can penetrate smoke, light, and other optical contaminants.

3. Experimental Setup

Figure 1 shows the experimental setup used to develop and test the proposed position error detection method. A Seiko D-TRAN RT-2000 robot was used for testing. The Seiko robot has a cylindrical configuration with four axes R (radial), T (rotational), Z (vertical), and A (gripper rotation).

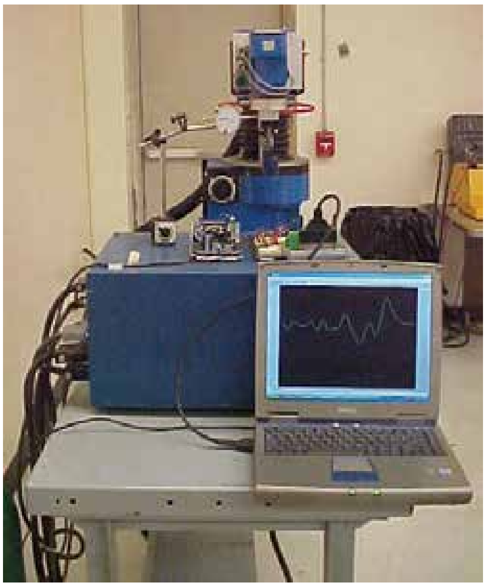


Fig. 1. Experimental setup.

Table 1 shows published repeatability and resolution specifications for each of the four robot axes. Robot reach in the R direction is 597 mm (23.5 in), maximum rotation about the T-axis is ± 145 degrees, stroke in the Z direction is 120 mm (4.72 in), and maximum rotation about the A-axis is ± 145 degrees.

The given robot controller stores a single calibration constant related to the fully extended length of the robot R axis. To calibrate the robot, the user must attach a rigid fixture, which has a precise length, to the robot and reset a stored calibration constant. Subsequently, on power-up the robot must be homed to recalibrate the robot TCP to the zero location of the workspace coordinate system. Homing moves the robot in each of the four axes to fixed limit switches and, thus, recalibrates the robot's internal servo encoders for the power-on position of the TCP zero point. Other commercial industrial robots use similar means to re-adjust the TCP to an accurate zero location. For example, Motoman, Inc. uses a special fixture (ToolSight) containing three LED sensors to re-center their welding robots (Forcinio, 1999).

Axis	**Repeatability**	**Resolution**
R	± 0.025 mm (0.001 in)	0.025 mm (0.001 in)
T	± 0.025 mm (0.001 in)	0.003 deg
Z	± 0.025 mm (0.001 in)	0.012 mm (0.0005 in)
A	± 0.025 mm (0.001 in)	0.005 deg

Table 1. Seiko D-TRAN RT-2000 repeatability and accuracy specifications.

For the experiments conducted, after robot homing, the robot was commanded to move from home position to a test point in the robot workspace coordinate system. As shown in Figure 2, a dial gauge, with a scale in English units, was used to accurately measure relative robot positions around the given test position. Figure 2 also shows the sensor circuit, composed of a Doppler radar motion detector and a low-pass filter, which was developed for measuring robot motion. The Doppler radar motion detector used was a model MDU 1620 Motion Detector Unit from Microwave Solutions (http://www.microwave-solutions.com).

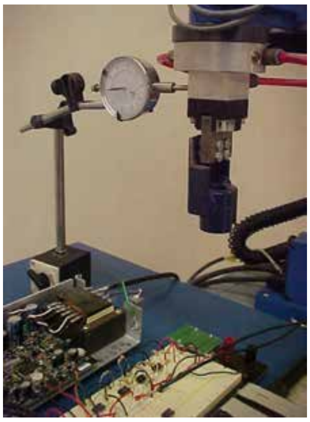


Fig. 2. Sensor circuit.

The MDU 1620 is an X-band (10.525 GHz) microwave transceiver that uses the Doppler shift phenomenon to "sense" motion (Microwave Solutions, 2002). The MDU 1620 Motion Detector Unit produces an intermediate frequency (IF) output signal with frequency proportional to the velocity of the moving object. IF output signal amplitude varies as a complex function of the size and reflectivity of the sensed object and the object's distance from the MDU (Microwave Solutions, 2002).

An Omega DaqP-308 data collection system was used to sample the output signal from the Doppler motion detector/low-pass filter combination. After each command issued to move the robot from

home position to the test position, the output signal from the Doppler motion detector/low-pass filter combination was measured for 4 seconds with a 0.1 msec sampling period (10 kHz sampling frequency). As a result, according to the Nyquist Theorem, the sampled data can be used to reconstruct frequency components up to 5 kHz in the original signal (Swanson, 2000).
To prevent aliasing during sampling, an electronic filter was used to band-limit the output signal from the Doppler radar motion detector before sampling, as shown in Figure 3. The low-pass filter uses an amplifier stage to increase motion detector output signal level, a fourth-order low-pass filter stage to band limit the measured signal and to eliminate high-frequency noise, and a final amplifier stage to match the output of the filter to the input range of the data collection system.

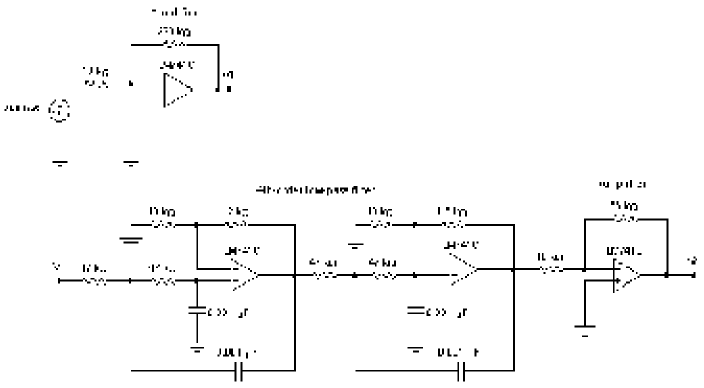


Fig. 3. Electronic filter.

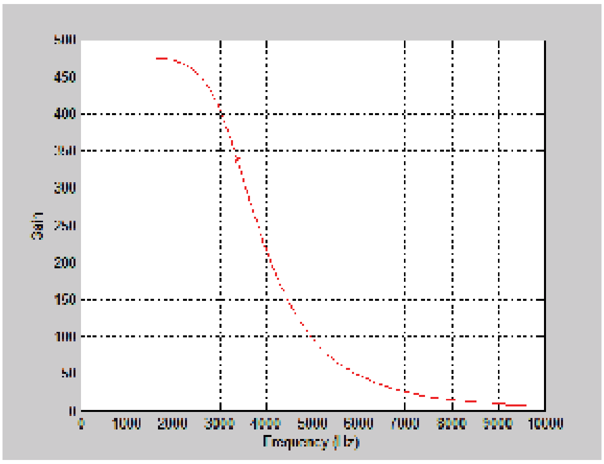


Fig. 4. Magnitude of electronic filter frequency response.

Figure 4 shows the theoretical frequency response of the filter. To meet the Nyquist sampling criterion, Swanson (2000) recommends using a band-limiting filter with an upper cut-off frequency which is roughly 0.4 times the sampling frequency. Thus, filter components were chosen such that the cut-off frequency of the fourth-order low-pass filter is 3.39 kHz (Millman & Halkias, 1972).
Captured data was analyzed using MathWorks Matlab (Version 6.5.0.1924 Release 13) and SAS JMP 5.

Five experiments were run to develop and test the proposed method for detecting on-line robot position errors in a single axis direction. Future studies will consider multi-axis robot position errors. Experiments 1 and 2 were run to verify that the robot used for testing met the manufacturer's published repeatability and resolution specifications, to verify that a dial gauge could be used to precisely measure robot position, and to characterize the drift characteristics of the robot over an extended period of cycling. Experiment 3 was run to develop a measure of robot position from sensor signals and to determine the precision of the sensor signal measure for robot moves to a single test position. Experiment 4 was run to establish a linear regression relationship between the robot position measure, which was developed in Experiment 3, and actual (induced) robot position errors. Experiment 5 was run to develop a robot position error detection model, from Experiment 3 and Experiment 4 results, and to test the prediction model for random robot moves about a single test position. The same experimental setup was used for all five experiments. The results of each experiment were used to adjust subsequent experiments, if needed. Since an incremental methodology was used, intermediate conclusions are reported with results from each experiment. Final conclusions are reported in the conclusions section at the end of the paper.

4. Experiment 1

The objectives of Experiment 1 were to:

1. Experimentally verify the repeatability of the Seiko D-TRAN RT-2000 robot used for testing,
2. Experimentally verify that a dial gauge can be used to precisely measure robot position, and
3. Experimentally determine if there is significant drift in the robot during cycling.

The method used to experimentally determine robot repeatability and drift characteristics consisted of five steps:

1. Command the robot to move to a test position 20 times,
2. Measure the position of the robot using a dial gauge,
3. Cycle the robot, between the workspace origin and the test position, for 3 hours,
4. Command the robot to move to the test position 20 times.
5. Measure the position of the robot using a dial gauge.

To simplify experimental setup and testing, the robot was moved to minimum Z-axis position, fully extended in the R-axis direction, and then commanded to move cyclically, in the T-axis direction only, between home position and a test point. The test point selected was with the robot at minimum Z-axis position, fully extended in the R-axis direction, and rotated to the 90-degree T-axis position. Minimum Z-axis position was selected to minimize distance between the sensor and the end effector, at the given test point. The fully-extended R-axis position was chosen to increase the potential for position errors, since, for the given robot, position accuracy decreases with distance from the workspace origin (Seiko, 1986). The 90-degree T-axis position was chosen so that the sensor could be mounted on the same table as the robot, to minimize potential sensor measurement errors due to robot vibrations.
Table 2 shows robot position dial gauge measurements taken at the test position before and after cycling the robot for 3 hours. Step 0, in Table 2, indicates the initial position to which the dial gauge was set, with the robot resting at the correct test position.
A one-way analysis of variance between the two groups of data shows that there is evidence of a statistically significant difference between the two group means ($\alpha = 0.05$, p-value < 0.001).

Step	Before Cycling (inches)	After Cycling (inches)
0	0.501	-
1	0.503	0.501
2	0.503	0.502
3	0.502	0.501
4	0.503	0.501
5	0.503	0.501
6	0.503	0.501
7	0.502	0.502
8	0.503	0.503
9	0.502	0.501
10	0.502	0.502
11	0.502	0.502
12	0.503	0.502
13	0.502	0.502
14	0.503	0.501
15	0.502	0.502
16	0.503	0.501
17	0.503	0.502
18	0.502	0.501
19	0.502	0.501
20	0.502	0.502

Table 2. Robot position dial gauge measurements for Experiment 1.

The mean for the Before Cycling group is 0.50243 inches (with a 95% confidence interval of 0.50216 – 0.50270 inches), and the mean for the After Cycling group is 0.501550 inches (with a 95% confidence interval of 0.50127 – 0.50183 inches). The difference between the two means is 0.00088 inches. With 95% confidence, a reasonable value for the difference between means lies between 0.00050 and 0.00126 inches. For the given robot, a reasonable value for the difference between robot position means, before and after 3 hours of cycling, lies between 0.00050 and 0.00126 inches. Experiment 1 results indicate that:

1. The robot appears to meet Seiko's published repeatability specification (0.025 mm or 0.001 inch), for measurements taken at a single time instance (before cycling or after cycling).
2. The dial gauge can be used to measure robot position precisely, (within approximately the robot repeatability specification).
3. There is evidence that the robot may drift slightly with extended cycling (3 hours). The upper limit of the 95% confidence interval for the difference between before cycling and after cycling means (0.00126 inches) is greater that the robot repeatability specification (0.001 inch), indicating that, with 95% confidence, a drift of 0.00026 inches beyond the robot repeatability specification could occur. As a result, an online method for detecting position errors might be useful for the given robot.

Since the time needed to induce a position error by cycling was relatively long, and since the magnitude of the error measured for Experiment 1 was relatively small compared to the robot repeatability specification, for Experiments 2-5, robot position errors were simulated by commanding the robot to move to positions slightly away from the test position.

5. Experiment 2

The objective of Experiment 2 was to:

1. Experimentally verify that the dial gauge could accurately detect single-axis robot position errors, for the given robot.

The method used to experimentally verify that the dial gauge could accurately detect single-axis position errors consisted of two steps:

1. Command the robot to move to the test position +/- 0.03 T-axis degrees, in 0.003 degree increments (the robot's T-axis accuracy specification is 0.003 degrees, which corresponds to 0.001 inches at the given test position).
2. Measure the position of the robot using the dial gauge.

Table 3 shows the 21 positions about the test point to which the Seiko D-TRAN RT-2000 robot was commanded to move (values in millimeters), as well as the corresponding dial gauge measurements (in inches). Note that the robot takes position commands as (x, y, z) Cartesian coordinate values, with (x, y, z) values in millimeters.

An analysis of variance shows evidence of a statistically significant relationship between dial gauge measurements and degree values ($\alpha = 0.05$, p-value < 0.0001).

Step	Position	X	Y	Dial Gauge
1	-89.970	0.313	-597.056	0.488
2	-89.973	0.281	-597.056	0.490
3	-89.976	0.250	-597.056	0.492
4	-89.979	0.219	-597.056	0.492
5	-89.982	0.188	-597.056	0.494
6	-89.985	0.156	-597.056	0.495
7	-89.988	0.125	-597.056	0.497
8	-89.991	0.094	-597.056	0.499
9	-89.994	0.063	-597.056	0.500
10	-89.997	0.031	-597.056	0.501
11	-90.000	0.000	-597.056	0.502
12	-90.003	-0.031	-597.056	0.504
13	-90.006	-0.063	-597.056	0.506
14	-90.009	-0.094	-597.056	0.507
15	-90.012	-0.123	-597.056	0.508
16	-90.015	-0.156	-597.056	0.509
17	-90.018	-0.188	-597.056	0.511
18	-90.021	-0.219	-597.056	0.513
19	-90.024	-0.250	-597.055	0.514
20	-90.027	-0.281	-597.055	0.515
21	-90.030	-0.313	-597.055	0.516

Table 3. Robot position dial gauge measurements for Experiment 2.

Equation 1 gives the equation of the least squares line shown in Figure 5:

$$\text{Predicted dial gauge value} = -41.69 - 0.4688 * \text{Robot position} \quad (1)$$

The model explains 99.74% of the variability in dial gauge measurements. Random measurement errors or other unexplained factors account for only a small amount of the observed variability in the data.

Experiment 2 results indicate that:

1. The Seiko D-TRAN RT-2000 robot appears to meet the published T-axis resolution specification (0.003 degrees). In other words, the robot can be accurately commanded to positions that differ by as little as 0.003 degrees.
2. The dial gauge can be used to detect given robot position errors to approximately the T-axis resolution specification.

Based upon Experiment 2 results, the given experimental setup was used for the remaining planned experiments.

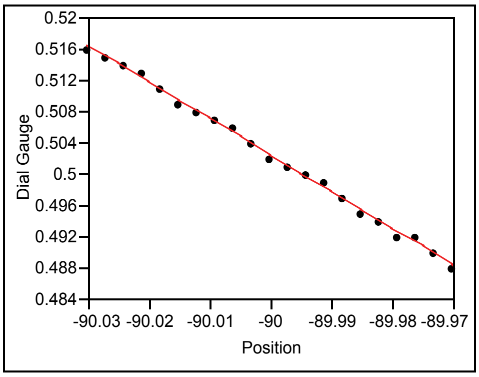


Fig. 5. Dial gauge measurements (inches) vs. robot T-axis position (degrees).

6. Experiment 3

The objectives of Experiment 3 were to:

1. Develop a measure from sensor signal samples for determining robot position,
2. Determine how well the sensor signal measure represents robot position,
3. Establish a mean signal to represent the robot moving to the correct test position, and
4. Calibrate the sensor for robot motions without position errors.

The method used to experimentally achieve Experiment 3 objectives consisted of seven steps:

1. Cycle the robot 20 times between home position and the nominal test position.
2. Measure robot position with the dial gauge.
3. Measure the sensor signal as the robot moves between home and the nominal test position. Sample the sensor signal at 0.1 msec intervals.
4. Average the values of the 20 sensor signals at each sampling time step to find the mean sensor signal value at each sampling time step.
5. Compute a root sum of squares error measure for each of the 20 sensor signals by summing squared error for each time sample with respect to the mean sensor signal value at each sampling time sample.
6. Compare standard deviation of the root sum of squares error measure for the 20 sensor signals to standard deviation of the 20 dial gauge readings.
7. Use the mean sensor signal and the standard deviation of the root sum of squares error measure as a sensor calibration standard for proper robot motion.

Figure 6 shows three representative sensor calibration signals, c_i and the mean calibration signal c_m. Figure 7 shows an expanded view of Figure 6 in the region near 2.5 seconds. The signals in Figures 6 and 7 were filtered, in Matlab, to remove any DC bias. The mean value of each signal was computed and subtracted from each of the signal's sample values.

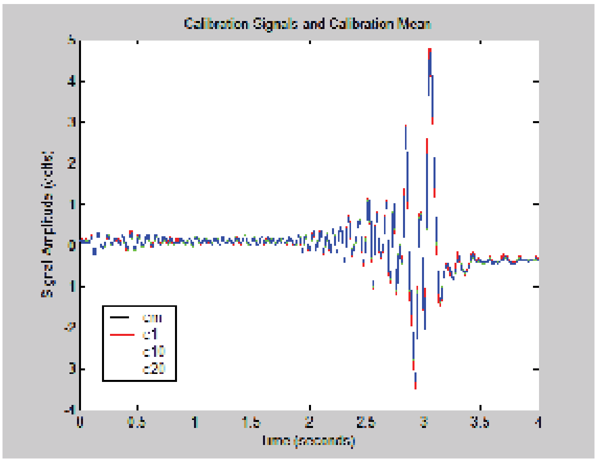


Fig. 6. Calibration signals and calibration mean.

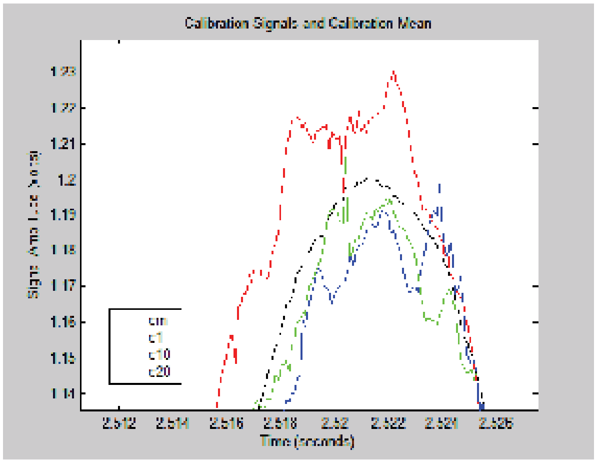


Fig. 7. Expanded view of Figure 6 near 2.5 seconds.

A frequency spectrum computed for calibration signal c_{10} shows that the sensor output signals for robot motion between home position and the test position are band limited to frequencies less than approximately 25 Hz. Therefore, the sampling period (0.1 msec) was more than adequate for accurately capturing signal content without aliasing.

To meet Experiment 3 objectives, each of the 20 filtered calibration signals were represented as an array of real numbers

$$c_i(n), \quad i = 1 \cdots 20;\ n = 1 \cdots 40000 \tag{2}$$

The mean value of all 20 signals at any given sample time step was calculated

$$c_m(n) = \frac{1}{20}\sum_{i=1}^{20} c_i(n) \tag{3}$$

As a measure of individual signal variation with respect to the mean of all 20 signals, a root-squared error value was computed for the 10,001 samples between 2.5 and 3.5 seconds

$$RSSc_i = \sqrt{\sum_{n=25000}^{35000} [c_i(n) - c_m(n)]^2} \tag{4}$$

The 10,001 samples between 2.5 and 3.5 seconds were used, rather than all 40,000 samples, to reduce computation time and to improve signal-to-noise ratio. Table 4 shows the 20 $RSSc_i$ measurements and the 20 corresponding dial gauge measurements taken for Experiment 3.

An analysis of variance indicates that there is no statistically significant relationship between $RSSc_i$ and dial gauge measurements (α = 0.05, p-value = 0.5462). The analysis of variance indicates that, with 95% confidence, variation in both measures is probably due to random measurement error. Mean for the 20 $RSSc_i$ is 2.98, and standard deviation for the 20 $RSSc_i$ is 2.01. Mean for the 20 dial gauge measurement is 0.50015, and standard deviation for the 20 dial gauge measurements is 0.00049.

Experiment 3 results indicate that:

1. The dial gauge is a more precise method for measuring robot position at the given test point.
2. However, if the sample of 20 calibration signals accurately represents the population of all sensor signals produced by the robot moving to the given test position, the $RSSc_i$ measure developed may be usable for non-invasive non-contact in-process robot position error detection, using statistical $\overline{X}$ control chart techniques.

For the $RSSc_i$ measure to be useable as an $\overline{X}$ chart quality measure, when errors occur, individual $RSSc_i$ measures, on average, must lie at least three standard deviations from the mean for the 20 sensor calibration signals (9.01 or larger) (Besterfield, 2001).

Signal	$RSSc_i$	**Dial Gauge**
c_1	11.2241	0.500
c_2	3.2244	0.499
c_3	2.7502	0.500
c_4	3.2171	0.500
c_5	2.9028	0.500
c_6	2.3334	0.500
c_7	1.9100	0.500
c_8	1.3153	0.501
c_9	1.9060	0.500
c_{10}	2.4277	0.500
c_{11}	2.1314	0.500
c_{12}	1.9294	0.500
c_{13}	2.1702	0.501
c_{14}	2.5620	0.500
c_{15}	2.5384	0.500
c_{16}	3.0110	0.501
c_{17}	2.9737	0.501
c_{18}	3.3289	0.500
c_{19}	2.7221	0.500
c_{20}	2.9593	0.500

Table 4. Error measures for calibration signals.

7. Experiment 4

The objectives of Experiment 4 were to:

1. Determine the feasibility of using sensor signals to detect in-process robot position errors, and
2. Experimentally establish a relationship between position errors and sensor signals.

The method used to achieve Experiment 4 objectives consisted of three steps:

1. Command the robot to move from the home position +/- 0.03 T-axis degrees to the test position +/- 0.03 T-axis degrees, in 0.003 degree increments (the robot's T-axis accuracy specification is 0.003 degrees).
2. Measure the position of the robot using a dial gauge.
3. Simultaneously measure the signal (e_i) generated by the sensor.

The robot was commanded to move from offset positions about the home position to offset positions about the test position to simulate on-line position errors that would occur due to collisions with the workpiece or other objects in the workplace environment, encoder errors, or servo drift.

Data collected from Experiment 3 and Experiment 4 was analyzed using statistical methods to establish a relationship between position errors and sensor signals. The resulting relationship was then used to detect or predict on-line robot position errors (Experiment 5).

The robot was commanded to move incrementally to 21 positions about, and including, the test position. For Experiment 4, due to the time required to collect and process collected data by hand, a single replication of the experiment was conducted. However, to determine the repeatability of Experiment 4 measurements, for Experiment 5, the robot was commanded to move to the same 21 positions, but in random, rather than incremental, order.

Figure 8 shows three representative sensor error signals, e_i and the mean sensor calibration signal c_m. Figure 9 shows an expanded view of Figure 8 in the region near 2.5 seconds. Table 5 shows the 21 positions from which the robot was commanded to move, the 21 positions to which the robot was commanded to move, and the corresponding final robot workspace x-coordinate values to which the robot was commanded to move. The final robot workspace y-coordinate values were the same for all 21 positions to which the robot was commanded to move (-597.056 mm). Table 5 also shows the 21 resulting $RSSe_i$ measurements and the 21 corresponding dial gauge measurements for Experiment 4.

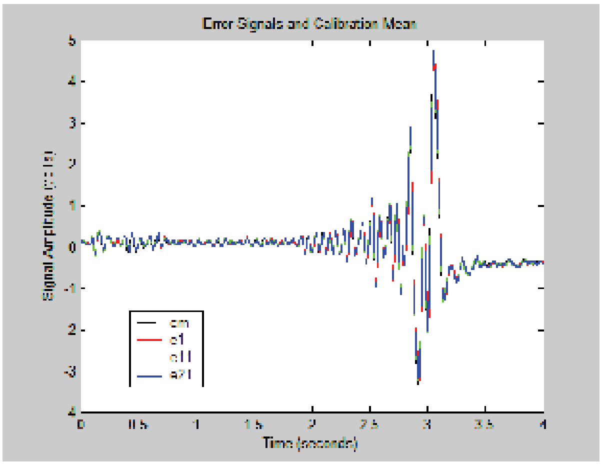


Fig. 8. Error signals and calibration mean.

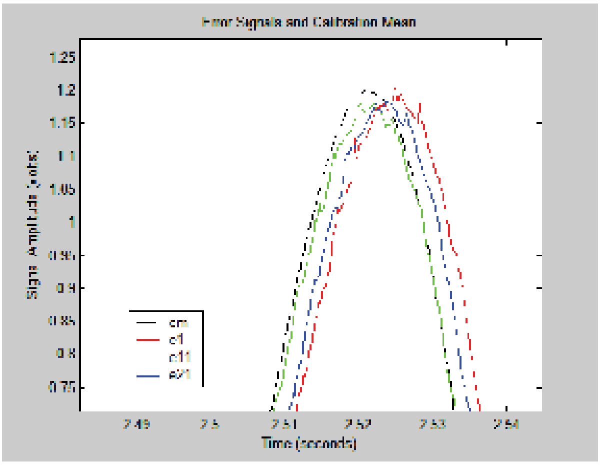


Fig. 9. Expanded view of Figure 8 near 2.5 seconds.

Signal	**From**	**To**	**x-coordinate**	$RSSe_i$	**Dial Gauge**
e_1	0.030	-89.970	0.313	24.1117	0.487
e_2	0.027	-89.973	0.281	24.0249	0.489
e_3	0.024	-89.976	0.250	23.7619	0.489
e_4	0.021	-89.979	0.219	23.5060	0.491
e_5	0.018	-89.982	0.188	22.8818	0.493
e_6	0.015	-89.985	0.156	21.1411	0.494
e_7	0.012	-89.988	0.125	20.9980	0.496
e_8	0.009	-89.991	0.094	20.6291	0.497
e_9	0.006	-89.994	0.063	21.6124	0.498
e_{10}	0.003	-89.997	0.031	21.0818	0.500
e_{11}	0.000	-90.000	0.000	5.6112	0.501
e_{12}	-0.003	-90.003	-0.031	20.9378	0.503
e_{13}	-0.006	-90.006	-0.063	19.1932	0.504
e_{14}	-0.009	-90.009	-0.094	20.5077	0.506
e_{15}	-0.012	-90.012	-0.123	17.2669	0.507
e_{16}	-0.015	-90.015	-0.156	17.1234	0.508
e_{17}	-0.018	-90.018	-0.188	16.9160	0.509
e_{18}	-0.021	-90.021	-0.219	16.6139	0.511
e_{19}	-0.024	-90.024	-0.250	16.0514	0.512
e_{20}	-0.027	-90.027	-0.281	15.0220	0.514
e_{21}	-0.030	-90.030	-0.313	13.9787	0.516

Table 5. $RSSe_i$ and dial gauge measurements for Experiment 4.

Figure 10 shows the 21 $RSSe_i$ measurements plotted as a function of the 21 corresponding final robot workspace *x*-coordinate values to which the robot was commanded to move. Figure 10 also shows the $RSSe_i$ error detection limit established in Experiment 3 (9.01). $RSSe_i$ measurements were calculated using the procedure described for Experiment 3:

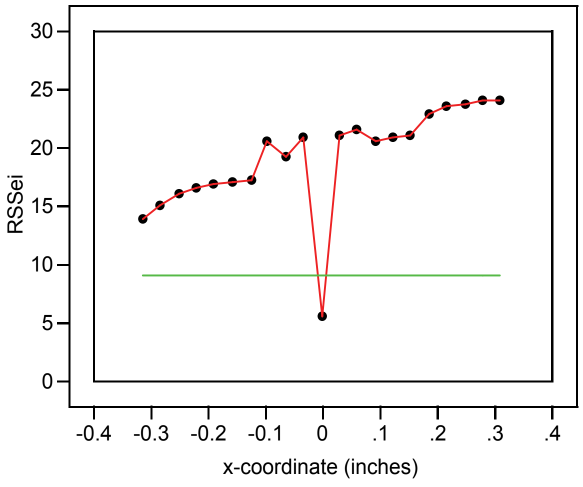


Fig. 10. $RSSe_i$ vs. robot workspace x-coordinate values.

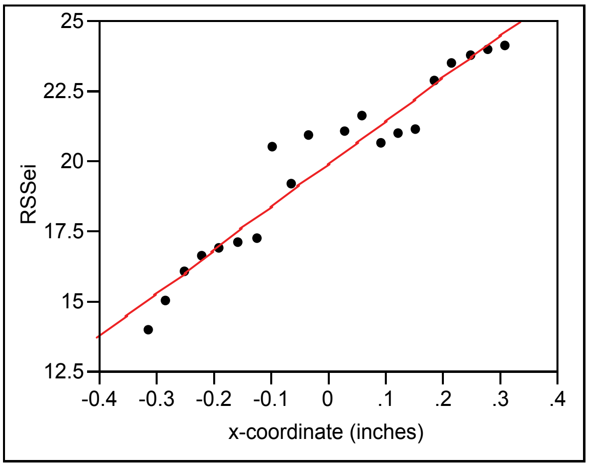


Fig. 11. $RSSe_i$ vs. robot workspace x-coordinate values for position errors.

$$RSSe_i = \sqrt{\sum_{n=25000}^{35000} [e_i(n) - c_m(n)]^2} \tag{5}$$

Figure 9 shows that sensor signals for both positive and negative final robot workspace x-coordinate values lag the calibration mean c_m, whereas e_{11}, the signal generated when the robot moves without offset from the home and test positions, closely matches the calibration mean. Both positive and negative final robot workspace x-coordinate values may lead to signals that lag the calibration mean because the generated sensor signals depend on both the distance between the sensor and the moving robot arm and the velocity of the moving robot arm.

Figure 10 shows that the $RSSe_i$ measures calculated for any of the offset robot motions exceed the single-point error limit established in Experiment 3. The method detects any induced robot position errors, to the repeatability specification of the robot. In addition, by excluding the point in Figure 10 corresponding to e_{11}, the non-error condition signal, an analysis of variance shows evidence of a statistically significant relationship between $RSSe_i$ measurements and commanded final *x*-coordinate values (α = 0.05, p-value < 0.0001). Equation 6 gives the equation of the least squares line shown in Figure 11:

$$\text{Predicted } RSSe_i = 19.87 + 15.31 * x\text{-coordinate} \quad (6)$$

The model explains 93.21% of the variability in $RSSe_i$ measurements. Random measurement errors or other unexplained factors account for only a small amount of the observed variability in the data.

Experiment 4 results indicate that:

1. Sensor signals can be used to detect single-axis on-line robot position errors at robot repeatability levels.
2. There is evidence of a statistically significant relationship between the error measure developed and actual robot position error. The relationship might allow not only detecting robot position errors, but also determining the directions and magnitudes of errors.

The original intention of the study was to develop a low-cost robust method for simply detecting on-line robot position errors. Results show that the proposed method can detect on-line position errors with 100% accuracy at robot repeatability levels. In addition, the linear relationship between the error measure developed and actual robot position error indicates that the method may provide additional capabilities, beyond position error detection.

In future studies, the proposed method can be improved by fully automating the data collection and analysis process and repeating the Experiment 3 process. In addition, the unexpected advanced error detection capabilitites of the method can be improved by replicating the Experiment 4 process to help reduce the effects unexplained variability on the linear error prediction model.

8. Experiment 5

The objective of Experiment 5 was to:

1. Test the robot position error detection model developed in Experiment 4.

The method used to test the error detection model consisted of six steps:

1. Command the robot to move from the home position +/- 0.03 T-axis degrees to the test position +/- 0.03 T-axis degrees, in 0.003 degree increments, and in random order.
2. For each move, measure the position of the robot using a dial gauge.
3. Simultaneously measure the signal (r_i) generated by the sensor.
4. Calculate the error detection measure ($RSSr_i$) for the given sensor signal.
5. For each output signal, use the developed error detection model to predict whether or not the robot was in an error condition.
6. Compare error detection model predictions to actual robot positions to determine the system's capability for detecting position errors.

Table 6 shows the 21 positions from which the robot was commanded to move, the 21 positions to which the robot was commanded to move, and the corresponding final robot workspace *x*-coordinate values to which the robot was commanded to move. The final robot workspace *y*-coordinate values were the same for all 21 positions to which the robot was commanded to move (-597.056 mm).

Signal	From (Degrees)	To (Degrees)	X-Coordinate (Mm)	$RSSr_i$	Dial Gauge (Inches)
r_1	-0.030	-90.030	-0.313	15.5917	0.515
r_2	0.000	-90.000	0.000	6.9508	0.502
r_3	-0.027	-90.027	-0.281	15.8215	0.514
r_4	-0.015	-90.015	-0.156	19.4724	0.508
r_5	-0.024	-90.024	-0.250	16.5845	0.513
r_6	0.018	-89.982	0.188	25.6551	0.493
r_7	0.021	-89.979	0.219	25.6022	0.491
r_8	0.027	-89.973	0.281	26.9904	0.489
r_9	-0.021	-90.021	-0.219	18.4504	0.511
r_{10}	-0.018	-90.018	-0.188	18.2697	0.510
r_{11}	0.006	-89.994	0.063	22.2440	0.498
r_{12}	0.012	-89.988	0.125	23.5641	0.496
r_{13}	0.030	-89.970	0.313	27.1052	0.488
r_{14}	0.024	-89.976	0.250	25.5870	0.491
r_{15}	-0.003	-90.003	-0.031	22.8124	0.504
r_{16}	0.009	-89.991	0.094	23.5282	0.498
r_{17}	-0.009	-90.009	-0.094	20.8457	0.506
r_{18}	-0.012	-90.012	-0.123	20.3149	0.506
r_{19}	-0.006	-90.006	-0.063	22.2316	0.504
r_{20}	0.015	-89.985	0.156	24.6732	0.494
r_{21}	0.003	-89.997	0.031	23.0340	0.501

Table 6. $RSSr_i$ and dial gauge measurements for Experiment 5.

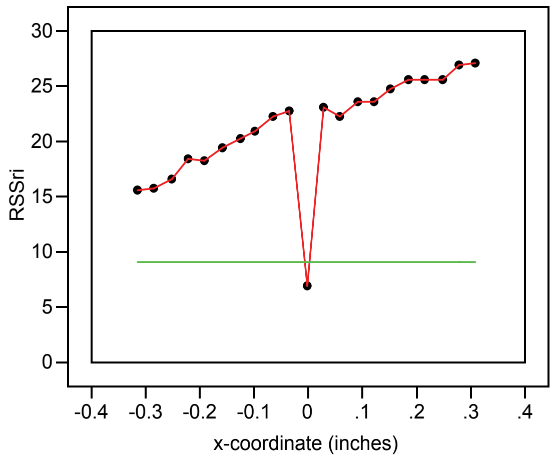


Fig. 12. *RSSri* vs. robot workspace *x*-coordinate values.

Table 6 also shows the 21 resulting $RSSr_i$ measurements and the 21 corresponding dial gauge measurements for Experiment 5.

Figure 12 shows the 21 $RSSr_i$ measurements plotted as a function of the 21 corresponding final robot workspace *x*-coordinate values to which the robot was commanded to move. Figure 12 also shows the $RSSe_i$ error detection limit established in Experiment 3 (9.01). $RSSr_i$ measurements were calculated using the procedure described for Experiment 3:

$$RSSr_i = \sqrt{\sum_{n=25000}^{35000} [r_i(n) - c_m(n)]^2} \tag{7}$$

From Experiment 3 results, the error detection model predicts a robot position error for any $RSSr_i$ value greater than 9.01. In addition, from Equation 6,

$$RSSr_i = 19.87 + 15.31 * \text{x - coordinate} \tag{8}$$

Therefore, the *x*-coordinate of the final robot position can be predicted:

$$\text{Predicted x - coordinate} = \frac{RSSr_i - 19.87}{15.31} = \text{-}1.298 + 0.0653 * RSSr_i \tag{9}$$

Finally, from the *x*-coordinate prediction, the direction and magnitude of the single-axis robot position error can also be predicted. Negative *x*-coordinate values indicate that the robot moved past the desired position; positive values indicate that the robot did not reach the desired position (with respect to the home position). The difference between the predicted and desired *x*-coordinate indicates the magnitude of the position error. For Experiment 5, the desired *x*-coordinate is always zero. Therefore, the value of the predicted *x*-coordinate indicates the magnitude of the position error.

Table 7 shows commanded (actual) and predicted *x*-coordinate values for Experiment 5. Table 7 also shows actual errors and predicted errors, whether or not the direction (sign) of the predicted error is correct, and the difference between the predicted error magnitude and the actual (induced) error magnitude (errors due to the prediction model).

Experiment 5 results show that:

1. The robot position error detection model developed in Experiment 4 predicts Experiment 5 errors with 100% accuracy, error direction with 81% accuracy, and error magnitude to within 0.223 mm.

Experiment 5 results indicate that the method developed can reliably identify robot position errors at robot repeatability levels. The method can also, to some degree, identify the direction of an error relative to the desired (commanded) position and the magnitude of the error.

In future studies, in addition to improvements recommended in Experiment 4 results, the proposed method can be improved by using standard $\overline{X}$ control chart techniques, including subgroup sampling and averaging. Using standard $\overline{X}$ control chart techniques could reduce random variation between prediction model and in-process measurements and, thereby, improve the accuracy and reliability of all three aspects of error detection and identification (error detection, error direction, and error magnitude).

Signal	Actual x-coordinate (mm)	Predicted x-coordinate (mm)	Actual Error	Predicted Error	Sign Correct	Model Error (mm)
r_1	-0.313	-0.2794	Yes	Yes	Yes	0.034
r_2	0.000	0.0000	No	No	Yes	0.000
r_3	-0.281	-0.2644	Yes	Yes	Yes	0.017
r_4	-0.156	-0.0260	Yes	Yes	Yes	0.130
r_5	-0.250	-0.2146	Yes	Yes	Yes	0.035
r_6	0.188	0.3779	Yes	Yes	Yes	0.190
r_7	0.219	0.3744	Yes	Yes	Yes	0.155
r_8	0.281	0.4651	Yes	Yes	Yes	0.184
r_9	-0.219	-0.0927	Yes	Yes	Yes	0.126
r_{10}	-0.188	-0.1045	Yes	Yes	Yes	0.084
r_{11}	0.063	0.1551	Yes	Yes	Yes	0.092
r_{12}	0.125	0.2413	Yes	Yes	Yes	0.116
r_{13}	0.313	0.4726	Yes	Yes	Yes	0.160
r_{14}	0.250	0.3734	Yes	Yes	Yes	0.123
r_{15}	-0.031	0.1922	Yes	Yes	No	0.223
r_{16}	0.094	0.2389	Yes	Yes	Yes	0.145
r_{17}	-0.094	0.0637	Yes	Yes	No	0.158
r_{18}	-0.123	0.0291	Yes	Yes	No	0.152
r_{19}	-0.063	0.1543	Yes	Yes	No	0.217
r_{20}	0.156	0.3137	Yes	Yes	Yes	0.158
r_{21}	0.031	0.2067	Yes	Yes	Yes	0.176

Table 7. Predicted vs. actual errors.

9. Conclusions

The investigator developed an non-invasive non-contact method for detecting on-line industrial robot position errors. The method uses a low-cost sensor to detect single-axis position errors. The sensor, composed of a low-cost microwave Doppler radar detector and a low-pass filter, converts robot motion into electronic signals that are A/D converted and processed using a computer.
Computer processing reduces captured signals into root-sum-of-squares error measures, with respect to a mean sensor calibration signal. Root-sum-of-squares error measures are compared to a threshold value that indicates, statistically, a 99.7% probability that an on-line position error has occurred. The threshold value can be adjusted to meet different application needs.
For the prototype constructed, and the experiments run, the sensor detected position errors with 100% accuracy, error direction with 81% accuracy, and error magnitude to within 0.223 mm.
The proposed method offers a low-cost non-invasive non-contact means for detecting on-line in-process robot position errors. Accurate in-process robot position error detection indicates the need for corrective action: robot homing, recalibration, or repair.
The proposed method offers advantages over other possible methods. The sensor developed uses a microwave Doppler radar detector, which is generally less expensive

and/or more robust in industrial environments than optical sensors, such as laser tracking systems or cameras. The proposed method is generally more practical for in-process error detection than contact devices used for robot recalibration, such as cable systems, trigger probes, or dial gauges.

The proposed method may eliminate the need for regularly scheduled robot homing or recalibration, thus improving productivity. At the same time, the proposed method identifies error conditions when they exist, reducing scrap, which also lowers costs and improves productivity.

Future proposed enhancements include:

1. Improving sensor design,
2. Improving sensor placement,
3. Detecting multi-axis position errors by choosing different sensor placement strategies or by using multiple sensors at a given position,
4. Fully automating the data collection and analysis process,
5. Using control chart techniques to improve error detection capabilities, particularly advanced error direction and error magnitude prediction capabilities, and
6. Considering different methods for removing DC bias from sensor signals.

10. References

The following references were cited to acknowledge relevant related research. They do not necessarily reflect the views or beliefs of the author.

Agapakis, J. E., Katz, J. M., Friedman, J. M., & Epstein, G. N. (1990). Vision-aided robotic welding: an approach and a flexible implementation. *International Journal of Robotics Research, 9*(5), 17-34.

Besterfield, D. H. (2001). *Quality Control, Sixth Edition*. New Jersey: Prentice-Hall.

Boubekri, N., & Sherif, W. (1999). A position control algorithm for a microcomputer controlled SCARA robot – part 1. *Computers & Industrial Engineering, 19*(1-4), 477-480.

Bruyninckx, H., Lefebvre, T., Mihaylova, L., Staffetti, E., De Schutter, J., & Xiao, J. (2001). A roadmap for autonomous robotic assembly, *Proceedings of the 4th IEEE International Symposium on Assembly and Task Planning*, Fukuoka, Japan, May 28-29, pp. 49-54.

Caccavale, F., & Walker, D. (1997). Observer-based fault detection for robot manipulators, *Proceedings of the IEEE International Conference on Robotics and Automation*, Albuquerque, NM, April, pp. 2881-2887.

Dungern, O. V., & Schmidt G. K. (1986). A new scheme for on-line fault detection and control of assembly operation via sensory information, *Proceedings of the 25th Conference on Decision and Control*, Athens, Greece, December, pp. 1891-1892.

Forcinio, H. (1999). Tighter process control ensures weld quality. *Robotics World, 17*(1), 17-20.

Mayer, J. R., & Parker, G. A. (1994). A portable instrument for 3-D dynamic robot measurements using triangulation and laser tracking. *IEEE Transactions on Robotics and Automation, 10*(4), 504-516.

Microwave Solutions (2002). *Small PCB Style - MDU1620*. Retrieved May 23, 2002, from http://www.microwave-solutions.com.

Millman, J., & Halkias, C. C. (1972). *Integrated electronics: analog and digital circuits and systems.* New York: Mc-Graw Hill.

Nakamura, H., Itaya, T., Yamamoto, K., & Koyama, T. (1995). Robot autonomous error calibration method for off-line programming system, *IEEE International Conference on Robotics and Automation*, pp. 1775-1982.

Owens, J. (1994). Robotrak: calibration on a shoestring. *Industrial Robot, 21*(6), 10-13.

Robinson, P., Orzechowski, P., James, P. W., & Smith, C. (1997). An automated robot calibration system, *IEEE International Symposium on Industrial Electronics, 1*, pp. SS285-SS290.

Rocadas, P. S., & McMaster, R. S. (1997). A robot cell calibration algorithm and its use with a 3D measuring system, *IEEE International Symposium on Industrial Electronics, 1*, pp. SS297-SS302.

Seiko Instruments USA, Inc. (1986). *Seiko D-TRAN Intelligent Robots: RT-3000 Installation, Programming, and Operation Manual*. Torrance, CA: Seiko Instruments USA, Inc.

Swanson, D. C. (2000). *Signal processing for intelligent sensor systems*. New York: Marcel Dekker.

van Albada, G. D., Lagerberg, J. M., & Visser, A. (1994). Eye in hand robot calibration. *Industrial Robot, 21*(6), 14-17.

Visinsky, M. L., Cavallaro, J. R., & Walker, I. D. (1994). Robotic fault detection and fault tolerance: a survey, *Reliability Engineering and System Safety, 46*, 139-158.

Xu, W., & Mills, J. K. (1999). A new approach to the position and orientation calibration of robots, *Proceedings of the 1999 IEEE International Symposium on Assembly and Task Planning*, Porto, Portugal, July.

Zhong, X-L., & Lewis, J. M. (1995). A new method for autonomous robot calibration, *IEEE International Conference on Robotics and Automation*.

Study on Three Dimensional Dual Peg-in-hole in Robot Automatic Assembly

Fei Yanqiong[1], Wan Jianfeng[2], Zhao Xifang[1]
[1]*Research Institute of Robotics, Shanghai Jiao Tong University, Huashan Road 1954, Shanghai 200030 (P. R. of China)*
[2]*School of Materials Science and Engineering, Shanghai Jiao Tong University, Huashan Road 1954, Shanghai 200030 (P. R. of China)*

1. Introduction and Importance

The analyses of the physics of robot assembly tasks are important in developing a flexible assembly system. Assembly analyses involve force/moment analysis due to contacts and jamming analysis due to improper force/moment.

Because the uncertainty of geometry and control, contact states exist during the assembly process. Xiao presented the notion of contact formation (CF) in terms of principal contacts (PCs) to characterize contact states, and defined contact states as CF-connected regions of contact configurations (Xiao, 1993). They also examined the neighboring relations between contact states and characterized the contact state space as a contact state graph. Xiao developed an algorithm that automatically generated a high-level contact state graph and planned contact motions between two known adjacent contact states (Xiao, 2001).

Force/moment analysis is an important element in robot assembly analyses. Lots of research has been done on single peg-in-hole. Simunivic analyzed a round peg insertion problem, and derived two dimensional single peg-in-hole jamming conditions (Simunivic, 1972). Whitney adopted Simunovic's approach to determine the force analysis of the same problem (Whitney, 1982). He identified wedging and jamming conditions and recommended ways to avoid insertion failure. A compliant mechanism, RCC (Remote-Center-Compliance) was designed for axisymmetric part insertions. Sturges and Laowattana analyzed the wedging condition in rectangular peg insertion and developed a Spatial Remote-Center-Compliance (SRCC) (Sturges, 1988, 1996, 1996). Non-axisymmetric part insertions can be performed. Tsaprounis focused on the determination of forces at contact points between a round peg and a hole (Tsaprounis, 1998).

Considerable theories have also been studied over the past years that define two dimensional mating between double rigid cylindrical parts (Arai, 1997, McCarragher, 1995 and Sathirakul, 1998). Arai described error models and analyzed the search strategy of a two dimensional dual peg-in-hole (Arai, 1997). McCarragher modeled assembly as a discrete event dynamic system using Petri nets and developed a discrete event controller

(McCarragher, 1995). He applied his method to a two dimensional dual peg-in-hole insertion successfully. Sturges analyzed two dimensional dual peg-in-hole insertion problems, enumerated possible contact states, and derived geometric conditions and force-moment equations for static-equilibrium states of two dimensional dual peg insertions (Sathirakul and Sturges, 1998). The jamming diagrams of a two dimensional dual peg-in-hole were obtained.

At present, research activities in the area are concentrated on:

(1) Three dimensional multiple peg-in-hole analyses
(2) Modeling of contact
(3) Design of assembly devices

Multiple peg-in-hole insertions represent a class of practical and complicated tasks in the field of robotic automatic assembly. Because of the difficulty and complexity in analyzing three dimensional multiple peg-in-hole insertions, most of the analyses of this class of assembly tasks to date were done by simplifying the problems into two dimensions (Arai, 1997, McCarragher, 1995 and Sathirakul, 1998). The analyses of the dual peg-in-hole in three dimensions are few (Rossitza, 1998). Rossitza considered the geometric model and quasi-static force model of a three dimensional dual peg-in-hole and presented a method for 3D simulation of the dual peg insertion. The assembly process, considered as a sequence of discrete contact events, is modeled as a transition from one state to another. Sturges' work on the three dimensional analysis of multiple-peg insertion gave insight into the behavior of the pegs during the chamfer-crossing phase of the problem (Sturges, 1996). The net forces and torque during chamfer-crossing of a triple-peg insertion were obtained. The force/torque prediction gave a better insight into the physics of multiple peg-in-hole insertion with the aid of compliance mechanisms (SRCC) (Sturges, 1996). Fei's geometric analysis of a triple peg-in-hole mechanism gives the base of three dimensional assembly problems (Fei, 2003). The geometric features of three-dimensional assembly objects are represented by two elements. The geometric conditions for each contact state are derived with the transformation matrix (Fei, 2003).

However, for three dimensional multiple peg-in-hole insertion tasks, there are not yet perfect assembly theory and experimental analyses at present. After analyzing multiple peg-in-hole in three dimensions, the whole assembly process and its complexity and difficulty can be known in detail. Thus, some simple and special machines can be designed to finish this type of tasks.

In the paper, a complete contact and jamming analysis of dual peg-in-hole insertion is described in three dimensions. By understanding the physics of three dimensional assembly processes, one can plan fine motion strategies and guarantee successful operations.

In this article, three dimensional dual peg-in-hole insertion problems are analyzed. Firstly, all possible contact states are enumerated in three dimensions. Secondly, contact forces can be described by the screw theory. The screw theory gives a very compact and efficient formulation. Applying the static equilibrium equations, the conditions on applied forces and moments for maintaining each contact state are formulated. Thirdly, jamming diagrams are obtained. 33 kinds of possible jamming diagrams are analyzed. The force/moment conditions to guarantee successful insertion are obtained from all possible jamming diagrams. Then, some contact states are given, which finish a dual peg-in-hole process. Their geometric conditions are derived to verify the analyses.

2. Contact states classification of dual peg-in-hole

The geometric model of a dual peg-in-hole problem is shown in Fig. 1. Left peg has a radius of r_{P1}, whereas right peg has a radius of r_{P2}. The radii of left hole and right hole are r_{H1} and r_{H2}, respectively. D (D_P) represents the distance between the peg axes, and D_H represents the distance between the hole axes. The clearances between left peg and left hole, and between right peg and right hole are c_1 and c_2, respectively. To simplify our analysis, we only consider multiple pegs which are round, parallel and have the same length. Besides these, we make the following assumptions:

1) In the assembly process, the angle θ between the axes of the pegs and those of the holes is small and can be neglected.
2) Insertion direction is only vertically downward.
3) Quasi-static motion.
4) All pegs have the same length and are parallel.
5) All axes of the holes are parallel to each other, and perpendicular to the horizontal plane.
6) The clearances between pegs and holes are positive.

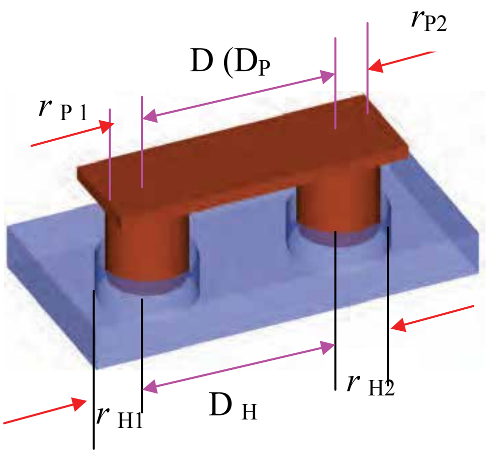


Fig. 1. Geometry of a dual peg-in-hole.

2.1 Contact states of a dual peg-in-hole in three dimensions

Because the uncertainty of geometry and control, contact states exist during the assembly process. For a dual peg-in-hole insertion, three-point contact states exist only when a certain set of conditions are satisfied. Three-point contact states or four-point contact states are transient and considered trivial (Sathirakul and Sturges, 1998). Thus, in the paper, one-point and two-point contact states will be analyzed in detail.

Expect three kinds of line contact states (Sathirakul and Sturges, 1998), there are ten kinds of point contact states when pegs tilt to the left, as shown in Table 1. Table 1 shows all possible point contact states of a dual peg-in-hole in three dimensions. The vectors of contact points, and the angular ranges between the vectors of contact forces and the x-axis can be seen in Table 1.

Dual peg-in-hole	Three dimensional contact states		α_i	r_i
When pegs tilt to the left, *k* stands for *l*. When pegs tilt to the right *k* stands for *r*.	One-point contact states	(*k*-1)	$\alpha_1 \in [0^\circ \ \ 90^\circ] \cup [270^\circ \ \ 360^\circ]$	$r_1 : (x_1 \ \ y_1 \ \ -jh_1)$
		(*k*-2)	$\alpha_2 \in [90^\circ \ \ 270^\circ]$	$r_2 : (x_2 \ \ y_2 \ \ -(1-j)h_2)$
		(*k*-3)	$\alpha_3 \in [0^\circ \ \ 90^\circ] \cup [270^\circ \ \ 360^\circ]$	$r_3 : (x_3 \ \ y_3 \ \ -jh_3)$
		(*k*-4)	$\alpha_4 \in [90^\circ \ \ 270^\circ]$	$r_4 : (x_4 \ \ y_4 \ \ -(1-j)h_4)$
	Two-point contact states	(*k*-5)	$\alpha_1 \in [0^\circ \ \ 90^\circ] \cup [270^\circ \ \ 360^\circ]$, $\alpha_2 \in [90^\circ \ \ 270^\circ]$	$r_1 : (x_1 \ \ y_1 \ \ -jh_1)$, $r_2 : (x_2 \ \ y_2 \ \ -(1-j)h_2)$
		(*k*-6)	$\alpha_3 \in [0^\circ \ \ 90^\circ] \cup [270^\circ \ \ 360^\circ]$, $\alpha_4 \in [90^\circ \ \ 270^\circ]$	$r_3 : (x_3 \ \ y_3 \ \ -jh_3)$, $r_4 : (x_4 \ \ y_4 \ \ -(1-j)h_4)$
		(*k*-7)	$\alpha_1 \in [0^\circ \ \ 90^\circ] \cup [270^\circ \ \ 360^\circ]$, $\alpha_3 \in [0^\circ \ \ 90^\circ] \cup [270^\circ \ \ 360^\circ]$	$r_1 : (x_1 \ \ y_1 \ \ -jh_1)$, $r_3 : (x_3 \ \ y_3 \ \ -jh_3)$
		(*k*-8)	$\alpha_1 \in [0^\circ \ \ 90^\circ] \cup [270^\circ \ \ 360^\circ]$, $\alpha_4 \in [90^\circ \ \ 270^\circ]$	$r_1 : (x_1 \ \ y_1 \ \ -jh_1)$, $r_4 : (x_4 \ \ y_4 \ \ -(1-j)h_4)$
		(*k*-9)	$\alpha_2 \in [90^\circ \ \ 270^\circ]$, $\alpha_3 \in [0^\circ \ \ 90^\circ] \cup [270^\circ \ \ 360^\circ]$	$r_2 : (x_2 \ \ y_2 \ \ -(1-j)h_2)$, $r_3 : (x_3 \ \ y_3 \ \ -jh_3)$
		(*k*-10)	$\alpha_2 \in [90^\circ \ \ 270^\circ]$, $\alpha_4 \in [90^\circ \ \ 270^\circ]$	$r_2 : (x_2 \ \ y_2 \ \ -(1-j)h_2)$, $r_4 : (x_4 \ \ y_4 \ \ -(1-j)h_4)$
When pegs tilt to the left *j*=1. When pegs tilt to the right *j*=0.				

Table 1. Assembly contact states of a dual peg-in-hole in three dimensions.

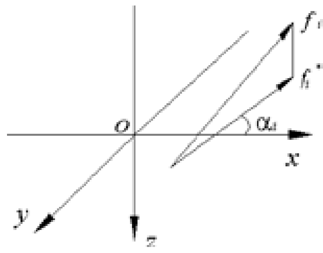


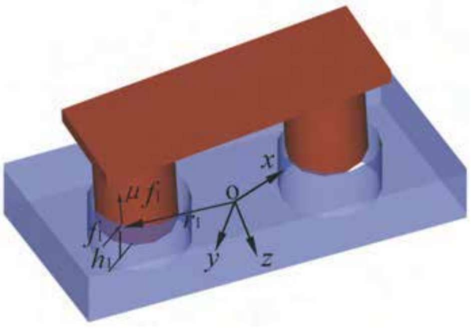
Fig. 2. α_i between the vector f_i * and the x-axis.

The coordinate system oxyz is set up at the center of dual pegs. Here i=1 represents that the left peg and left hole contact in the left, i=2 represents that the left peg and left hole contact in the right, i=3 represents that the right peg and right hole contact in the left, i=4 represents that the right peg and right hole contact in the right. i=1 or i=3 represents left-contact; i=2 or i=4 represents right-contact. Left-contact is the situation in which the left or right peg makes contacts with its hole in the left; Right-contact is the situation in which the left or right peg makes contacts with its hole in the right. Mapping the contact force f_i to the plane xoy, the corresponding vector f_i^* can be obtained; α_i is an angle between the vector f_i^* and the x-axis (Fig. 2). According to the above assumption, θ is small and can be neglected. Thus, f_i^* is equal to f_i; r_i is a vector which represents contact point's position in fixed coordinate system oxyz; h_i is an insertion depth; μ is friction coefficient. According to the above analysis, there are also ten kinds of contact states when pegs tilt to the right (Table 1).

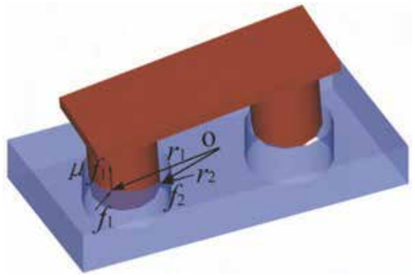
For example, let pegs tilt to the left, if *i*=1, one-point contact state (*l*-1) exists. If *i*=1and *i*=4, two-point contact state (*l*-8) exists (Fig. 3). Let pegs tilt to the right, If *i*=1, one-point contact state (*r*-4) exists; if *i*=1and *i*=4, two-point contact state (*r*-8) exists (Fig. 3). During the insertion, the dual-peg may undergo several contact states. In the paper, some states can be chosen, such as, pegs tilting to the left: state (*l*-1), state (*l*-5), state (*l*-8), a possible three-point contact state and a mating state or pegs tilting to the right: state (*r*-4), state (*r*-6), state (*r*-8), a possible three-point contact state and a mating state to finish a dual peg-in-hole assembly process. Their three dimensional charts and two dimensional charts are shown in Fig. 3 and Fig. 4.

2.2 Contact states of a dual peg-in-hole in two dimensions

When $\alpha_i = 0°$ or 180°, three dimensional problems can be simplified to two dimensional problems. The two dimensional chart of each state in Fig. 3 is shown in Fig. 4.



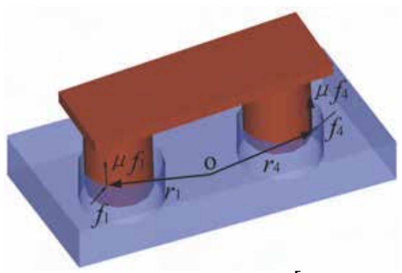
(state *l*-1): $\alpha_1 \in [0° \quad 90°] \cup [270° \quad 360°]$, $r_1 : (x_1 \quad y_1 \quad -h_1)$



$\alpha_1 \in [0° \quad 90°] \cup [270° \quad 360°], \alpha_2 \in [90° \quad 270°]$

$r_1 : (x_1 \quad y_1 \quad -h_1)$, $r_2 : (x_2 \quad y_2 \quad 0)$

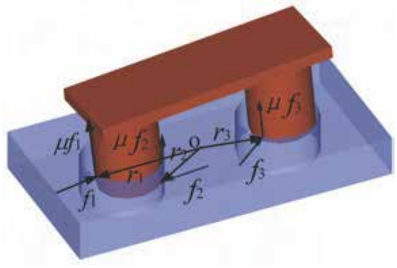
(state *l*-5)



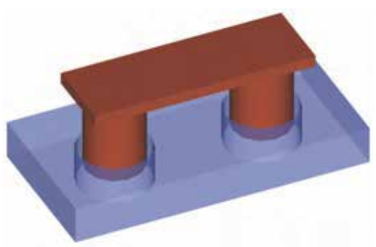
$\alpha_1 \in [0° \quad 90°] \cup [270° \quad 360°], \alpha_4 \in [90° \quad 270°]$

$r_1 : (x_1 \quad y_1 \quad -h_1)$, $r_4 : (x_4 \quad y_4 \quad 0)$

(state *l*-8)

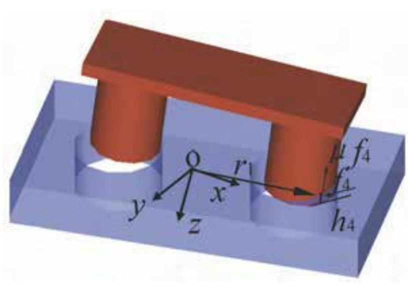


a possible three-point contact state

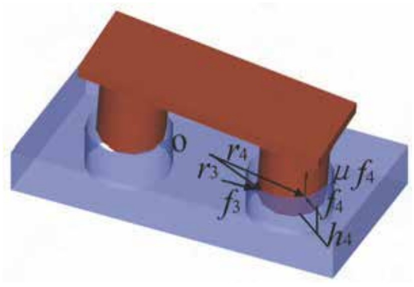


mating state

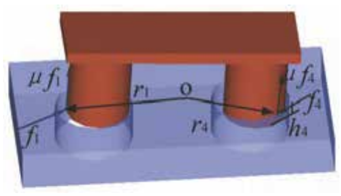
(a) pegs tilt to the left



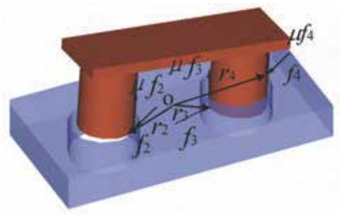
(state *r*-4)



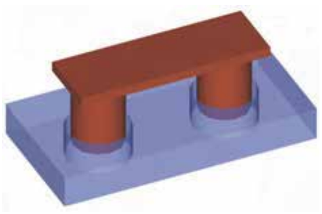
(state *r*-6)



(state *r*-8)

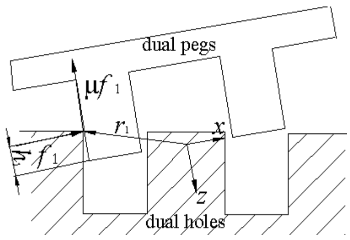


a possible three-point contact state

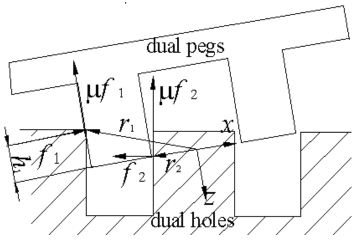

mating state

(b) pegs tilt to the right

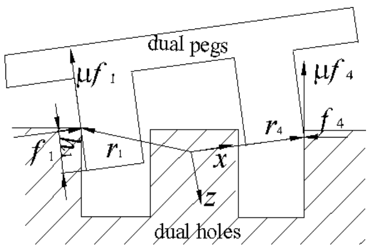
Fig. 3. Assembly states of a dual peg-in-hole in three dimensions.



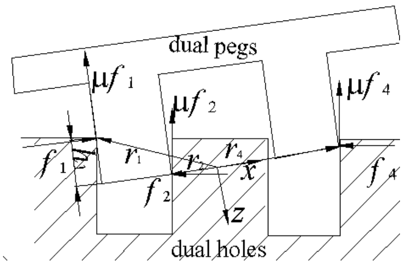
(state *l*-1) (α_1=0°)



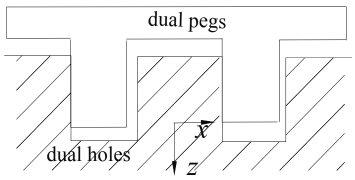
(state *l*-5) (α_1=0°, α_2=180°)



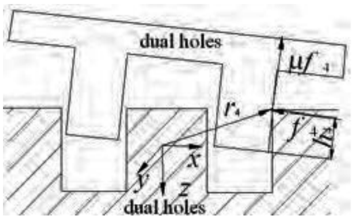
(state *l*-8) (α_1=0°, α_4=180°)



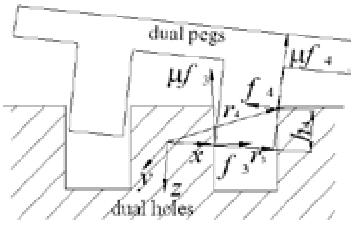
a possible three-point contact state



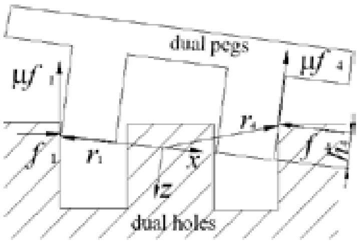
mating state
(a) pegs tilt to the left



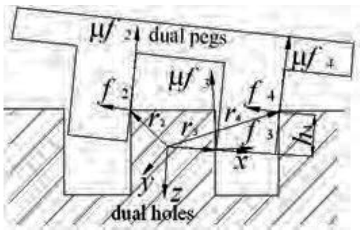
state *r*-4 (α_4=180°)



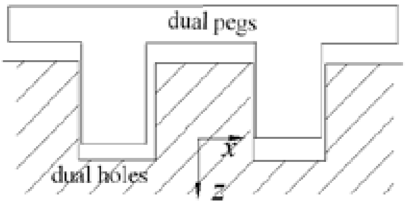
state *r*-6 (α_3=0°, α_4=180°)



state *r*-8(α_1=0°, α_4=180°)



a possible three-point contact state



mating state
(b) pegs tilt to the right

Fig. 4. Assembly states of a dual peg-in-hole in two dimensions.

3. Geometric constraints and contact states analysis

According to the above chosen states (pegs tilting to the left: state *l*-1, state *l*-5, state *l*-8, a possible three-point contact state and a mating state or pegs tilting to the right: state *r*-4, state *r*-6, state *r*-8, a possible three-point contact state and a mating state), the relations between these states and their geometric constraints are analyzed. Thus, the complexity of the three dimensional dual peg-in-hole can be known. The dimensions of a dual peg and a dual hole are given as follows (Fig. 1): $2r_{P_1} = 2r_{P_2} = 9.9mm$, $2r_{H_1} = 2r_{H_2} = 10.0mm$, $D_P = 90.2mm$, $D_H = 90.25mm$, $C_1 = C_2 = 0.075mm$, μ=0.1.

3.1 Two dimensions

The boundary state of state *l*-1 or state *r*-4 is shown in Fig. 5. Its geometric constraint is

$$\begin{cases} h_0 s\theta_0 + 2r_{P_1} c\theta_0 = 2r_{H_1} \\ h_0 c\theta_0 = 2r_{P_1} s\theta_0 \end{cases} \tag{1}$$

Thus,θ_0=8.1°, h_0=1.397mm can be obtained. When θ is less than θ_0, state *l*-1 or state *r*-4 can be reached.

According to Fig. 3, for pegs tilting to the left, the geometric constraint of state *l*-5 can be expressed as,

$$h_1 s\theta + 2r_{P_1} c\theta = 2r_{H_1} \tag{2a}$$

The geometric constraint of state *l*-8 can be expressed as,

$$h_1 s\theta + \left(D_P + r_{P_2} + r_{P_1}\right)c\theta = \left(D_H + r_{H_2} + r_{H_1}\right) \tag{3a}$$

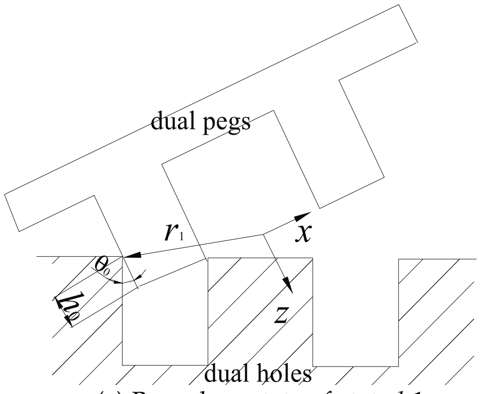
According to Fig. 3, for pegs tilting to the right, the geometric constraint condition of state *r*-6 in two dimensions can be expressed as,

$$h_4 s\theta + 2r_{P_1} c\theta = 2r_{H_1} \tag{2b}$$

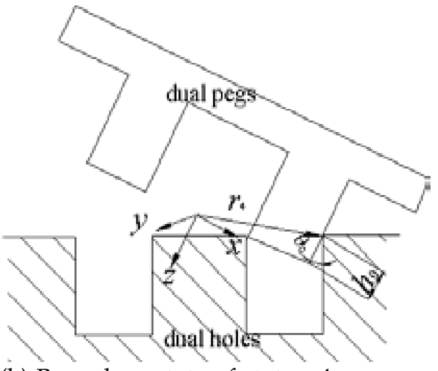
The geometric constraint condition of state *r*-8 in two dimensions can be expressed as,

$$h_4 s\theta + \left(D_P + r_{P_2} + r_{P_1}\right)c\theta = \left(D_H + r_{H_2} + r_{H_1}\right) \tag{3b}$$

If Eqs. (2a) and (3a) or Eqs. (2b) and (3b) have one solution at least, a three-point contact state of two dimensional dual peg-in-hole can be reached. Fig.6 suggests that a three-point contact state is relative to the geometry of assembly objects. At the point where two curves of plots intersect, the three-point contact states can exist. From Fig. 6, we can see that three-point contact states are transient. Hereθ_0 andθ are angles between the axes of a peg and a hole, h_0 and h_i are insertion depths.



(a) Boundary state of state *l*-1



(b) Boundary state of state *r*-4.

Fig. 5. Boundary state.

3.2 Three dimensions

Three dimensional multiple peg-in-hole problems are more complex than two dimensional peg-in-hole problems. But they have similar geometric constraint conditions. For the boundary state of state *l*-1 or state *r*-4 in three dimensions, we can obtain:

$$\begin{cases} h_0 s\theta_0 + 2r'_{P_1} c\theta_0 = 2r'_{H_1} \\ h_0 c\theta_0 = 2r'_{P_1} s\theta_0 \end{cases}, \; r'_{P_1} \le r_{P_1}, \; r'_{H_1} \le r_{H_1} \tag{4}$$

If $\theta < \theta_0$, state *l*-1 or state *r*-4 can be reached.

For state *l*-5 in three dimensions, we can obtain the following formulas:

$$h_1 s\theta + 2r'_{P_1} c\theta = 2r'_{H_1},\ r'_{P_1} \le r_{P_1},\ r'_{H_1} \le r_{H_1} \tag{5a}$$

For state *l*-8 in three dimensions, we can obtain the following formulas:

$$h_1 s\theta + \left(D'_P + r'_{P_2} + r'_{P_1}\right) c\theta = \left(D'_H + r'_{H_2} + r'_{H_1}\right),$$

$$r'_{P_1} \le r_{P_1},\ r'_{P_2} \le r_{P_2},\ D'_P \le D_P,\ r'_{H_1} \le r_{H_1},\ r'_{H_2} \le r_{H_2},\ D'_H \le D_H \tag{6a}$$

For state *r*-6 in three dimensions, we can obtain the following constraint formulas:

$$h_4 s\theta + 2r'_{P_1} c\theta = 2r'_{H_1},\ r'_{P_1} \le r_{P_1},\ r'_{H_1} \le r_{H_1} \tag{5b}$$

For state *r*-8 in three dimensions, we can obtain the following formulas:

$$h_4 s\theta + \left(D'_P + r'_{P_2} + r'_{P_1}\right) c\theta = \left(D'_H + r'_{H_2} + r'_{H_1}\right),$$

$$r'_{P_1} \le r_{P_1},\ r'_{P_2} \le r_{P_2},\ D'_P \le D_P,\ r'_{H_1} \le r_{H_1},\ r'_{H_2} \le r_{H_2},\ D'_H \le D_H \tag{6b}$$

If constraint conditions are formulas (5a) and (6a) or (5b) and (6b) at the same time, three-point contact states of the three dimensional peg-in-hole can be obtained.

Mapping two contact points on the peg to the plane xoy, we can obtain two corresponding points. The distance between these two points is $2r'_{P_1}$ ($2r'_{P_2}$). $2r'_{H_1}$ or $2r'_{H_2}$ represents the distance between two contact points on the left hole or on the right hole, respectively. Mapping one contact point on the left peg and the other contact point on the right peg to the plane xoy, we can obtain two corresponding points. The distance between these two points is D'_P. D'_H represents the distance between one contact points on the left hole and the other contact point on the right hole.

According to the analysis of the two dimensional dual peg-in-hole, the following results can be inferred in three dimensions. For different contact geometric dimensions (such as $r'_{P_1}, r'_{P_2}, D'_P, r'_{H_1}, r'_{H_2}$ and D'_H) of the three dimensional dual peg-in-hole, state *l*-5 and state *l*-8 show different curves, but their change trends are similar. At the point where two curves of state *l*-5 and state *l*-8 intersect, three-point contact states can exist, which are relative to the geometry of assembly objects. It means that three-point contact states, in 3D cases, can exist only when a certain set of conditions are satisfied. They are transient and considered trivial.

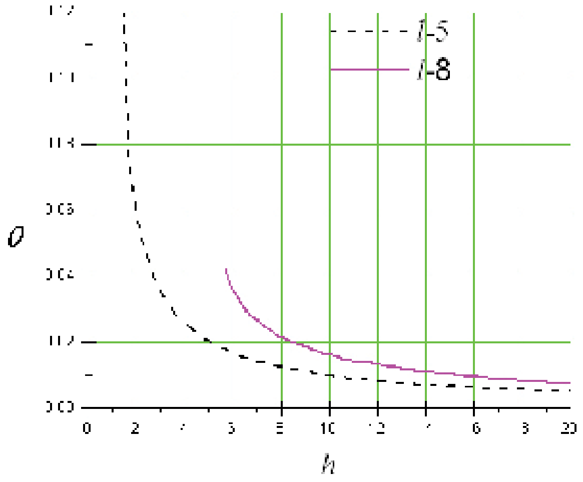


Fig. 6. θ vs *h* of state *l*-5 and state *l*-8 in two dimensions.

4. Analyzing contact force/moment of three dimensional dual peg-in-hole based on the screw theory

By understanding the physics of assembly processes, one can plan fine motion strategies and guarantee successful operations. An analysis of forces and moments during the three dimensional dual-peg insertion is important in planning fine motions. In order to insert dual pegs into their holes, insertion forces and moments $(F_x \quad F_y \quad F_z \quad M_x \quad M_y \quad M_z)$ applied to the pegs at point *o* have to satisfy a number of conditions. In this section, the conditions on applied forces and moments for maintaining each contact state are derived. Then jamming conditions are obtained. These are basic conditions of fine motions and can avoid unsuccessful insertions. The conditions which guarantee successful insertions can be obtained from the jamming diagrams.

As mentioned previously, there are 20 kinds of point contact states expect 6 kinds of line contact states. The contact states that have more than 2 contacts are ignored. According to a series of chosen contact states in Fig. 3, forces and moments can be described by screw theory (Tsaprounis, 1998). Because a quasi-static process is assumed, each of the contact states is analyzed from static equilibrium conditions. With the static equilibrium conditions and geometric constraints, the relationship between moments and forces is analyzed. The coordinate system (oxyz) is set up at the center of dual pegs and contact forces are shown in Fig. 3,4. The dual-peg is about to move down or is moving down. Examples of one-point contact state *l*-1 or state *r*-4 and two-point contact state *l*-8 or state *r*-8 are analyzed in detail. The relationship between forces and moments of other contact states can be analyzed similarly (Table 2). Table 2 shows the relationship between forces and moments of all possible point contact states about the three dimensional dual peg-in-hole.

The wrenches of the contact cases are described as follows

$$\hat{\mathbf{F}}_i = \mathbf{F}_i + \varepsilon(\mathbf{F}_i \times \mathbf{r}_i) \tag{7}$$

$$\hat{\mathbf{f}}_i = \mathbf{f}_i + \varepsilon(\mathbf{f}_i \times \mathbf{r}_i) \tag{8}$$

$$f_i = \mu F_i$$

$\hat{\mathbf{F}}_i$ is the wrench of the contact force; $\hat{\mathbf{f}}_i$ is the wrench of the friction force corresponding to the contact force. The real parts of the above equations (7) and (8) express the contact forces and the dual parts express the moments. Thus, we can obtain

$$\hat{\mathbf{F}}_i : \left(\mathbf{F}_{ix} \quad \mathbf{F}_{iy} \quad \mathbf{F}_{iz} \quad (\mathbf{F}_i \times \mathbf{r}_i)_x \quad (\mathbf{F}_i \times \mathbf{r}_i)_y \quad (\mathbf{F}_i \times \mathbf{r}_i)_z\right)$$

$$\hat{\mathbf{f}}_i : \left(\mathbf{f}_{ix} \quad \mathbf{f}_{iy} \quad \mathbf{f}_{iz} \quad (\mathbf{f}_i \times \mathbf{r}_i)_x \quad (\mathbf{f}_i \times \mathbf{r}_i)_y \quad (\mathbf{f}_i \times \mathbf{r}_i)_z\right)$$

(1) One-point contact state *l*-1 and state *r*-4:

State *l*-1: the compact forms of the wrenches of contact forces in contact points are described as follows,

$$\left(f_1 c\alpha_1 \quad f_1 s\alpha_1 \quad 0 \quad h_1 f_1 s\alpha_1 \quad -h_1 f_1 c\alpha_1 \quad x_1 f_1 s\alpha_1 - y_1 f_1 c\alpha_1\right)$$

$$\left(0 \quad 0 \quad -\mu f_1 \quad -y_1 \mu f_1 \quad x_1 \mu f_1 \quad 0\right)$$

The wrenches of the assembly forces are described as follows $(F_x \quad F_y \quad F_z \quad M_x \quad M_y \quad M_z)$.

According to the static equilibrium equations, we can obtain

$$\begin{cases} \mathrm{F_x} + f_1 c\alpha_1 = 0 \\ F_y + f_1 s\alpha_1 = 0 \\ F_z - \mu f_1 = 0 \\ M_x + h_1 f_1 s\alpha_1 - y_1 \mu f_1 = 0 \\ M_y - h_1 f_1 c\alpha_1 + x_1 \mu f_1 = 0 \\ M_z + x_1 f_1 s\alpha_1 - y_1 f_1 c\alpha_1 = 0 \end{cases} \tag{9}$$

From Eq. (9), we can obtain the following relationship equations,

$$\begin{cases} \dfrac{F_x}{F_z} = -\dfrac{c\alpha_1}{\mu} \\ \dfrac{M_y}{F_z} = \dfrac{h_1 c\alpha_1}{\mu} - x_1 \end{cases} \tag{10}$$

$$\begin{cases} \dfrac{F_y}{F_z} = -\dfrac{s\alpha_1}{\mu} \\ \dfrac{M_x}{F_z} = -\dfrac{h_1 s\alpha_1}{\mu} + y_1 \end{cases} \tag{11}$$

where $\alpha_1 \in [0^\circ \quad 90^\circ] \cup [270^\circ \quad 360^\circ]$, $r_1 : (x_1 \quad y_1 \quad -h_1)$.

State *r*-4: the wrenches of contact forces in contact points are as follows,

$$\begin{pmatrix} f_4 c\alpha_4 & f_4 s\alpha_4 & 0 & h_4 f_4 s\alpha_4 & -h_4 f_4 c\alpha_4 & x_4 f_4 s\alpha_4 - y_4 f_4 c\alpha_4 \end{pmatrix}$$
$$\begin{pmatrix} 0 & 0 & -\mu f_4 & -y_4 \mu f_4 & x_4 \mu f_4 & 0 \end{pmatrix}$$

The wrenches of the assembly forces are as follows $\begin{pmatrix} F_x & F_y & F_z & M_x & M_y & M_z \end{pmatrix}$.

According to the static equilibrium equations, we can obtain

$$\begin{cases} \mathrm{F_x} + f_4 c\alpha_4 = 0 \\ F_y + f_4 s\alpha_4 = 0 \\ F_z - \mu f_4 = 0 \\ M_x + h_4 f_4 s\alpha_4 - y_4 \mu f_4 = 0 \\ M_y - h_4 f_4 c\alpha_4 + x_4 \mu f_4 = 0 \\ M_z + x_4 f_4 s\alpha_4 - y_4 f_4 c\alpha_4 = 0 \end{cases} \tag{12}$$

Thus, we can obtain the following relationship equations,

$$\begin{cases} \dfrac{F_x}{F_z} = -\dfrac{c\alpha_4}{\mu} \\ \dfrac{M_y}{F_z} = \dfrac{h_4 c\alpha_4}{\mu} - x_4 \end{cases} \tag{13}$$

$$\begin{cases} \dfrac{F_y}{F_z} = -\dfrac{s\alpha_4}{\mu} \\ \dfrac{M_x}{F_z} = -\dfrac{h_4 s\alpha_4}{\mu} + y_4 \end{cases} \tag{14}$$

where $\alpha_4 \in [90^\circ \quad 270^\circ]$, $r_4 : (x_4 \quad y_4 \quad -h_4)$.

(2) two-point contact state *l*-8 and state *r*-8:

State *l*-8: the compact forms of the wrenches of contact forces in contact points are described as follows,

$$\begin{pmatrix} f_1 c\alpha_1 & f_1 s\alpha_1 & 0 & h_1 f_1 s\alpha_1 & -h_1 f_1 c\alpha_1 & x_1 f_1 s\alpha_1 - y_1 f_1 c\alpha_1 \end{pmatrix}$$
$$\begin{pmatrix} 0 & 0 & -\mu f_1 & -y_1 \mu f_1 & x_1 \mu f_1 & 0 \end{pmatrix}$$
$$\begin{pmatrix} f_4 c\alpha_4 & f_4 s\alpha_4 & 0 & 0 & 0 & x_4 f_4 s\alpha_4 - y_4 f_4 c\alpha_4 \end{pmatrix}$$
$$\begin{pmatrix} 0 & 0 & -\mu f_4 & -y_4 \mu f_4 & x_4 \mu f_4 & 0 \end{pmatrix}$$

The wrenches of the assembly forces are as follows, $\begin{pmatrix} F_x & F_y & F_z & M_x & M_y & M_z \end{pmatrix}$.

According to the static equilibrium equations, we obtain the expressions of moments M_y, M_x in terms of insertion forces as:

$$\frac{M_y}{DF_z} = -\frac{\mu}{c\alpha_1 - c\alpha_4}\left[\frac{x_4 - x_1}{D} + \frac{h_1 c\alpha_1}{\mu D}\right]\frac{F_x}{F_z} + \frac{\mu x_1 c\alpha_4 - \mu x_4 c\alpha_1 - h_1 c\alpha_1 c\alpha_4}{\mu D(c\alpha_1 - c\alpha_4)} \quad (15)$$

$$\frac{M_x}{DF_z} = -\frac{\mu}{s\alpha_1 - s\alpha_4}\left[-\frac{y_4 - y_1}{D} - \frac{h_1 s\alpha_1}{\mu D}\right]\frac{F_y}{F_z} + \frac{-\mu y_1 s\alpha_4 + \mu y_4 s\alpha_1 + h_1 s\alpha_1 s\alpha_4}{\mu D(s\alpha_1 - s\alpha_4)} \quad (16)$$

Eqs. (15) and (16) show linear relations, respectively. For Eq. (15), when $\frac{F_x}{F_z} = -\frac{c\alpha_1}{\mu}$, this equation reduces to Eq.(10). We can write $\frac{M_y}{DF_z} = \frac{h_1 c\alpha_1}{\mu D} - \frac{x_1}{D}$. When $\frac{F_x}{F_z} = -\frac{c\alpha_4}{\mu}$, we can write $\frac{M_y}{DF_z} = -\frac{x_4}{D}$ (in Fig. 7(a)). For Eq. (16), when $\frac{F_y}{F_z} = -\frac{s\alpha_1}{\mu}$, this equation reduces to Eq.(11). We can write $\frac{M_x}{DF_z} = -\frac{h_1 s\alpha_1}{\mu D} + \frac{y_1}{D}$. When $\frac{F_y}{F_z} = -\frac{s\alpha_2}{\mu}$, we can write $\frac{M_x}{DF_z} = \frac{y_4}{D}$ (in Fig. 7(b), Fig. 7(c)).

where $\alpha_1 \in [0^\circ \ \ 90^\circ] \cup [270^\circ \ \ 360^\circ]$, $\alpha_4 \in [90^\circ \ \ 270^\circ]$, $r_1 : (x_1 \ \ y_1 \ \ -h_1)$, $r_4 : (x_4 \ \ y_4 \ \ 0)$.

State *r*-8: the wrenches of contact forces in contact points are as follows,

$$\begin{pmatrix} f_1 c\alpha_1 & f_1 s\alpha_1 & 0 & 0 & 0 & x_1 f_1 s\alpha_1 - y_1 f_1 c\alpha_1 \end{pmatrix}$$
$$\begin{pmatrix} 0 & 0 & -\mu f_1 & -y_1 \mu f_1 & x_1 \mu f_1 & 0 \end{pmatrix}$$
$$\begin{pmatrix} f_4 c\alpha_4 & f_4 s\alpha_4 & 0 & h_4 f_4 s\alpha_4 & -h_4 f_4 c\alpha_4 & x_4 f_4 s\alpha_4 - y_4 f_4 c\alpha_4 \end{pmatrix}$$
$$\begin{pmatrix} 0 & 0 & -\mu f_4 & -y_4 \mu f_4 & x_4 \mu f_4 & 0 \end{pmatrix}$$

The wrenches of the assembly forces are as follows, $\begin{pmatrix} F_x & F_y & F_z & M_x & M_y & M_z \end{pmatrix}$.

According to the static equilibrium equations, the following equations can be obtained,

$$\frac{M_y}{DF_z} = -\frac{\mu}{c\alpha_4 - c\alpha_1}\left[\frac{x_1 - x_4}{D} + \frac{h_4 c\alpha_4}{\mu D}\right]\frac{F_x}{F_z} + \frac{\mu x_4 c\alpha_1 - \mu x_1 c\alpha_4 - h_4 c\alpha_1 c\alpha_4}{\mu D(c\alpha_4 - c\alpha_1)} \quad (17)$$

$$\frac{M_x}{DF_z} = -\frac{\mu}{s\alpha_4 - s\alpha_1}\left[-\frac{y_1 - y_4}{D} - \frac{h_4 s\alpha_4}{\mu D}\right]\frac{F_y}{F_z} + \frac{-\mu y_4 s\alpha_1 + \mu y_1 s\alpha_4 + h_4 s\alpha_1 s\alpha_4}{\mu D(s\alpha_4 - s\alpha_1)} \quad (18)$$

Eqs. (17) and (18) show linear relations. For Eq. (17), when $\frac{F_x}{F_z} = -\frac{c\alpha_1}{\mu}$, we can write $\frac{M_y}{DF_z} = -\frac{x_1}{D}$. When $\frac{F_x}{F_z} = -\frac{c\alpha_4}{\mu}$, we can write $\frac{M_y}{DF_z} = \frac{h_4 c\alpha_4}{\mu D} - \frac{x_4}{D}$. For Eq. (18), when

$\frac{F_y}{F_z}=-\frac{s\alpha_1}{\mu}$, we can write $\frac{M_x}{DF_z}=\frac{y_1}{D}$. When $\frac{F_y}{F_z}=-\frac{s\alpha_4}{\mu}$, we can write $\frac{M_x}{DF_z}=-\frac{h_4 s\alpha_4}{\mu D}+\frac{y_4}{D}$.

where $\alpha_1\in[0^\circ \quad 90^\circ]\cup[270^\circ \quad 360^\circ]$, $\alpha_4\in[90^\circ \quad 270^\circ]$, $r_1:(x_1 \quad y_1 \quad 0)$, $r_4:(x_4 \quad y_4 \quad -h_4)$.

(3) The compact forms of the wrenches of three-point contact can be described as follows,

$$\left(f_1c\alpha_1 \quad f_1s\alpha_1 \quad 0 \quad h_1f_1s\alpha_1 \quad -h_1f_1c\alpha_1 \quad x_1f_1s\alpha_1-y_1f_1c\alpha_1\right)$$
$$\left(0 \quad 0 \quad -\mu f_1 \quad -y_1\mu f_1 \quad x_1\mu f_1 \quad 0\right)$$
$$\left(f_2c\alpha_2 \quad f_2s\alpha_2 \quad 0 \quad 0 \quad 0 \quad x_2f_2s\alpha_2-y_2f_2c\alpha_2\right)$$
$$\left(0 \quad 0 \quad -\mu f_2 \quad -y_2\mu f_2 \quad x_2\mu f_2 \quad 0\right)$$
$$\left(f_3c\alpha_3 \quad f_3s\alpha_3 \quad 0 \quad h_3f_3s\alpha_3 \quad -h_3f_3c\alpha_3 \quad x_3f_3s\alpha_3-y_3f_3c\alpha_3\right)$$
$$\left(0 \quad 0 \quad -\mu f_3 \quad -y_3\mu f_3 \quad x_3\mu f_3 \quad 0\right)$$

Because three-point contact states are transient and considered trivial, their analysis is ignored.

Three dimensional contact states		Force/moment of contact states
One-point contact states	(k-1)	$\begin{cases}\frac{F_x}{F_z}=-\frac{c\alpha_1}{\mu}\\ \frac{M_y}{F_z}=j\frac{h_1c\alpha_1}{\mu}-x_1\end{cases}$ $\begin{cases}\frac{F_y}{F_z}=-\frac{s\alpha_1}{\mu}\\ \frac{M_x}{F_z}=-j\frac{h_1s\alpha_1}{\mu}+y_1\end{cases}$
	(k-2)	$\begin{cases}\frac{F_x}{F_z}=-\frac{c\alpha_2}{\mu}\\ \frac{M_y}{F_z}=(1-j)\frac{h_2c\alpha_2}{\mu}-x_2\end{cases}$ $\begin{cases}\frac{F_y}{F_z}=-\frac{s\alpha_2}{\mu}\\ \frac{M_x}{F_z}=-(1-j)\frac{h_2s\alpha_2}{\mu}+y_2\end{cases}$
	(k-3)	$\begin{cases}\frac{F_x}{F_z}=-\frac{c\alpha_3}{\mu}\\ \frac{M_y}{F_z}=j\frac{h_3c\alpha_3}{\mu}-x_3\end{cases}$ $\begin{cases}\frac{F_y}{F_z}=-\frac{s\alpha_3}{\mu}\\ \frac{M_x}{F_z}=-j\frac{h_3s\alpha_3}{\mu}+y_3\end{cases}$
	(k-4)	$\begin{cases}\frac{F_x}{F_z}=-\frac{c\alpha_4}{\mu}\\ \frac{M_y}{F_z}=(1-j)\frac{h_4c\alpha_4}{\mu}-x_4\end{cases}$ $\begin{cases}\frac{F_y}{F_z}=-\frac{s\alpha_4}{\mu}\\ \frac{M_x}{F_z}=-(1-j)\frac{h_4s\alpha_4}{\mu}+y_4\end{cases}$
	(k-5)	$\frac{M_y}{DF_z}=-\frac{\mu}{c\alpha_1-c\alpha_2}\left[\frac{x_2-x_1}{D}+j\frac{h_1c\alpha_1}{\mu D}-(1-j)\frac{h_2c\alpha_2}{\mu D}\right]\frac{F_x}{F_z}+\frac{\mu x_1c\alpha_2-\mu x_2c\alpha_1-jh_1c\alpha_1c\alpha_2+(1-j)h_2c\alpha_1c\alpha_2}{\mu D(c\alpha_1-c\alpha_2)}$ $\frac{M_x}{DF_z}=-\frac{\mu}{s\alpha_1-s\alpha_2}\left[-\frac{y_2-y_1}{D}-j\frac{h_1s\alpha_1}{\mu D}+(1-j)\frac{h_2s\alpha_2}{\mu D}\right]\frac{F_y}{F_z}+\frac{-\mu y_1s\alpha_2+\mu y_2s\alpha_1+jh_1s\alpha_1s\alpha_2-(1-j)h_2s\alpha_1s\alpha_2}{\mu D(s\alpha_1-s\alpha_2)}$
	(k-6)	$\frac{M_y}{DF_z}=-\frac{\mu}{c\alpha_3-c\alpha_4}\left[\frac{x_4-x_3}{D}+j\frac{h_3c\alpha_3}{\mu D}-(1-j)\frac{h_4c\alpha_4}{\mu D}\right]\frac{F_x}{F_z}+\frac{\mu x_3c\alpha_4-\mu x_4c\alpha_3-jh_3c\alpha_3c\alpha_4+(1-j)h_4c\alpha_3c\alpha_4}{\mu D(c\alpha_3-c\alpha_4)}$ $\frac{M_x}{DF_z}=-\frac{\mu}{s\alpha_3-s\alpha_4}\left[-\frac{y_4-y_3}{D}-j\frac{h_3s\alpha_3}{\mu D}+(1-j)\frac{h_4s\alpha_4}{\mu D}\right]\frac{F_y}{F_z}+\frac{-\mu y_3s\alpha_4+\mu y_4s\alpha_3+jh_3s\alpha_3s\alpha_4-(1-j)h_4s\alpha_3s\alpha_4}{\mu D(s\alpha_3-s\alpha_4)}$

two-point contact states	(k-7)	$\frac{M_y}{DF_z}=\frac{\mu}{c\alpha_1-c\alpha_3}\left(\frac{x_1-x_3}{D}+j\frac{-h_1c\alpha_1+h_3c\alpha_3}{\mu D}\right)\frac{F_x}{F_z}+\frac{x_1\mu c\alpha_3-x_3\mu c\alpha_1+j(-h_1c\alpha_1c\alpha_3+h_3c\alpha_1c\alpha_3)}{\mu D(c\alpha_1-c\alpha_3)}$ $\frac{M_x}{DF_z}=-\frac{\mu}{s\alpha_1-s\alpha_3}\left(\frac{y_1-y_3}{D}+j\frac{-h_1s\alpha_1+h_3s\alpha_3}{\mu D}\right)\frac{F_y}{F_z}+\frac{y_3\mu s\alpha_1-y_1\mu s\alpha_3+j(h_1s\alpha_1s\alpha_3-h_3s\alpha_1s\alpha_3)}{\mu D(s\alpha_1-s\alpha_3)}$
	(k-8)	$\frac{M_y}{DF_z}=-\frac{\mu}{c\alpha_1-c\alpha_4}\left[\frac{x_4-x_1}{D}+j\frac{h_1c\alpha_1}{\mu D}-(1-j)\frac{h_4c\alpha_4}{\mu D}\right]\frac{F_x}{F_z}+\frac{\mu x_1c\alpha_4-\mu x_4c\alpha_1-jh_1c\alpha_1c\alpha_4+(1-j)h_4c\alpha_1c\alpha_4}{\mu D(c\alpha_1-c\alpha_4)}$ $\frac{M_x}{DF_z}=-\frac{\mu}{s\alpha_1-s\alpha_4}\left[-\frac{y_4-y_1}{D}-j\frac{h_1s\alpha_1}{\mu D}+(1-j)\frac{h_4s\alpha_4}{\mu D}\right]\frac{F_y}{F_z}+\frac{-\mu y_1s\alpha_4+\mu y_4s\alpha_1+jh_1s\alpha_1s\alpha_4-(1-j)h_4s\alpha_1s\alpha_4}{\mu D(s\alpha_1-s\alpha_4)}$
	(k-9)	$\frac{M_y}{DF_z}=-\frac{\mu}{c\alpha_3-c\alpha_2}\left[\frac{x_2-x_3}{D}+j\frac{h_3c\alpha_3}{\mu D}-(1-j)\frac{h_2c\alpha_2}{\mu D}\right]\frac{F_x}{F_z}+\frac{\mu x_3c\alpha_2-\mu x_2c\alpha_3-jh_3c\alpha_3c\alpha_2+(1-j)h_2c\alpha_2c\alpha_3}{\mu D(c\alpha_3-c\alpha_2)}$ $\frac{M_x}{DF_z}=-\frac{\mu}{s\alpha_3-s\alpha_2}\left[-\frac{y_2-y_3}{D}-j\frac{h_3s\alpha_3}{\mu D}+(1-j)\frac{h_2s\alpha_2}{\mu D}\right]\frac{F_y}{F_z}+\frac{-\mu y_3s\alpha_2+\mu y_2s\alpha_3+jh_3s\alpha_3s\alpha_2-(1-j)h_2s\alpha_2s\alpha_3}{\mu D(s\alpha_3-s\alpha_2)}$
	(k-10)	$\frac{M_y}{DF_z}=\frac{\mu}{c\alpha_2-c\alpha_4}\left(\frac{x_2-x_4}{D}+(1-j)\frac{-h_2c\alpha_2+h_4c\alpha_4}{\mu D}\right)\frac{F_x}{F_z}+\frac{\mu x_2c\alpha_4-\mu x_4c\alpha_2+(1-j)(-h_2c\alpha_2c\alpha_4+h_4c\alpha_2c\alpha_4)}{\mu D(c\alpha_2-c\alpha_4)}$ $\frac{M_x}{DF_z}=-\frac{\mu}{s\alpha_2-s\alpha_4}\left(\frac{y_2-y_4}{D}+(1-j)\frac{-h_2s\alpha_2+h_4s\alpha_4}{\mu D}\right)\frac{F_y}{F_z}+\frac{\mu y_4s\alpha_2-\mu y_2s\alpha_4+(1-j)(h_2s\alpha_2s\alpha_4-h_4s\alpha_2s\alpha_4)}{\mu D(s\alpha_2-s\alpha_4)}$
When pegs tilt to the left, *k* stands for *l* and *j*=1. When pegs tilt to the right, *k* stands for *r* and *j*=0.		

Table 2. Force/moment of contact states about the three dimensional dual peg-in-hole.

5. Jamming diagrams of three dimensional dual peg-in-hole

Jamming is a condition in which the peg will not move because the forces and moments applied to the peg through the support are in the wrong proportions (Whitney, 1982).

5.1 Jamming diagrams

Because the relation between forces and moments is not proper, pegs are struck in the holes and jamming appears. In this section, according the relations between $\frac{M_y}{F_z}$ and $\frac{F_x}{F_z}$, $\frac{M_x}{F_z}$ and $\frac{F_y}{F_z}$, we can obtain three kinds of jamming diagrams about three dimensional dual peg-in-hole problems (Fig.7):

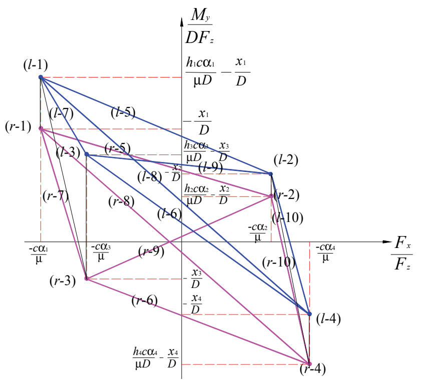
(a) $\frac{M_y}{DF_z}\propto\frac{F_x}{F_z}$; (b) $\frac{M_x}{DF_z}\propto\frac{F_y}{F_z}$, $\alpha_1\in[0^\circ\ \ 90^\circ]$, $\alpha_3\in[0^\circ\ \ 90^\circ]$, $\alpha_2\in[180^\circ\ \ 270^\circ]$, $\alpha_4\in[180^\circ\ \ 270^\circ]$; (c) $\frac{M_x}{DF_z}\propto\frac{F_y}{F_z}$, $\alpha_1\in[270^\circ\ \ 360^\circ]$, $\alpha_3\in[270^\circ\ \ 360^\circ]$, $\alpha_2\in[90^\circ\ \ 180^\circ]$, $\alpha_4\in[90^\circ\ \ 180^\circ]$. Here $\frac{M_y}{DF_z}\propto\frac{F_x}{F_z}$ represents the linear relation between $\frac{M_y}{DF_z}$ and $\frac{F_x}{F_z}$, and $\frac{M_x}{DF_z}\propto\frac{F_y}{F_z}$ represents the linear relation between $\frac{M_x}{DF_z}$ and $\frac{F_y}{F_z}$.

There are some differences between the jamming diagram of a two dimensional dual peg-in-hole and that of a three dimensional dual peg-in-hole. In two dimensions, the jamming diagram only shows the linear relation between $\frac{M_x}{F_z}$ and $\frac{F_y}{F_z}$. The jamming diagram is not

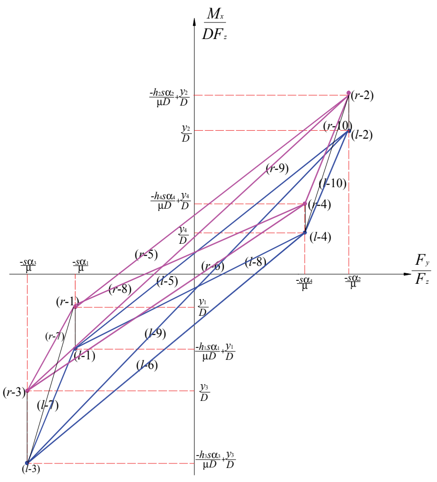
relative to α_i and contact point positions (x_i y_i). It is only relative to μ, the insertion depth and the radii of peg and hole. In three dimensions, the linear relations between $\frac{M_y}{F_z}$ and $\frac{F_x}{F_z}$, $\frac{M_x}{F_z}$ and $\frac{F_y}{F_z}$ must be considered. The jamming diagrams are not only relative to μ, the insertion depth and the radii of peg and hole, but also relative to α_i and contact point positions (x_i y_i). Because the relation between $\frac{M_x}{F_z}$ and $\frac{F_y}{F_z}$ is relative to $sin\alpha_i$, according to the different range of α_i, the jamming diagrams can be discussed in Fig. 7(b) and Fig. 7(c).
In this section, the jamming diagrams of a three dimensional dual peg-in-hole are analyzed. The range of the jamming analysis is extended. Thus, the assembly process is analyzed in detail. We can know the whole assembly process of the three dimensional dual peg-in-hole clearly.
Table 2 lists the force/moment conditions for 20 possible contact states. Fig. 7 shows the jamming diagrams of a three dimensional dual-peg problem with these contact states. When the force/moment conditions of all possible contact states of the three dimensional dual-peg insertion problem are plotted in force/moment space, the plots form a closed boundary which separates the force/moment space into two spaces; statically unstable space, and jamming space. The unstable space is the area enclosed by the boundary, and represents the conditions on the applied forces and moments that cause the pegs to unstably fall into the holes. The jamming space, on the other hand, is the area outside the boundary, and represents the force/moment conditions in which the pegs appear stuck in the holes because the applied forces and moments cannot overcome the net forces and moments due to frictional contact forces (Fig. 7). The jamming diagrams can supply us some successful insertion information. If the conditions on the applied forces and moments fall into the unstable space, the pegs can unstably fall into the holes and the continuity of insertion can be guaranteed. Since all 20 contact states cannot exist at a particular geometric condition, each jamming diagram representing a particular insertion problem is constructed from only some of the straight lines shown in Fig. 7. When the geometric dimensions and the insertion depth are known, some possible contact states can be formed and some possible jamming diagrams can be obtained (Fig. 8).

5.2 33 kinds of possible jamming diagrams

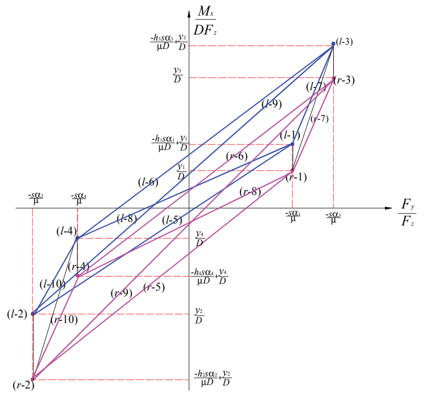
When the exact dimensions of the pegs and the insertion depth are known, possible contact states can be derived, and the jamming diagrams can be constructed. These jamming diagrams provide us with the information on how to apply forces and moments to the pegs in order to yield successful insertions. For three kinds of jamming diagrams in Fig. 7, each has 11 kinds of possible jamming diagrams. For example, for some particular insertion condition and geometric condition, the possible jamming diagrams can be constructed by joining four corresponding lines in Fig. 7 (a) forming an enclosed region (Fig. 8 a). There are several possible jamming diagrams that can be constructed in this way. In Fig. 8 a, vertexes of the quadrangle represent one-point contact states, such as (*l*-1), (*l*-2), (*r*-1), (*r*-2) in Fig. 8 (a-1). Lines of the quadrangle represent two-point contact states, such as (*l*-5), (*r*-5) in Fig. 8 (a-1).



(a) $\frac{M_y}{DF_z} \propto \frac{F_x}{F_z}$

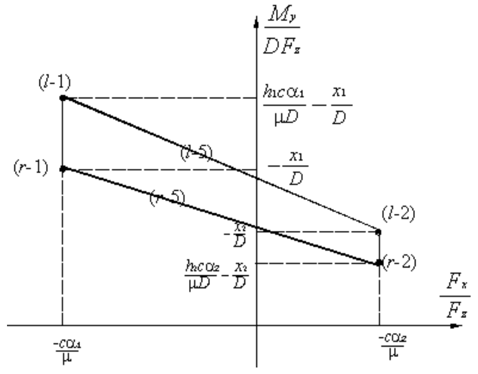


(b) $\frac{M_x}{DF_z} \propto \frac{F_y}{F_z}$, $\alpha_1 \in [0^\circ \quad 90^\circ], \alpha_3 \in [0^\circ \quad 90^\circ], \alpha_2 \in [180^\circ \quad 270^\circ], \alpha_4 \in [180^\circ \quad 270^\circ]$

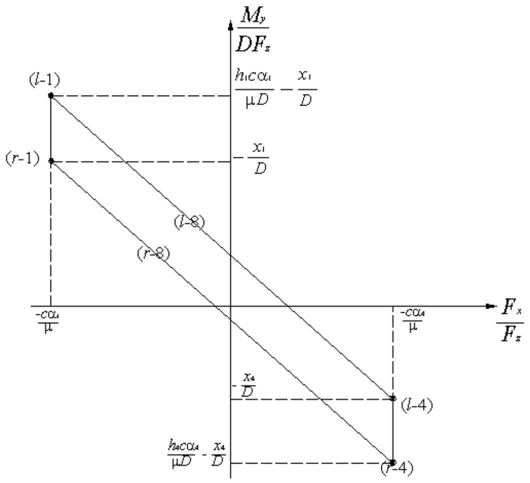


(c) $\frac{M_x}{DF_z} \propto \frac{F_y}{F_z}, \alpha_1 \in [270^\circ \quad 360^\circ], \alpha_3 \in [270^\circ \quad 360^\circ], \alpha_2 \in [90^\circ \quad 180^\circ], \alpha_4 \in [90^\circ \quad 180^\circ]$

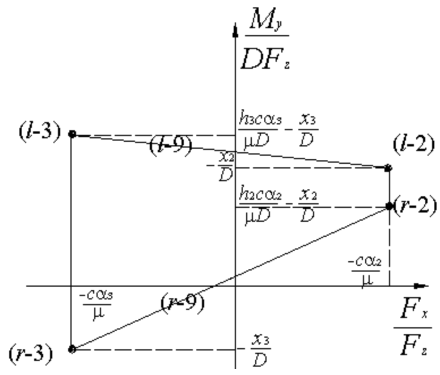
Fig. 7. Jamming diagrams.



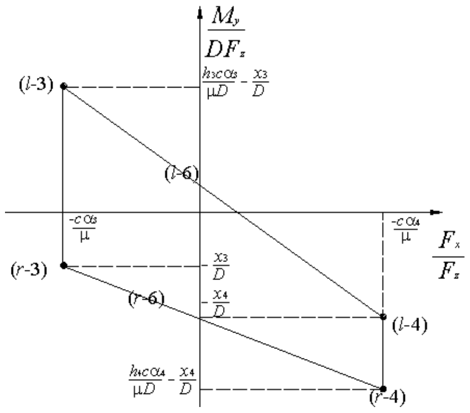
(a-1)



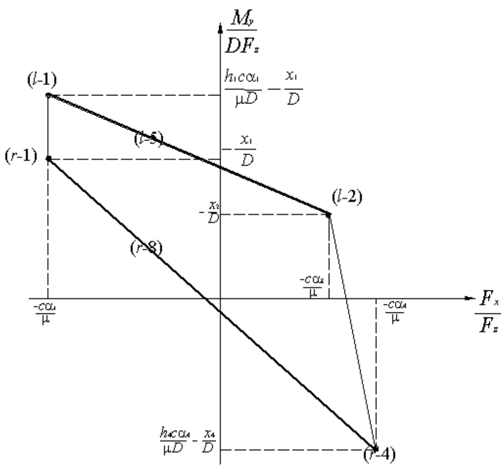
(a-2)



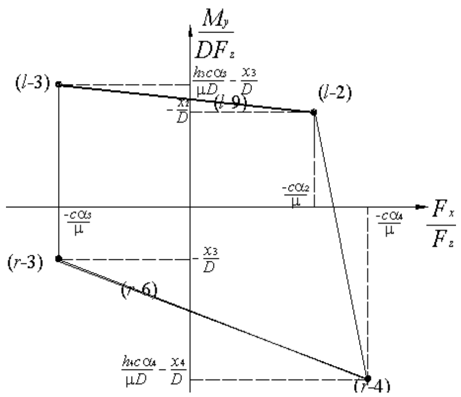
(a-3)



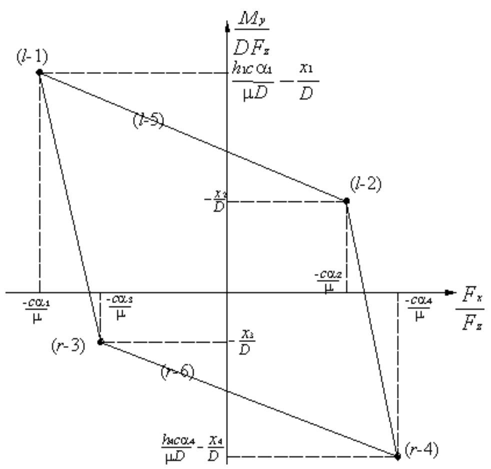
(a-4)



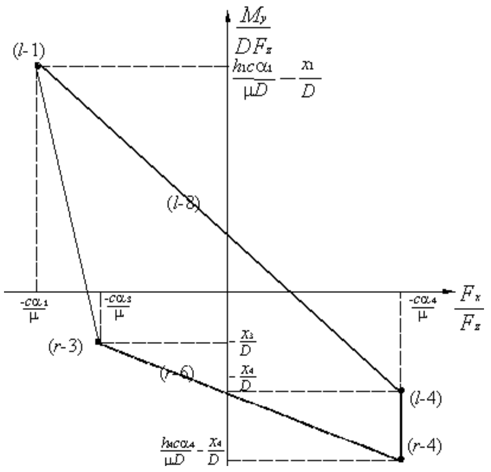
(a-5)



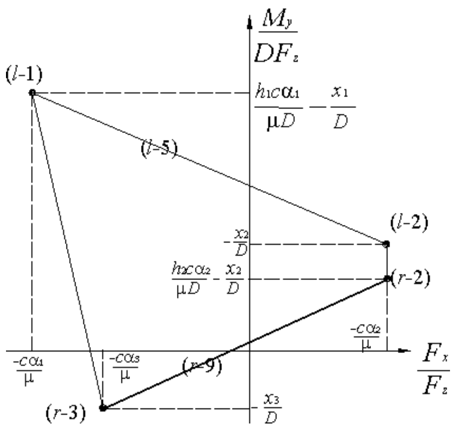
(a-6)



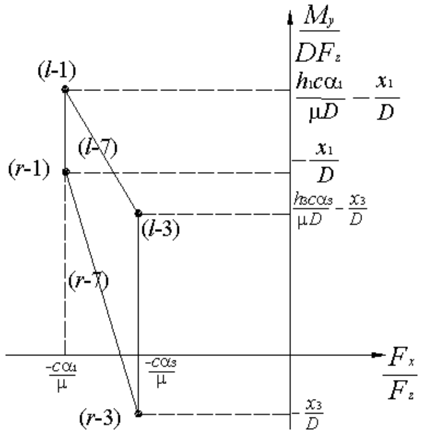
(a-7)



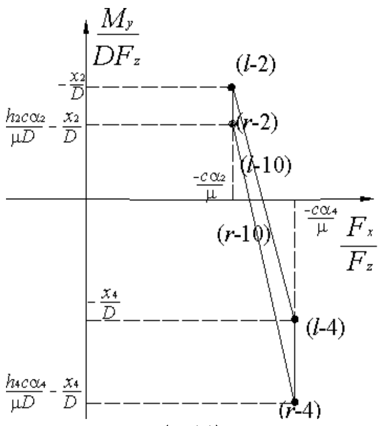
(a-8)



(a-9)

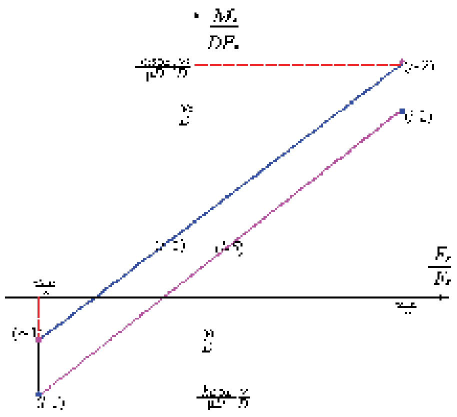


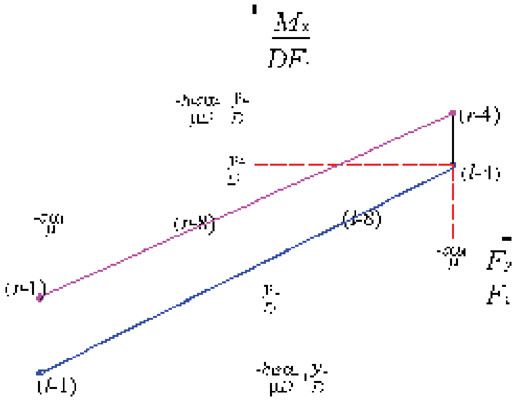
(a-10)

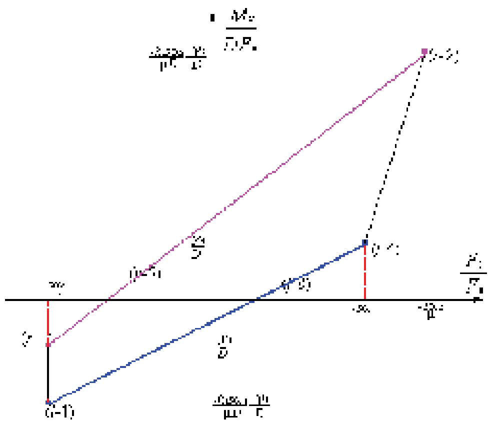


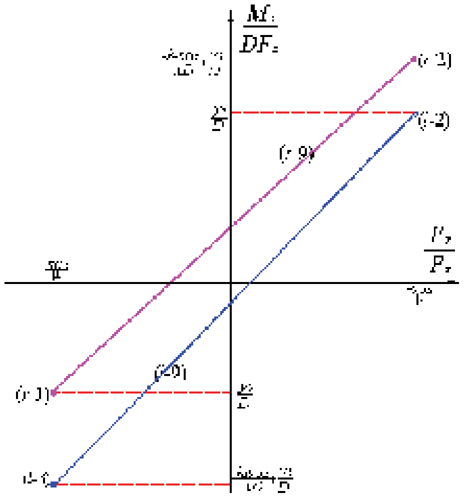
(a-11)

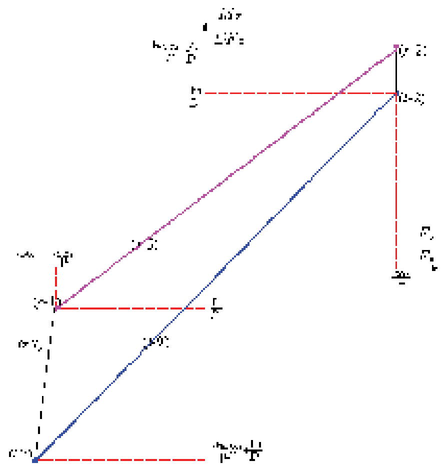
(a) Possible jamming diagrams ($\frac{M_y}{DF_z} \propto \frac{F_x}{F_z}$).

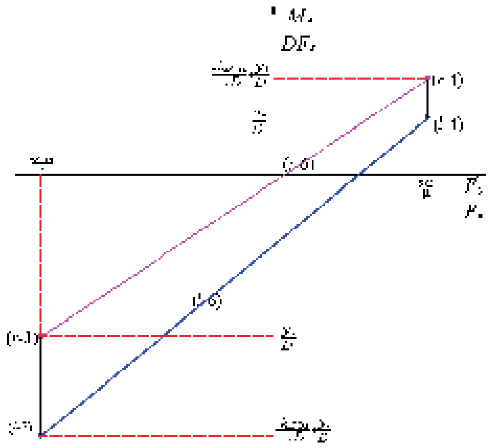

(b-1)

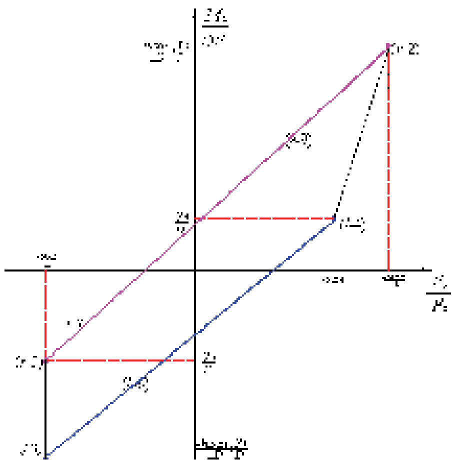

(b-2)

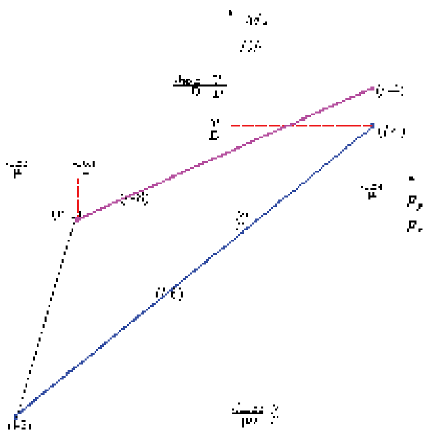

(b-3)

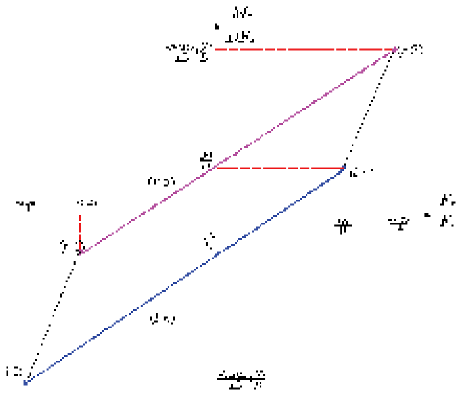

(b-4)

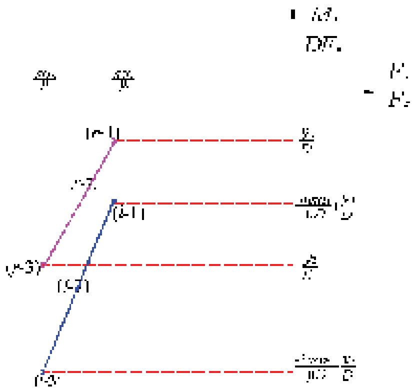

(b-5)

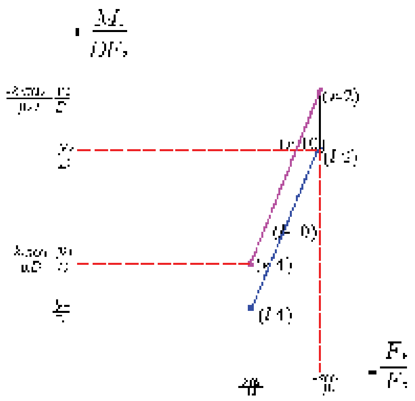

(b-6)


(b-7)

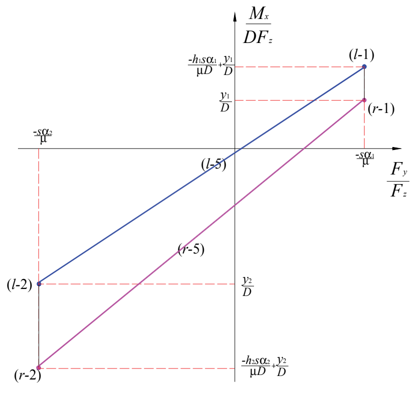

(b-8)

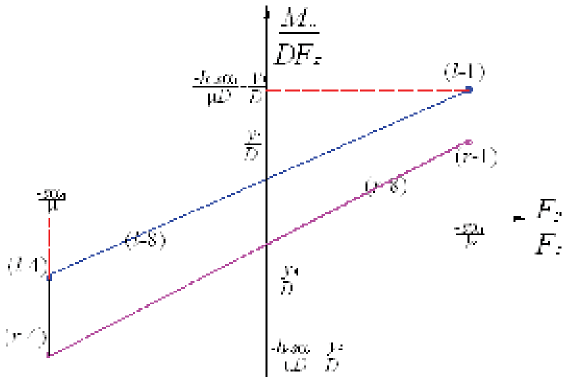

(b-9)

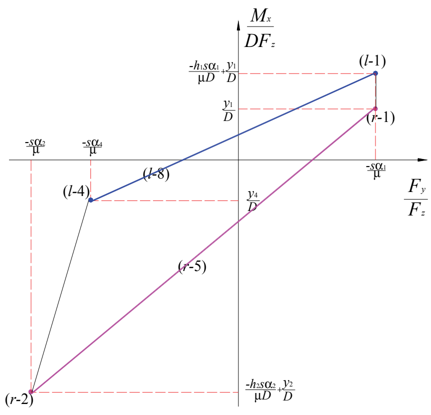

(b-10)

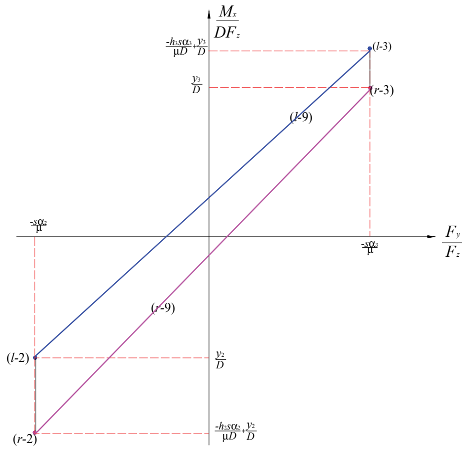

(b-11)

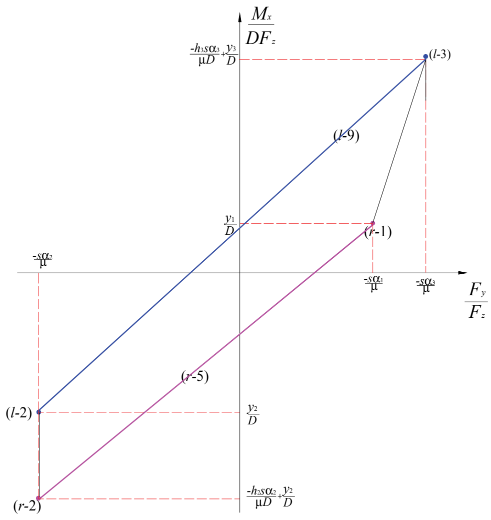
(b) Possible jamming diagrams: $\frac{M_x}{DF_z} \propto \frac{F_y}{F_z}$, $\alpha_1 \in [0° \quad 90°]$, $\alpha_3 \in [0° \quad 90°]$, $\alpha_2 \in [180° \quad 270°]$, $\alpha_4 \in [180° \quad 270°]$

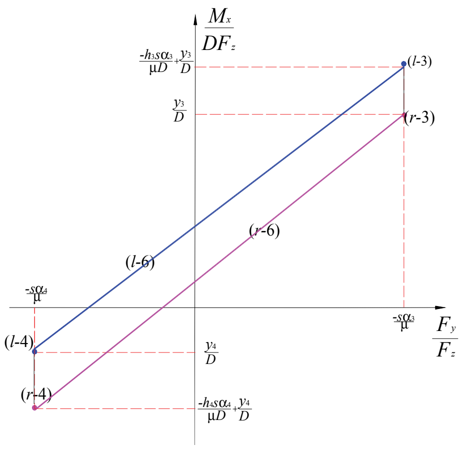

c-1

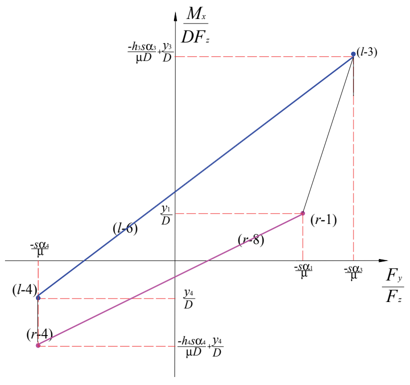

c-2


c-3

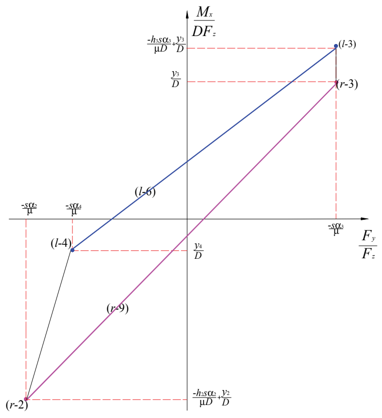

c-4


c-5

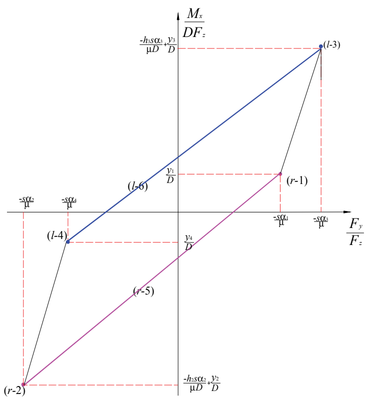

c-6



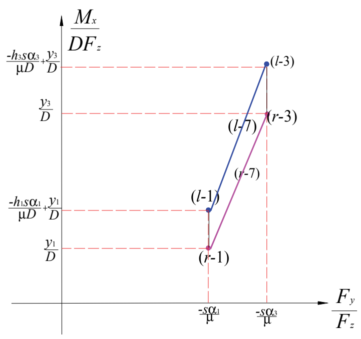
c-7



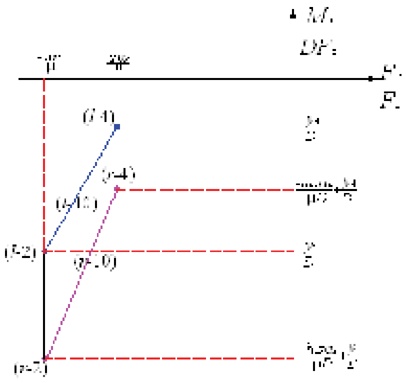
c-8



c-9



c-10



c-11

(c) Possible jamming diagrams: $\frac{M_y}{DF_z} \propto \frac{F_x}{F_z}$, $\alpha_1 \in [270° \quad 360°]$, $\alpha_3 \in [270° \quad 360°]$,

$\alpha_2 \in [90° \quad 180°]$, $\alpha_4 \in [90° \quad 180°]$

Fig. 8. Possible jamming diagrams.

In two dimensions, there are only 9 kinds of jamming diagrams, which are static and relative to μ, the insertion depth and the radii of peg and hole. In three dimensions, there are 33 kinds of jamming diagrams, which are relative to not only μ, the insertion depth and the radii of peg and hole, but also α_l and contact point positions (x_i y_i). They are dynamic. The quadrangles are more complicated than those in two dimensions. In three dimensions, the possible jamming diagrams, such as Fig.8 (a-10) and Fig.8 (a-11), may appear. Fig.8 (a-10) shows the possible jamming diagram of left-contact. Fig.8 (a-11) shows the possible jamming diagram of right-contact.
The 33 kinds of possible jamming diagrams in Fig.8 may be interpreted as follows. The vertical lines in the diagrams describe line contact states. Combinations of F_x, F_z, and M_y falling on the parallelograms'edges describe equilibrium sliding in. Outside the parallelogram lie combinations which jam the dual-peg, either in one- or two-point contact. Inside, the dual-peg is in disequilibrium sliding or falling in.

6. Conclusions

In order to know the assembly process of a dual peg-in-hole and plan an insertion strategy in detail, it is important to analyze the dual peg-in-hole in three dimensions. In this paper, the assembly contact and jamming of a dual peg-in-hole in three dimensions are analyzed in detail. Firstly, 20 contact states of the three dimensional dual-peg are enumerated and geometric conditions are derived. Secondly, the contact forces are described by the screw theory in three dimensions. With the static equilibrium equations, the relationship between forces and moments for maintaining each contact state is derived. Thirdly, the jamming diagrams of the three dimensional dual peg-in-hole are presented. Different peg and hole geometry, and different insertion depth may yield different jamming diagrams. 33 different types of possible jamming diagrams for each jamming diagram are identified. These jamming diagrams can be used to plan fine motion strategies for successful insertions. In addition, they also show that the assembly process in six degrees of freedom is more complicated than that for planar motion. The above analyses are the theoretical bases for multiple peg-in-hole insertion. It is one most significant advance in robotic assembly.

7. References

Arai T., et al (1997). Hole search planning for peg-in-hole problem, *Manufacturing Systems*, 26(2), 1997, 119-124, ISSN: 0748-948X.

Fei Y.Q. and Zhao X.F.(2003). An assembly process modeling and analysis for robotic multiple peg-in-hole, *Journal of Intelligent and Robotic Systems*, 36(2), 2003, 175-189, ISSN: 0921-0296.

McCarragher B.J. and Asada H.(1995). The discrete event control of robotic assembly tasks, *Journal of Dynamic System, Measurement and Control*, 117(3), 1995, 384-393, ISSN: 0022-0434 .

Rossitza S. and Damel B.(1998). Three-dimensional simulation of accommodation, *Assembly Automation*, 18(4), 1998, 291-301, ISSN: 0144-5154.

Simunovic S.N.(1972). Force information in assembly process, *Proc. Of the 5th Int'l Symposium on Industrial Robots*, pp. 113-126, Chicago, Illinois, 1972, IIT Research Institute Chicago, Illinois.

Sturges R.H.(1988). A three-dimensional assembly task quantification with application to machine dexterity, *The Int. Journal of Robotics Research*, 7(1), 1988, 34-78, ISSN: 0278-3649.

Sturges R.H. and Laowattana S.(1996). Virtual wedging in three-dimensional peg insertion tasks, *Journal of Mechanical Design*, 118, 1996, 99-105, ISSN: 1050-0472.

Sturges R.H. and Laowattana S.(1996). Design of an orthogonal compliance for polygonal peg insertion, *Journal of Mechanical Design*, 118, 1996, 106-114, ISSN: 1050-0472.

Sturges R.H. and Sathirakul K.(1996). Modeling multiple peg-in-hole insertion tasks, *Proc. of the Japan/USA Symposium on Flexible Automation*, pp.819-821, Conference code: 46109, Boston, MA, USA, July 7-10, 1996, ASME.

Sathirakul K. and Sturges R.H.(1998). Jamming conditions for multiple peg-in-hole assemblies, *Robotica*, 16, 1998, 329-345, ISSN: 0263-5747.

Tsaprounis C.J. and Aspragathos N.(1998). Contact point identification in robot assembly strategies under uncertainty, *Robotica*, 16, 1998, 1679-690, ISSN: 0263-5747.

Whitney D.E.(1982). Quasi-static assembly of compliantly supported rigid parts, *ASME Journal of Dynamic Systems, Measurement, and Control*, 104, 1982, 65-77, ISSN: 0022-0434.

Xiao J. (1993). Automatic determination of topological containts in the presence of sensing uncertainties, *Proc. of IEEE Int. Conf. on Robotics and Automation*, pp.65-70, ISBN: 0-8186-3450-2, Atlanta, GA, USA, May 2-6, 1993, IEEE.

Xiao J.and Ji Xuerong (2001). Automatic generation of high-level contact state space, *The Int. Journal of Robotics Research*, 20(7), 2001, 584-606, ISSN: 1050-0472.

22

Computer Vision and Fuzzy Rules Applied to a Dispensing Application in an Industrial Desktop Robot

Cristina P Santos, Manuel J Ferreira
Industrial Electronics Department, University of Minho
Portugal

1. Introduction

In an era when new product styles are being introduced with ever-shortening life cycles, the cost of high preparation times for automation equipment every time a new part or product must be produced is becoming prohibitive, both in terms of time and money. In modern flexible industrial production, the capabilities of the machinery to adapt to changing production demands are a key factor for success.
Desktop and Scara Robots are universal tools for various industrial applications like dispensing, soldering, screw tightening, pick'n place, welding or marking. This type of robots is suitable for various production line systems (e.g. cell-line, in-line), and can be adapted to meet diverse requirements. They are easy to use, can be applied economically and, nowadays, a complex programming in robot language is unnecessary, thus reducing installation time and providing added convenience. These robots are typically programmed off-line by using waypoints and path information. However, the coordinates and types of waypoints have to be entered manually or taught. Typically, small workpieces with a high complexity of linear paths raise programming efforts.
Once a robot has been programmed off-line for a workpiece, the system should be able of identifying it and autonomously initiate the correct working procedure. Further, a semi-automated system has to be capable to autonomously deal with misalignments and compensate small deviations during loading, which may result in a bad execution of the robot off-line stored programs.
The concept of sensing for robotics is essential for the design of systems where the real time operation, flexibility and robustness are major design goals. The ability of a system to sense its surroundings and perform the task according to the existing conditions is an effective way to reduce costs and raise flexibility. Highest precision and minimum amount of programming time is the result. Advanced sensor systems are now emerging in different activities which will significantly increase the capabilities of industrial robots and will enlarge their application potential.
In this work, sensor data processing concepts on the basis of fuzzy logic are applied, to enable a robot to deal autonomously with typical uncertainties of an industrial working environment. Specifically, it is proposed a flexible, adaptive and low-cost solution to address some problems which often limit the production rate of small industries. Thus, by

additional capabilities the robot can autonomously adapt to changing production needs, compensate misalignments and avoid injuries in the work tool and piece.
As a case study, consider a desktop robot executing dispensing applications on workpieces/fixtures. For each workpiece, the robot is programmed off-line. In order to improve the performance and flexibility of these industrial systems, we equipped the robot with a CCD Camera. The process is divided into two phases: a learning and an execution phase. On the learning phase, the worker programs the robot off-line such that it executes the dispensing operation over the workpiece. At this stage, an image of the workpiece is acquired, a set of descriptors describing it are computed, a fuzzy rule describing the workpiece is generated and included in a database together with the robot's program. On the execution phase, the worker loads a set of workpieces onto the robot's working table. An image is acquired, the corresponding descriptors are computed and, through a parsing and classification procedure, the workpieces are identified.
Alignments and offset values are calculated fully automatically which allows the robot to ensure accurate placement of tools. Workers stay busy loading and unloading workpieces/fixtures while a desktop robot, equipped with a vision system, is performing precision dispensing tasks. The system is also capable of autonomously starting a learning phase in case an unknown workpiece is shown to the system, and robust to deal with common errors such as a missing fixture. This significantly reduces development time for these tedious processes. Further, reduces costs by compensating misalignments in the workpiece location during loading avoiding injuries both in the workpiece and tool. Another result of this approach is that computation grounded on information derived from sensation enables the achievement of autonomy. Autonomy implies that through sensation and action it might be possible for a system to start some conceptualization process of high level.
This chapter is organized as follows. Section II describes the vision system and the software architecture for the learning and execution phases. The representation and description schemes are also described in this section as well as calibration and general image processing procedures. Section III describes the experimental results along with the hardware requirements of the system. A complete cycle of the execution phase is also depicted in this section. Finally, Section IV outlines the main conclusions and some future work is discussed.

2. The dispensing application

In a typical dispensing application the procedure is to program off-line the robot such that it executes the work over the workpiece. For each type of workpiece, a robot program is stored. This is the learning phase. During the production stage, the worker sets to run the program for the particular type of workpiece, loads the workpiece, issues a command to the robot which starts to run the dispensing application and, finally, unloads the workpiece. These procedures (execution phase) implement a full working cycle. However, two main problems may arise. Firstly, misalignments during loading may result in injuries both in the workpiece and tool. Secondly, in case other known types of workpieces are introduced in the production line, requires the worker to identify the corresponding stored robot program and load it onto the robot. This introduces delays and sometimes serious injuries due to worker failure. Further, it requires a worker able to directly interact with the robot. Such typical problems limit the production rate of small manufacturing industries.

Herein, we propose a cheap solution which improves the overall flexibility of a typical dispensing application and minimizes the two problems discussed above. Similarly to the procedure described, for each type of workpiece the robot is firstly programmed off-line and the program is stored. The main difference is that during the production stage, it is the system that autonomously identifies the loaded workpieces using visual information and a fuzzy inference classifier.

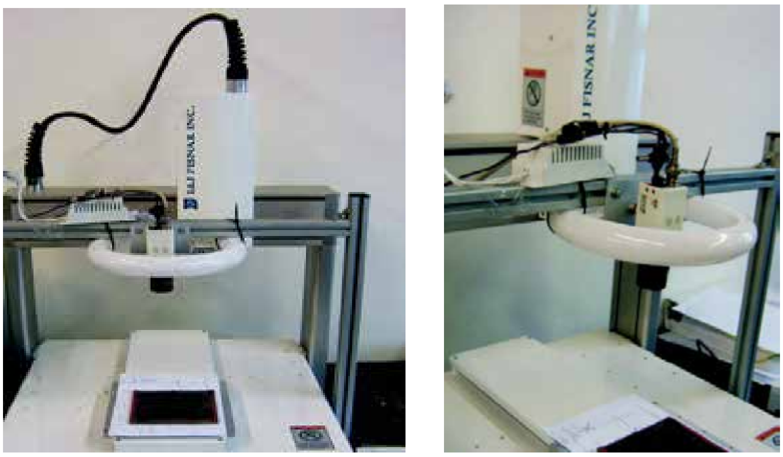
In this study, a JR2000 Series Desktop robot from I&J Fisnar Inc (Janome, 2004) with simultaneous control of 3 axis is used as the test bed. The robot performs 3D linear and arc interpolation to include compound arcs in a working area of 150x150mm. The overall experimental setup is shown in Fig. 1. A CCD camera has been mounted over the robot, and despite the applied algorithm to improve light uniformity, a fluorescent light was placed around the CCD Camera to assure that the scene is well illuminated. A front lighting technique was applied.

The CCD camera is a TRC TC5173 colour camera with a resolution of 768x576 pixels. Image digitalization is done on a general purpose image acquisition board, National Instruments PCI1411, mounted inside a 100MHz Pentium III PC. The PC is connected to the robot by a serial RS-232C protocol.

3. System Architecture

Fig. 2 presents the architecture of the processing system, in which two paths were specified: one for the learning phase (P1) and another for the execution phase (P2).

The first two modules are identical for both P1 and P2 and deal with object analysis. The Pre-processing module enhances the image quality and extracts the blob objects of the image. This module is necessary because the acquired images are not perfect for analysis. The Feature Extraction module extracts the feature vector that best characterizes each object. P1 has a Fuzzy Grammar module which generates the fuzzy rules that describe the objects. These rules are stored in a database.



a) b)

Fig. 1. Experimental setup showing the desktop robot with the mounted CCD Camera, the fluorescent lamp and a mould. a) General front view. b) Detailed view of the front lighting technique.

In the execution phase P2, the feature vectors extracted for each object are submitted to a Parsing Procedure module developed with the compilers yacc and lex (Appel, 1998; Bumble-Bee, 1999). These vectors are submitted to each rule stored in the database and a value is obtained for each of them. Finally, the Classification module verifies which rule has a higher value thus identifying the workpiece under analysis. A threshold is specified such that an alarm sounds when an unknown or misclassified workpiece is detected. Further, a learning phase is automatically initiated and an appropriate fuzzy rule is generated for that workpiece.

4. Preprocessing

To perform a robust industrial application, and prior to apply segmentation procedure to extract the diferent objects in the image, the following aspects must be minimized: 1) random noise in the acquisition process; 2) lack of light uniformity, which depends of the illumination technique; and 3) image distortions due to the type of lenses.
Noise was reduced by calculating the final image to process as an average of five consecutive acquired images.

4.1 Light calibration procedure

A light calibration procedure was developed (Russ, 1995) and employed to cope with the lack of light uniformity. A black and a white object, covering the all working area, are acquired. Each of these images is divided in non-overlapping windows of 7x7pixels and the mean of the gray-levels within each of the windows is calculated (N_{cb} and N_{cw} for the black and white windows respectively). The final histogram is calculated by the histogram stretching of each window as depicted in Fig. 3.

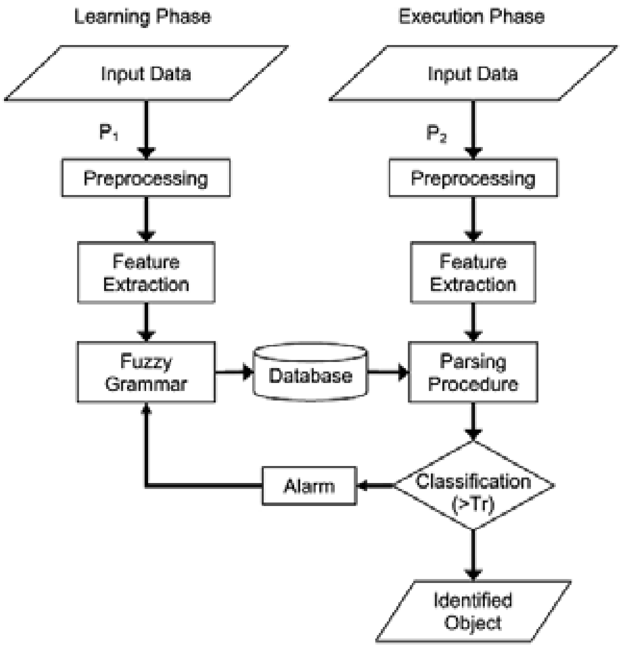


Fig. 2. Architecture of the processing system.

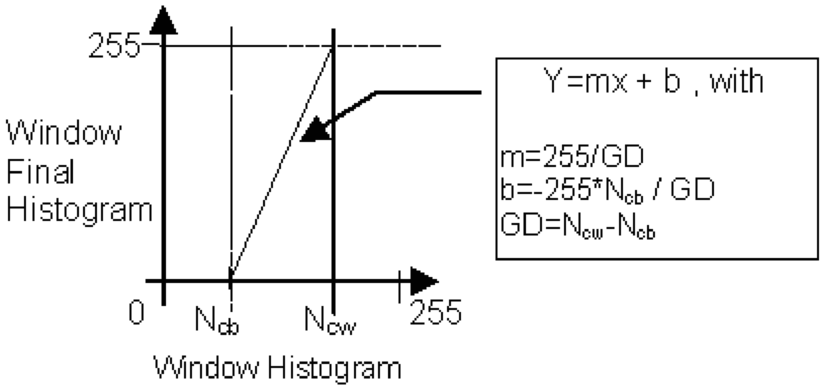


Fig. 3. Light calibration procedure.

4.2 Image distortion procedure

Image distorting is solved by applying an image correction using a well-known grid calibration procedure (Matrox, 1998). An image of a grid with size, δ_i, is acquired. Image correction is done according to the mapping between distorted, P_{di} (i=0,1,2,3), and non-distorted, P_{ndi} (i=0,1,2,3), elements of the grid (see Fig. 4). δ_i is chosen such that the sides of the distorted elements of the grid are straight lines and depends on the magnitude of distortion.

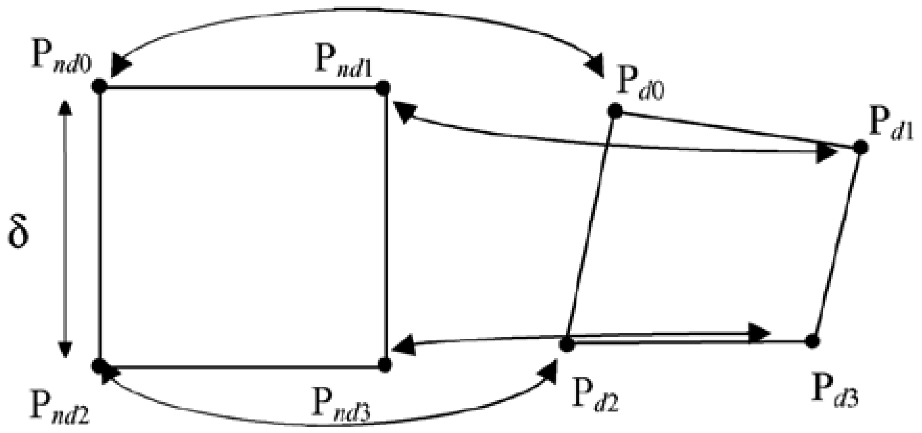


Fig. 4. Mapping between a distorted (P_{di} (i = 0, 1, 2, 3)) and a nondistorted element. (P_{ndi} (i = 0, 1, 2, 3)) of the grid. These elements have (x, y) coordinates.

The homogeneous coordinate transformation between $P_{di}=(x_{di}, y_{di})$ and $P_{ndi}=(x_{ndi}, y_{ndi})$, for i=0,1,2,3, is given by

$$w_i \begin{pmatrix} x_{ndi} \\ y_{ndi} \\ 1 \end{pmatrix} = \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix} \begin{pmatrix} x_{di} \\ y_{di} \\ 1 \end{pmatrix} \tag{1}$$

where

$$x_{ndi} = \frac{a_{11}x_{di} + a_{12}y_{di} + a_{13}}{a_{31}x_{di} + a_{32}y_{di} + a_{33}} \tag{2}$$

$$y_{ndi} = \frac{a_{21}x_{di} + a_{22}y_{di} + a_{23}}{a_{31}x_{di} + a_{32}y_{di} + a_{33}} \tag{3}$$

4.3 Segmentation procedure

The extraction of the blobs that represent the objects is accomplished through a binarization with a fixed threshold and through a blob-coloring like algorithm (Ballard & Brown, 1982). The segmentation of the image into regions could be achieved applying line finding or region growing techniques (Ballard & Brown, 1982). However, line finding techniques if followed by a floodfill procedure may produce incorrect results in non-simple connected regions. Region growing techniques commonly use only properties of local groups of pixels (local techniques). Another possibility would be split and merge techniques, however these are more complex and time consuming.
In case the image is highly contrasted and consists of dark (or white) objects in a white (or black) background, as in our case, simple local techniques can be used. In such conditions, a blob-coloring like algorithm is time effective. In the final result regions are geographically separated, meaning that each blob can be addressed in an efficient manner.
Consider the sample object illustrated in Fig. 5(a). Fig. 5(b) illustrates the corresponding extracted blob.

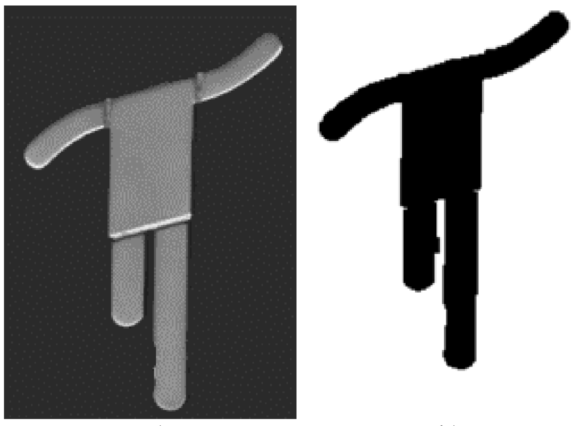


Fig. 5. A sample object and the extracted blob. a) Object sample. b) Extracted blob.

5. Feature extraction

After image segmentation, it is necessary to choose a representation and a description scheme for the resulting aggregate of segmented pixels in a form suitable for further computer processing.
As pointed out by Williams (1999), although description of a 2D shape is one of the foremost issues in pattern recognition, a satisfactory general theory does not exist. Ideally, the selected features must be invariant to the shape scaling and rotation and should support clustering or grouping; resulting in the generation of a robust representation. Low order geometric

moments are arguably the most common features. Diameter features, like Feret diameters and distance-versus-angle signatures, tend to lead to topological and symmetry considerations and are more robust to noise and small changes of shape (Ballard & Brown, 1982; Williams, 1999; Micheli-Tzanakou, 2000; Costa & Cesar, 2001; Gonzalez & Woods, 2002; Kindratenko, 2004).

The perimeter, the first and second central moments and the Feret diameter representations were tested in order to verify those that allow maximum flexibility, meaning to allow the coexistence of objects with different shapes in the same database. The best results were obtained using the Feret diameters (longest distance between any two points along the object boundary) at different rotation angles, θ, of the object, and thus were chosen to build the feature vectors of the representation scheme in our work.

Fig. 6 presents some Feret diameters for object depicted in Fig. 5(a).

By trial and error, we have chosen an increment between rotation angles of 10 degrees.

This type of external representation scheme is very useful in the computation of descriptors and is very adequate because the primary focus is on morphological features. However, this feature vector is highly dependent on the object's orientation, which poses a difficulty in the identification process. To solve this, we first orient the object by setting the axis of higher inertia always in a predefined position. Further, the fuzzy inference system implies that the magnitude of each element of the feature vector must be in the interval [0,1] and thus a normalization of the obtained feature vector is required. Therefore, normalization is achieved simply by normalizing the obtained curve to unit maximum value as given by

$$NFD(\theta) = \frac{FD(\theta)}{FD_{\max} - FD_{\min}} - \frac{FD_{\min}}{FD_{\max} - FD_{\min}} \tag{4}$$

where FD(θ) is the Feret diameter at angle θ and $FD_{\max}$, $FD_{\min}$ are the maximum and minimum value of the Feret diameters for the feature vector, respectively. The normalized Feret diameters for the objects depicted in Fig. 7 are illustrated in Fig. 8.

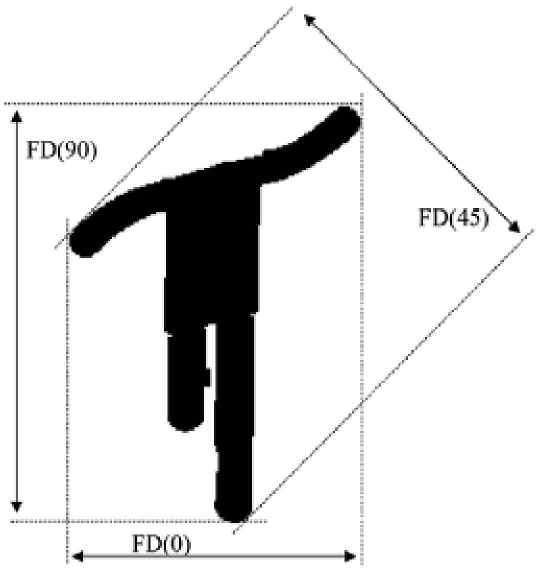


Fig. 6. The Feret diameters for object depicetd in Fig. 5 at angles 0, 45 and 90°

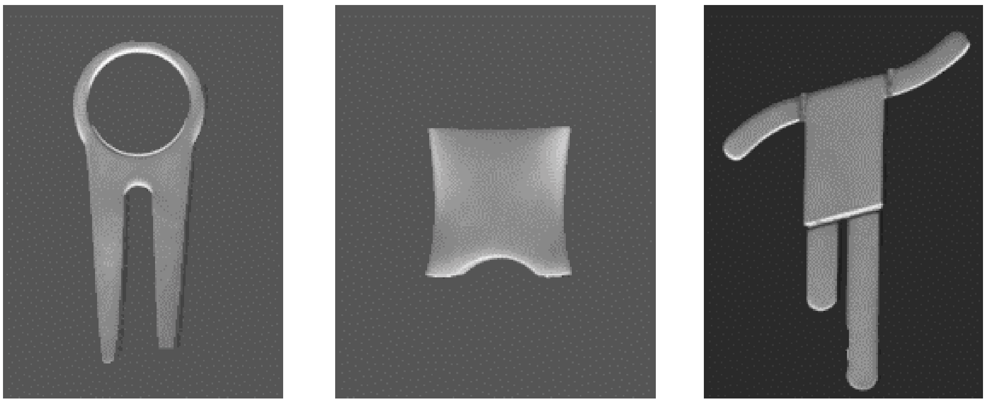


Fig. 7. Some objects used in the choice of the external representation type. a) Object 1. b) Object 2. c) Object 3

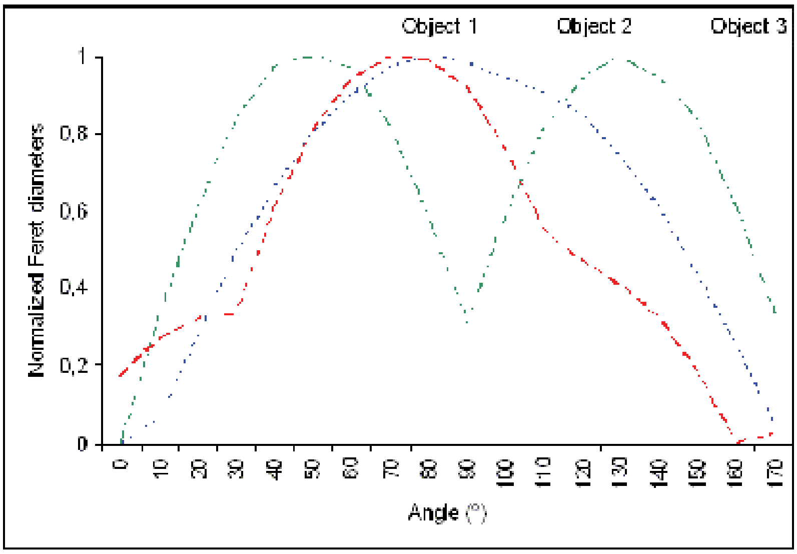


Fig. 8. Normalized Feret diameters for objects depicted in Fig 7. Slod, dash and dash-dot traces represent objects 1, 2 and 3, respectively.

Equation 4 is also independent of the size factor. For this particular application, this is a drawback since objects with different sizes require different robot programs. In order to identify objects with the same shape but with different sizes, we established a size dependent feature. We have introduced the feature S, which classifies the object's shape relatively to its size and is given by the FD_{max}, normalized to the maximum size allowed for an object.

6. Fuzzy Grammar

After the extraction of the feature vector that characterizes an object, it is necessary to classify the object according to its attributes. Specifically, our application deals with the following constraints: a) to deal with a high diversity of objects; b) to recognize

simultaneously several different type of objects and c) to autonomously detect a new type of objects during the execution phase and thus initiate a learning phase, using the intervention of the human operator only to program the robot. To accomplish these goals the learning phase of the recognition process must be done with a unique sample of each type of object.
Regarding the classifiers and recognizers, there are different approaches based on statistical and artificial intelligence methods (Bezdek & Pal, 1992; Kerre & Nachtegael, 2000; Micheli-Tzanakou, 2000; Costa & Cesar, 2001; Looney, 2002). The most common solutions commercially available use recognizers based on the calculus of metrics like Euclidean, Minkowsky e Mahalanobis distance measures (Williams, 1999). However, these recognizers, as well as the ones based on neural, fuzzy logic and neurofuzzy networks, demand a great amount of samples from the population to perform learning. Despite the fact that these modern technologies are now firmly established as leading advanced control techniques in use in industry, they do not fulfil the constraints of the dispensing application.
In this work, a fuzzy system modelling approach was developed in which a fuzzy inference system identifies the fuzzy rules representing relationships among 2D shape features. There are several approaches that generate these fuzzy rules. The most often applied are based on statistics, neural networks and genetic algorithms (Ivancic & Malaviya, 1998; Peters et al., 1998; Looney, 2002).
However, none of these methods satisfy the needs of this specific application. Therefore, we decided to apply a fuzzy grammar approach. Fuzzy grammar is a pattern classification syntactic model used to represent the structural relations of patterns (Fu & Booth, 1986ab; Bezdek & Pal, 1992; Malaviya, 1996; Stanchev & Green, 2000) and describe the syntax of the fuzzy languages that generate the fuzzy rules. This inference system fulfils the project demands due to its linguistic description approach, which keeps the number of rules small, and due to its capability to generate a fuzzy rule using only one sample of a pattern.
Herein, we briefly review some basic concepts of fuzzy grammar (for a full discussion see (Lee & Zadeh, 1969; Pal & Majumber, 1986; Bezdek & Pal, 1992; Yager & Zadeh, 1992)). Fuzzy grammar GF is a quintuple $GF=(V_N,V_T,P,S_0,\mu)$, in which V_N and V_T are finite disjoint sets of non-terminal and terminal vocabulary respectively, such that $V=V_N\cup V_T$ is the total vocabulary of the grammar. P is a finite set of production rules of the type $\alpha\rightarrow\beta$, with $\alpha\in V_N$ and β is a member of the set V^* of all strings (including the null string ε). $S0\in V_N$ is the starting symbol. $\mu$is the mapping of $P\rightarrow[0,1]$, such that $\mu(p)$ denotes the possibility of the current language sentence $p\in P$.
The syntax of the developed language L(GF) is depicted in Fig. 9 and includes 4 different steps:

1) The codification of the features to primitives. In this work, the features are the Feret diameters ($NFD(\theta)$) and the size S, which are coded to the primitives $FD\theta$ and SN, respectively. When more than one sample of an object is presented to the system the mean value of each feature is used.
2) The definition of linguistic terms HistVar:#. This setting is done according to Table 1. The membership function Π is illustrated in Fig. 10 for $\Pi(x,b,c)$. The parameter c is chosen such that the eleven membership functions cover the all universe of discourse X and have disjoint maximums.
3) The definition of fuzzy modifiers (FM): "More than", "Less than" and "Between". The FM "More than" LT is defined by

$$\mu_{MT}\langle LT\rangle = \begin{cases} 1 & x \geq L \\ S(x, L-lb, L-lb/2, L & x < L \end{cases} \tag{5}$$

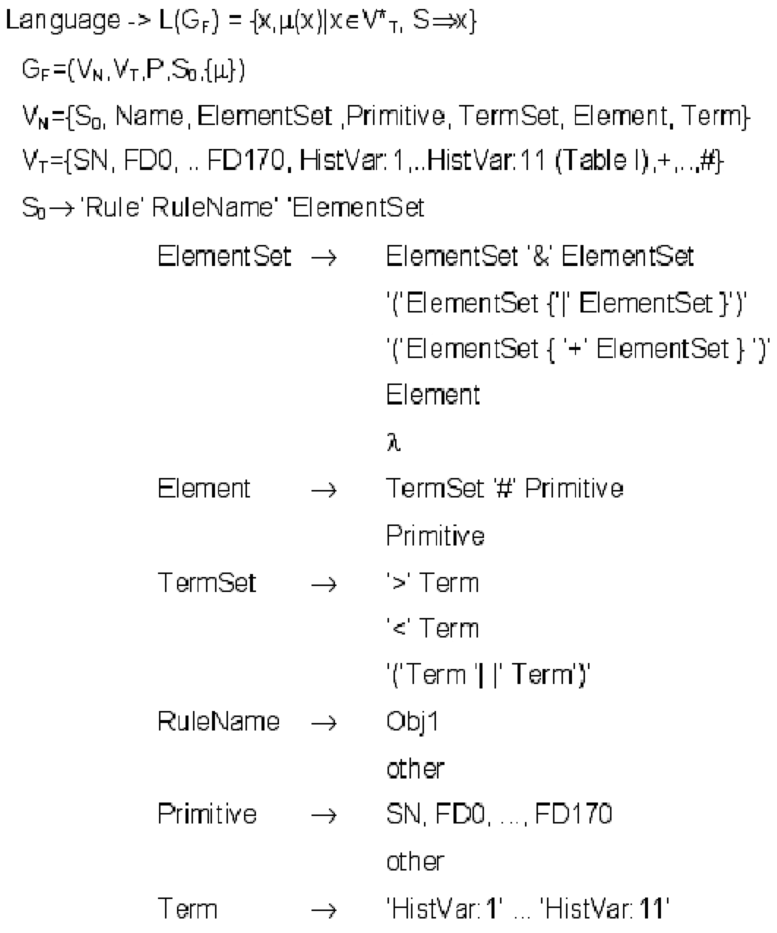
where L is a threshold value and lb is the bandwidth value of the S membership function (Bezdek & Pal, 1992; Malaviya, 1996). The FM "Less than" LT is given by

$$\mu_{LT}\langle LT\rangle = \begin{cases} 1 & x \leq L \\ 1 - S(x, L, L+lb/2, L+lb & x > L \end{cases} \tag{6}$$

The FM "Between" LT1 e LT2, is given by

$$\mu_B < TL_1 >< TL_2 > \; = \begin{cases} 1 - S(x, w_1, w_1 + lb/2, w_1 + lb) & x > w_1 \\ 1 & w_2 \leq x \leq w_1 \\ S(x, w_2 - lb, w_2 - lb/2, w_2) & x < w_2 \end{cases} \tag{7}$$

where w1 and w2 are threshold values (Bezdek & Pal, 1992; Malaviya, 1996).


Language -> $L(G_F) = \{x, \mu(x) | x \in V^*_T, S \Rightarrow x\}$

$G_F = (V_N, V_T, P, S_0, \{\mu\})$

V_N={S_0, Name, ElementSet ,Primitive, TermSet, Element, Term}

V_T={SN, FD0, .. FD170, HistVar: 1,..HistVar: 11 (Table I),+,..,#}

$S_0 \rightarrow$ 'Rule' RuleName' 'ElementSet

ElementSet	$\rightarrow$	ElementSet '&' ElementSet
		'('ElementSet {'|' ElementSet }')'
		'('ElementSet { '+' ElementSet } ')'
		Element
		λ
Element	$\rightarrow$	TermSet '#' Primitive
		Primitive
TermSet	$\rightarrow$	'>' Term
		'<' Term
		'('Term '| |' Term')'
RuleName	$\rightarrow$	Obj1
		other
Primitive	$\rightarrow$	SN, FD0, ..., FD170
		other
Term	$\rightarrow$	'HistVar: 1' ... 'HistVar: 11'

Fig. 9. Syntax of the developed fuzzy language L(GF).

Designation	Function
HistVar:1	Π(x,0.2,0.0)
HistVar:2	Π(x,0.2,0.1)
HistVar:3	Π(x,0.2,0.2)
HistVar:4	Π(x,0.2,0.3)
HistVar:5	Π(x,0.2,0.4)
HistVar:6	Π(x,0.2,0.5)
HistVar:7	Π(x,0.2,0.6)
HistVar:8	Π(x,0.2,0.7)
HistVar:9	Π(x,0.2,0.8)
HistVar:10	Π(x,0.2,0.9)
HistVar:11	Π(x,0.2,1.0)

Table 1. Linguistic terms used on the fuzzy grammar.

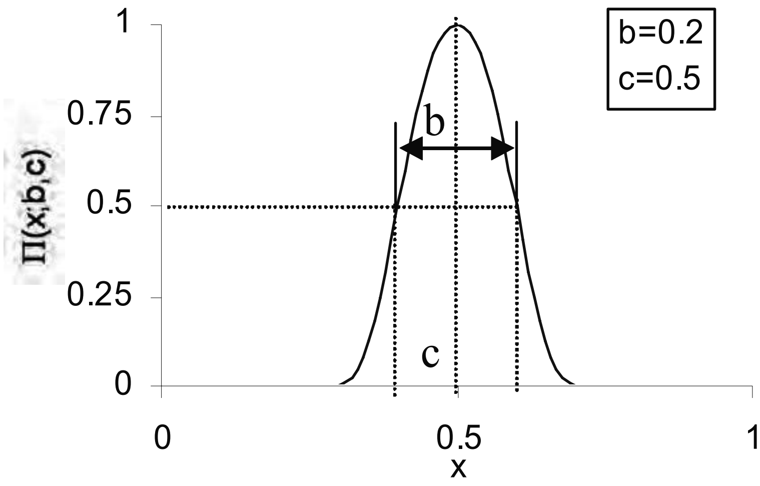


Fig. 10. Membership function PI.

4) The definition of fuzzy operators (FO) which define the relations between the linguistic terms and primitives. We defined the following FO:
   a) &, representing the AND of two primitives. It is given by the Yager intersection (Pal & Majumber, 1986).
   b) >, representing "More than" LT and is given by μMT<LT>.
   c) <, means "Less than" LT and is given by the function μLT<LT>.
   d) | |, describes "Between two" LT and is given by μB<LT1><LT2>.
   e) #, means a "Separator between a" primitive and a LT.
   f) ( ), imposes a hierarchy in the rule.

Consider as an example object 2 depicted in Fig. 7. Fig. 11 illustrates the primitive FD20, obtained from the Feret diameter feature, NFD(θ)=0.6, when θ=20 degrees. This primitive has non-zero degrees of membership for LT HistVar:6, LT HistVar:7 and LT HistVar:8 (Fig. 11). The highest fuzzy value is obtained using LT HistVar:7. Thus, HistVar:7#FD20 is part of the fuzzy rule which characterizes object 2. Finally, the rule created by the fuzzy grammar is:

HistVar:1#FD0&HistVar:4#FD10&HistVar:7#FD20&#HistVar:9#FD30&HistVar:11#FD40&
HistVar:1#FD50&HistVar:11#FD60&HistVar:9#FD70&HistVar:7#FD80&HistVar:4#FD90&#
HistVar:7#FD100&HistVar:9#FD110&>HistVar:10#FD120&HistVar:11#FD130&>HistVar:10
#FD140&HistVar:9#FD150&HistVar:7#FD160&HistVar:4#FD170&HistVar:2#SN.

If more than one linguistic term gives a fuzzy value superior to 0.75; we apply fuzzy modifiers like "More than", "Less than" and "Between", to combine the obtained results. Fig. 12 illustrates the procedure for fuzzy modifier "More than" HistVar:10 for the primitive FD140. The final fuzzy value results from the combination of LT HistVar:10 and LT HistVar:11. Similar procedures apply for fuzzy modifiers "Less than" (Fig. 13) and "Between" (Fig. 14).

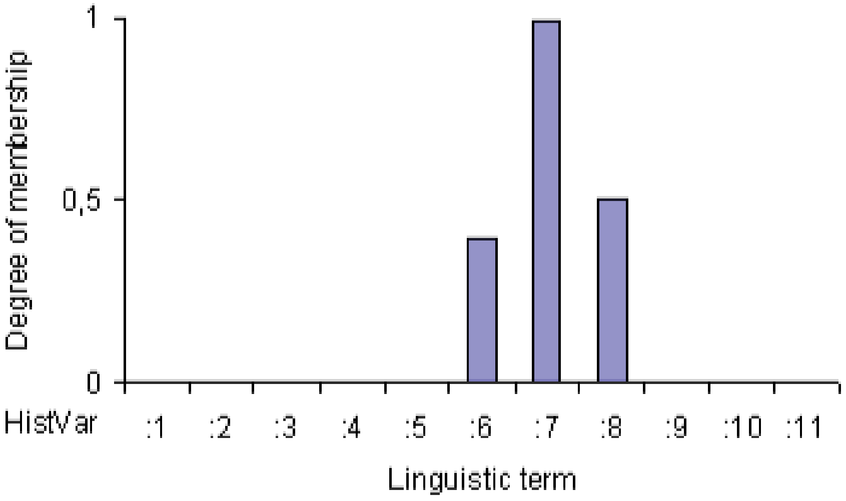


Fig. 11. The highest fuzzy value for LV FD20 is obtained using LT HistVar:7.

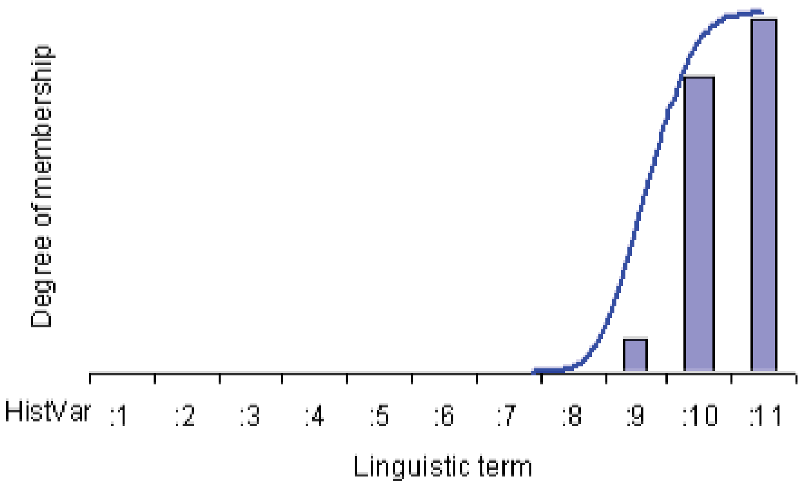


Fig. 12. Linguistic term for the primitive FD140 – Fuzzy Modifier „More than" HistVar:10.

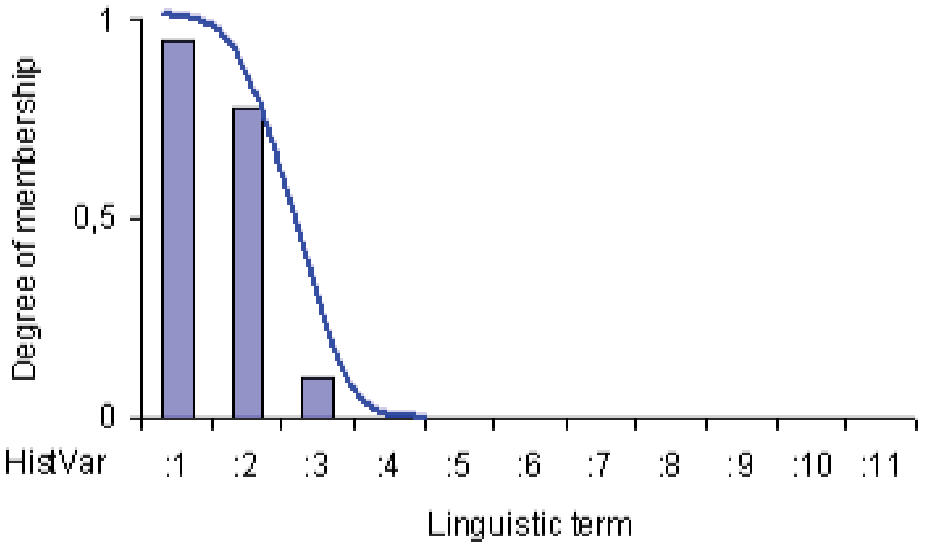


Fig. 13. Fuzzy Modifier „Less than" HistVar:2.

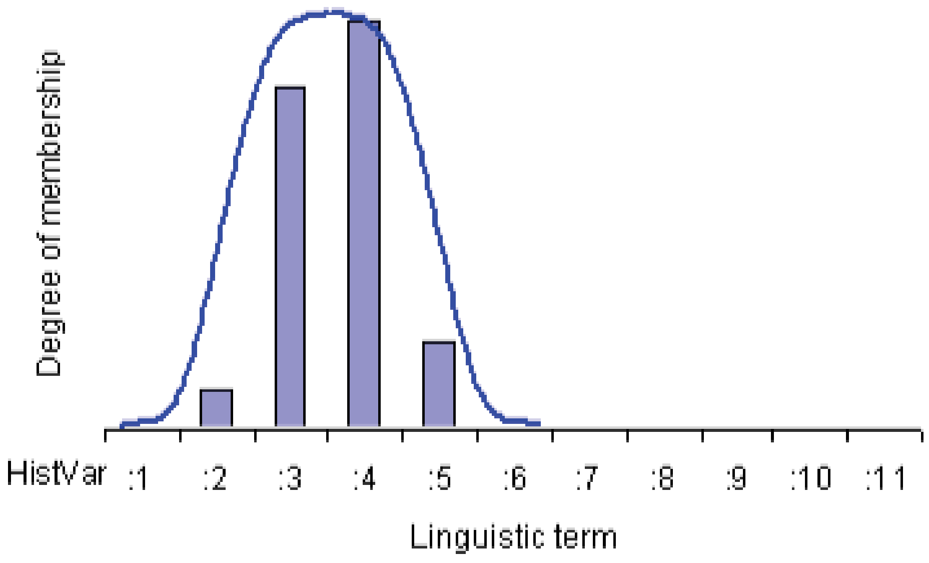


Fig. 14. Fuzzy Modifier „Between" HistVar:3 and HistVar:4.

7. Parsing procedure

The parsing procedure was developed for the fuzzy grammar. The inputs are the feature vectors extracted for an object from the Feature Extraction module and the rules stored in the database. The feature vectors are submitted to each rule of the database. The output of

the parsing procedure is a value in the interval [0,1] reflecting the grade of membership of the object for each class.
Consider as a simple example the feature vectors which are presented in Table 2. They describe the objects depicted in Fig. 7.

Angle (º)	Object 1	Object 2	Object 3
0	0,00	0,00	0,17
10	0,06	0,33	0,27
20	0,28	0,61	0,32
30	0,48	0,83	0,33
40	0,65	0,97	0,60
50	0,79	1,00	0,80
60	0,89	0,97	0,93
70	0,97	0,80	1,00
80	1,00	0,58	0,98
90	0,98	0,30	0,91
100	0,94	0,58	0,75
110	0,90	0,80	0,54
120	0,84	0,94	0,46
130	0,73	1,00	0,40
140	0,60	0,94	0,32
150	0,43	0,8	0,18
160	0,25	0,61	0.00
170	0,04	0,33	0,02

Table 2. Feature Vectors for the Objects depicted in Fig. 7.

If we consider a database only made with the objects' rules depicted in Fig. 7, the output results of the parsing procedure are presented in Table 3.

	Rule Object 1	Rule Object 2	Rule Object 3
Object 1	0,89	0,00	0,00
Object 2	0,00	0,9	0,00
Object 3	0,00	0,00	0,89

Table 3. Results of parsing procedure.

8. Classification

This module uses the output of the parsing and verifies which rule produces the higher value for the feature vector. For the example of Table 3 the result of the classification procedure is shown in Table 4.

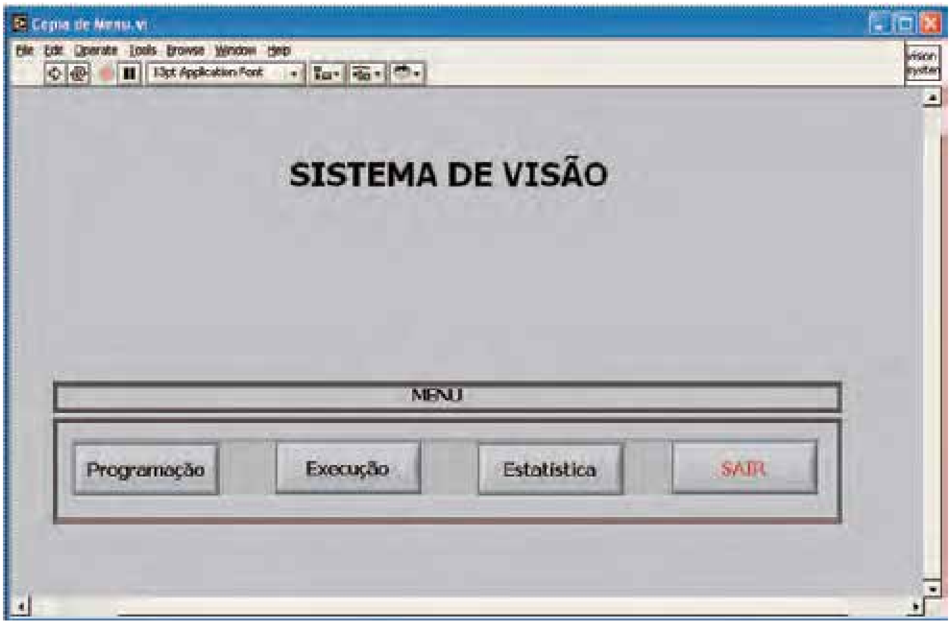
	Classification	
	Rules higher result	Object Type
Object 1	0.89	1
Object 2	0.9	2
Object 3	0.89	3

Table 4. Results of classification procedure.

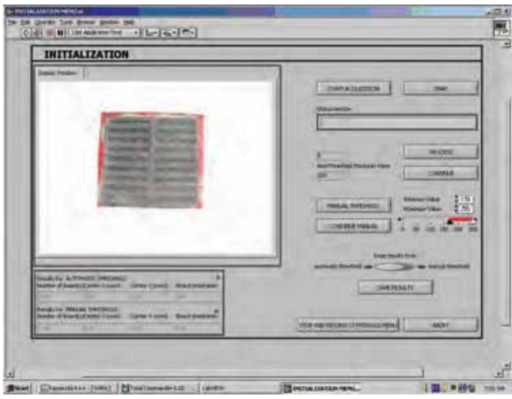
If this value is less than a defined threshold (Tr of Fig. 2) it is assumed that a new type of object is present. In such case, the feature vector that characterizes this new object is submitted to the fuzzy grammar module in order to generate the new appropriated fuzzy rule.

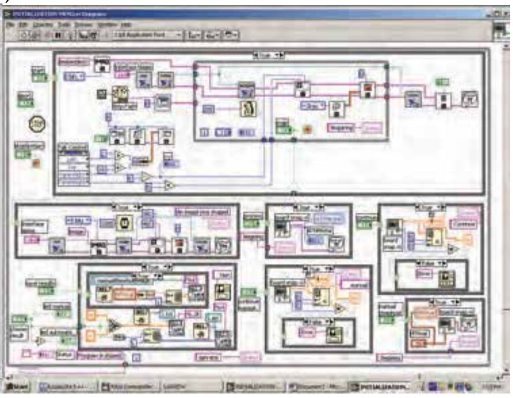
9. Experimental Results

The commercial software LabView 6.1 with IMAQ 6.0 was used, in order to increase the processing speed and to reduce the development time. This was also a requirement from the company that supports the development of the dispensing application. The fuzzy grammar was developed in C++. To call the fuzzy grammar from the LabView a DLL was created to encapsulate the C++ code. Fig. 15 illustrates the three principal LabView panels and vi diagrams of the application.

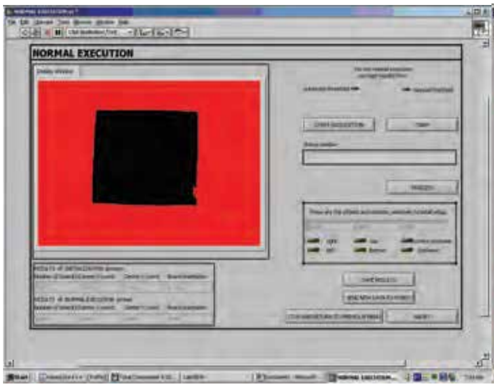


a)

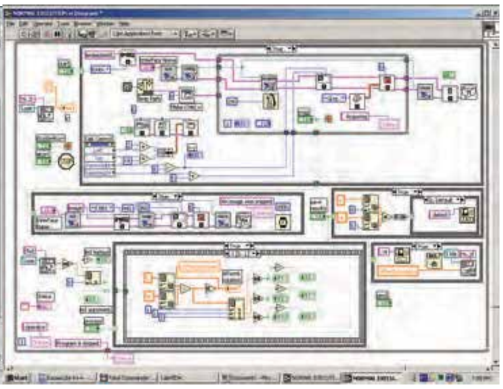




b) c)



d)



e)

Fig. 15. Different panels of the developed application. A selection must be done among: learning, execution phase and a statistical option for showing statistical data. a) Main panel. Learning phase: b) front panel, c) vi diagram. Execution phase: d) front panel, e) vi diagram.

The robot program that the robot performs over the workpiece, is sent to the robot via RS-232C protocol under the control of the JR software (trademark). However, the formats of the file that contains both the robot program and the information send to the RS-232 port are not known. This constraint was overcome through the development of a DLL that establishes the communication between LabView and the JR software. This DLL sends Microsoft Windows messages to the JR software, providing the appropriate commands to change the robot software. Finally, a message with the start command is sent to the JR software in order to initiate the robot program. JR software must be running during learning and execution phases. This development enables the robot to be controlled by the computer vision software.

9.1 A complete cycle

The feasibility and efficiency of the used approach have been studied by performing a set of experiments using 10 different types of objects (see Fig. 16).

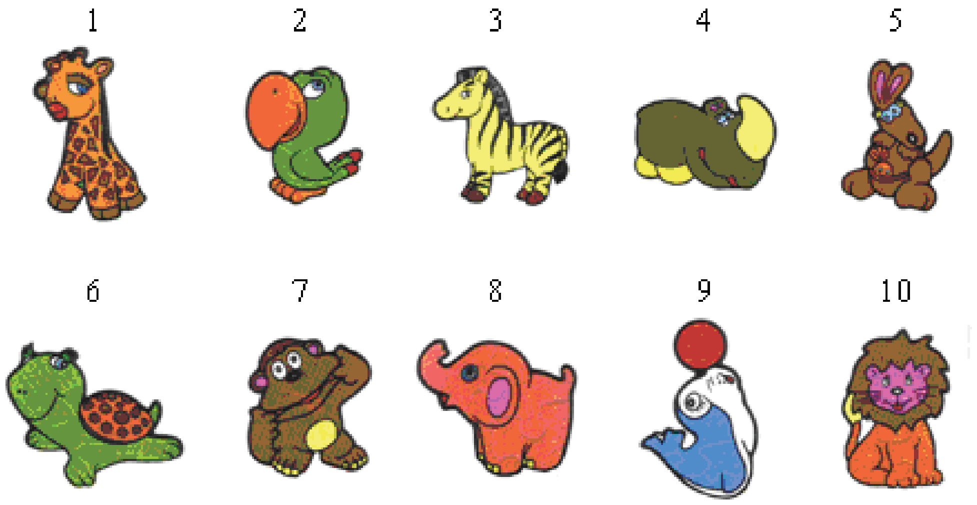


Fig. 16. Objects used in final experiment.

During the learning phase, for each object, the used robot program, the generated fuzzy rule, the orientation and the position are stored together in the database. On the execution phase, these workpieces are presented to the system having different positions and orientations relatively to the learning phase. The developed approach was able to identify each workpiece. Rotations, R, alignments and offset values in x, y were calculated, the robot's stored programs were adjusted accordingly and sent to the robot. Finally, the robot executed the changed program over each workpiece. The minimum offset that the system was able to calculate was as small as 0.2mm. The minimum rotation was 3 degrees.

Second column of Table 5 shows the generated linguistics terms for each primitive in the learning phase for object 10 of Fig. 16. An identical object but rotated of 40 degrees and with an offset in position of (x,y)=(10,15)mm was processed during the execution phase. Third and fourth column of Table IV show the obtained primitives and Linguistic terms, respectively. The classification result for the object 10 rule is 0.90, whereas for the other objects is 0.0. The calculated offset and orientation was of (10.0,15.1)mm and 39.3 degrees, respectively.

Table 6 shows the percentage of good classifications when each object is placed with 20 different locations and orientations. In some cases, objects were classified as Not Available in Database (NAD).

	LT	Primitive value	LT value
Primitive	(Learning phase)	(Execution phase)	(Execution phase)
FD0	HistVar:1	0.00	1.00
FD10	HistVar:2	0.13	0.92
FD20	HistVar:5	0.38	0.99
FD30	HistVar:7	0.60	1.00
FD40	HistVar:9	0.75	0.90
FD50	HistVar:10	0.91	0.98
FD60	HistVar:11	1.00	1.00
FD70	HistVar:11	1.00	1.00
FD80	HistVar:10	0.93	0.92
FD90	HistVar:9	0.84	0.90
FD100	HistVar:10	0.85	0.90
FD110	HistVar:10	0.94	0.90
FD120	HistVar:11	0.96	0,94
FD130	HistVar:10	0.91	0.98
FD140	HistVar:9	0.78	0.99
FD150	HistVar:7	0.59	0.99
FD160	HistVar:5	0.35	0.90
FD170	HistVar:3	0.17	0.96
SN	HistVar:5	0.36	0.91

Table 5. Example of execution data for object 10 from Fig. 16.

Object	1	2	3	4	5	6	7	8	9	10	NAD
1	95										5
2		100									0
3			100								0
4				95							5
5		5			90						5
6						95					5
7							90			5	5
8								95			5
9									100		
10										95	5

Table 6. Classifications of Objects (IN %).

As we can see from the above results, the developed approach can be applied when objects have different locations and orientations and only one sample was used during the learning phase. The advantage is that a high percentage of type of objects (greater than 90%), when submitted to rules of objects of other types, gives 0 as a result. This means that the system creates disjoints rules and assures a good classification.

10. Conclusion

In this work, we have used sensing technology to endow an industrial Desktop robot with a greater degree of flexibility in dealing with its environment. The goal was an adaptive, flexible, low-cost solution to maximize efficiencies in dispensing applications. Concretely, a CCD Camera was mounted over the robot and the visual information was used to autonomously change a previously off-line stored robot program to each workpiece.
The results illustrate the flexibility and robustness of the overall application. Further, the employed approach assures a good classification of workpieces and a minimum offset and rotation values of 0.2 mm and 3 degrees, respectively.
To further improve the classification procedure we intend to introduce new features in the rules and to experiment other methods than fuzzy logic. The overall application can be improved in such a way that the robot's program could also be automatically generated through the extraction of the relevant waypoints and path information.
The solution proposed can easily be extended to other type of machinery applications. For instance, to quality control inspection procedures including: dimensional measurement and gagging, verification of the presence of components, hole location and type of holes, detection of surface flaws and defects. It can also be easily extended to other categories of machine vision applications, in fact this approach was already applied to texture segmentation for tracking purposes (Ferreira, 2006). This application differs from the one that was presented here only by the type of features extracted from the images.

11. References

Appel, A.W. (1998), *Modern compiler implementation in C*, Cambridge University Press, ISBN: 0-521-60765-5, Cambridge.

Ballard, D.H. & Brown, C.M. (1982), *Computer vision*, Prentice Hall, ISBN:0131653164 New Jersey.

Bezdek, J.C. & Pal, S.K. (1992), *Fuzzy Models for pattern recognition*, IEEE Press, ISBN: 0780304225, New York.

Bumble-Bee (1999), *Parser Generator Manual* [online], Bumble-Bee Software, Available from http://www.bumblebeesoftware.com.

Costa, L.F. & Cesar Jr., R.M. (2001), *Shape Analysis and Classification*, Image Processing Series, Philip A. Laplante, (Ed.), CRC Press, ISBN 0-8493-3493-4, Florida.

Ferreira, M. J., Santos C, (2006), Tracking system using texture cue based on wavelet transform. *7th Portuguese Conference on Automatic Control*, Controlo 2006, September 2006, Lisboa, Portugal.

Fu, K.S. & Booth, T.L. (1986a), Grammatical inference: introduction and survey, Part I, *IEEE Transactions on pattern analysis and machine intelligence*, Vol. PAMI -8, No. 3, page numbers (343-359), ISSN: 0162-8828.

Fu, K.S. & Booth, T.L. (1986b), Grammatical inference: introduction and survey, Part II, *IEEE Transactions on pattern analysis and machine intelligence*, Vol. PAMI 8, No. 3, page numbers (360-375), ISSN: 0162-8828.

Gonzalez, R.C. & Woods, R.E. (2002), *Digital Image Processing*, Prentice Hall, ISBN: 0201180758, New Jersey.

Ivancic, F. & Malaviya, A. (1998), An Automatic Rule Base Generation Method for Fuzzy Pattern Recognition with Multi-phased Clustering, *Proceedings of KES'98 IEEE Conference of Knowledge Engineering System*, page numbers (66-75), Australia, April 1998, IEEE Press, Adelaide.

Janome (2004), *JR 2000 Software manual*, Series Desktop Robot, I&J Fisnar Inc.

Kerre, E.E. & Nachtegael, M. (2000), *Fuzzy techniques in image processing*, Studies in fuzziness and soft computing, Vol. 52, Physica-Verlag, ISBN: 3790813044, New York.

Kindratenko, V. (2004), Shape analysis, In: *CVonline: Compendium of Computer Vision* [online], R.Fisher, (Ed), Available from http://homepages.inf.ed.ac.uk/rbf/CVonline/.

Lee, E.T. & Zadeh, L.A. (1969), Note on fuzzy languages, *Information Sciences*, Vol. 1, (October 1969), page numbers (421-434).

Looney, C.G. (2002), Pattern Recognition, In: *Opto-Mechatronic Systems Handbook: Technical Applications - Handbook Series for Mechanical Enginnering*, Hyngsuck Cho, (Ed.), Vol 10, CRC Press, ISBN: 0849311624, New York.

Malaviya, A. (1996), *On-line handwriting recognition with a fuzzy feature description language*, PHD Thesis, Technische Universitat Berlin, Berlin.

Matrox (1998), *Matrox Imaging Library User guide*, Manual No. 10513-MN-0500, Matrox Electronic System Ltd, Canada.

Micheli-Tzanakou, E (2000), *Supervised and Unsupervised Pattern recognition, feature extraction and computational*, Industrial Electronics Series, J. David Irwin, (Ed.), CRC Press, ISBN: 0-8493-2278-2, Florida.

Pal, S.K. & Majumber, D.K. (1986), *Fuzzy mathematical approach to pattern recognition*, Halsted Press Book, John Wiley & Sons, ISBN: 0470274638, New Delhi.

Peters, L. & Leja, C., Malaviya, A. (1998), A Fuzzy Statistical Rule Generation Method for Handwriting Recognition, *Expert Systems Journal*, Elsevier Press, Vol. 15, No. 1, page numbers (1591-1604).

Russ, J.C. (1998), The image processing handbook, 3ed, CRC Press, ISBN: 0849325323 London.

Stanchev, P.L. & Green Jr, D. (2002), Formal languages for image primitives description, *International journal information theories and applications*, Vol. 9, No. 2, page numbers (51-60).

Williams, P.S. (1999), *The automatic hierarchical decomposition of images into sub-images for use in image recognition and classification*, PHD Thesis, University of Western Australia.

Yager, R.R. & Zadeh, L.A. (ed) (1992), *An introduction to fuzzy logic applications in intelligent systems*, Kluwer Academic, ISBN: 0792391918, Boston.

23

Virtual and Mixed Reality in Telerobotics: A Survey

Costas S. Tzafestas
National Technical University of Athens, School of Electrical and Computer Engineering, Division of Signals Control and Robotics, Zographou Campus 15773, Athens, Greece (Email: ktzaf@softlab.ntua.gr)

1. Introduction

Virtual Reality (VR) constitutes now a well-established term, which is employed basically to describe systems that enable an intuitive and natural interaction in real-time between a human and a computer animated 3D graphical environment. The media that are employed to support such a human-computer interaction should ideally involve all human senses (and more generally sensori-motor abilities), that is, not only vision (through the rendering of 3D graphics models) but also audition, as well as the sense of touch, through a haptic interaction with virtual objects (Burdea & Coiffet, 94). The challenge is to develop computer simulation environments (simulating an existent or fictitious world) in such a way as to create to the human user an illusion of reality. VR thus constitutes a multidisciplinary field covering a variety of scientific and technological subjects, from 3D computer graphics and multimedia technologies, to the design and control of mechatronic human-computer interaction devices, human factors and modeling of sensori-motor human skills, as well as topics related to the field of human perception and psychophysics.
VR and its applications have flourished significantly particularly during the last decade, not only through the development of new software and hardware tools, devices and methodologies of human-computer interaction, but also by the implementation of these techniques into a constantly increasing number of new application paradigms and human-centered activities. Augmented and mixed realities constitute in fact one form of applied VR techniques and refer to the processes of overlapping synthetic (virtual) model images onto real world images, that is, to the combination (mixing) of virtual and real world. In other words, in an augmented reality (AR) system virtual models complement or modify in a way reality (providing in a sense some additional information as needed) instead of completely substituting (replacing) reality, as is the objective in VR systems. The goal here is to provide the user with the illusion of 'co-existence' of virtual and real world at the same time and space domain.
Augmented Reality, as a research and technology field, has attracted considerable interest during the last years with new domains of practical applications arising. Much effort is focused on the new potential opened by the application of AR techniques, and concern enhancing the ability of the human user (human operator) to perceive aspects of a real world by overlaying additional (visual or other type of) information, increasing the abilities

of interacting within this world. The auxiliary information that can be conveyed through the visual, haptic, or other form of display of virtual models may facilitate the user to perceive in a more direct and intuitive way specific "hidden" or else "fuzzy" characteristics of a real world, which may be needed for the efficient execution of a certain task. Therefore, while in general VR promises to revolutionize the way human-computer interaction systems are conceived, AR techniques seem to lead to the creation of excellent tools making execution of complex and demanding tasks more easy and intuitive for the human.
Virtual, Augmented and Mixed Reality technologies are now recognized as constituting a challenging scientific and technological field that can provide breakthrough solutions to a wide spectrum of application domains, where intuitive and natural human/computer and human/machine interaction is needed. Telerobotics, involving a human operator to control a robot from a remote location usually via a computer interface and a computer network, is one of the fields that can directly benefit from the potential offered by VR and AR human/machine interfacing technologies. Application of VR in Telerobotics (VRT) and the related concept of telepresence, or tele-symbiosis (Vertut & Coiffet, 1984), are ideas which have been around for more than twenty years, and have been used mainly for the telemanipulation/teleoperation of robotic mechanisms in hostile environments (such as in the nuclear industry, space, underwater, or for other difficult or dangerous service/intervention tasks, like bomb disposal, civil works etc.). In fact, all the approaches involving the integration of VR techniques in telerobotics, as will be explained further in this chapter, constitute basically: (a) a generalization of the concept of "predictive displays", coping with the problem of time delay and stability in teleoperation systems, and (b) an attempt to provide human operator assistance and achieve better transparency characteristics for the teleoperation system.
Nowadays, on the other hand, the rapid development of new broadly expanded networking technologies, such as those related to the Internet, and the numerous relevant applications, described by the general term of e-commerce/e-business, can give new potential to the use of VRT in novel application domains. In fact VRT and Internet technologies can mutually benefit from ideas developed in the respective fields. This merging of technological potential can lead to a generalization of the concept of telework, where remote control through the network of actual physical processes will be possible. One can even think, for instance, of supervising and actively controlling a whole manufacturing process without having to move from his home. A major research objective must be of course to enable and promote new potential applications that can derive from the merging of such technologies, so that wider categories of the population can finally take benefit of these technological advances.
At the rest of this chapter, we focus on analysing the theoretical foundations of this field and on describing the practical application domains that are involved, by presenting some characteristic case studies. Section 2 starts with a description of the basic principles related to virtual and augmented reality systems, and then presents an overview of applications related to the field of robotics. In Section 3 we describe the basic concepts that govern telerobotic systems and present a historical survey of the field. Section 4, then, presents typical application scenarios of these technologies, related to the two main robotic systems categories, namely robot manipulators and mobile robotic vehicles, and highlights the link with the new VR-based field of haptics. Concluding remarks and future research directions are given in Section 5.

2. Virtual and Mixed Reality: General Description

During the last ten to fifteen years, Virtual Reality (VR), as a theoretical and applied research field, has attracted the interest of the international scientific community, as well as of the public opinion through extensive use of the term by the information, communication and entertainment media. However, the latter often results in an "abusive" use of this term, which is probably due to the lack of a formal definition of the field. In the sequel, we attempt to describe the basic principles that govern the field of VR, as well as of the more recent domain of Augmented and Mixed Reality systems, and we give an overview of related applications.

2.1 Definitions and Basic Principles

In an attempt to define what is a VR system, in relation to what can be seen as a simple human-computer or human-machine interaction, we can say that VR refers to: (a) computer generated and animated, *three-dimensional realistic visualization space*, enabling (b) *real-time and multimodal interaction* involving multiple sensori-motor channels of the human user, aiming to achieve (c) a sense of *immersion* and (virtual) *presence* in this synthetic (simulated) environment. The common factor here is, thus, the stimulation of human perceptual experience to produce an impression of something that does not really occur, but which is perceived and believed (potentially invoking, at some extent, human *imagination*) as being physically present and existing as a real world. The three important dimensions characterizing VR systems, and differentiating them from typical computer simulation environments, are: *interaction*, *immersion* and *imagination*, all contributing to create a sense of virtual presence and realism (Burdea & Coiffet, 94).

A Virtual Environment (VE) created via graphics is a communication medium having both physical and abstract components. The three basic constituents of a VE are the *content*, the *geometry* and the *dynamics* (Ellis, 1995). The content consists of objects and actors. The geometry is a description of the environmental field of action, and has dimensionality, metrics (rules establishing an ordering of the contents) and extent (range of possible values for the elements of the position vector). Dynamics is represented by the rules of interaction among the VE contents, describing their performance as they exchange information or energy. The components of a VE are useful for enhancing the interaction of the operators with their simulations. *Virtualisation* is defined to be the process by which an observer (viewer) interprets patterned sensory impressions to represent objects in environment other than that from which the impressions physically originate. Virtualisation can be applied to all senses: vision, audition, contact, shape and position (haptic sense).

The three complementary technologies used to create the illusion of immersion in a VE are:

- Sensors (e.g. head position tracker or hand shape sensors)
- Effectors (e.g. stereoscopic displays or headphones)
- Special purpose hardware and software (connecting the sensors and effectors in such a way as to create experiences encountered by people immersed in a physical environment)

A general diagram showing the structure of a VR-based system and the linkages of its components is shown in Fig. 1. The human operator can interact with a VE presented by means of head and body referenced displays, the success depending on the fidelity with which sensory information is presented to the user. The environment experienced by the user via a VE simulation is of course imaginary. On the contrary, when referring to a teleoperation interface, the human operator is provided with a perception of an environment that is real (e.g. image views of the remote physical-task environment). One can then immediately consider the use of VR environments as intermediate representations "interfacing" the human operator with the remote task environment, with the objective being to assist him in several ways to perform more efficiently the desired physical task. In such teleoperation interfaces, real and simulated data can be combined via digital processing to produce intermediate environments of real and simulated (synthetic) objects. The mixture of real and virtual entities within the same environment refers to augmented and mixed reality interfaces, a new field that has evolved as a special category of VR systems, as described in the following section.

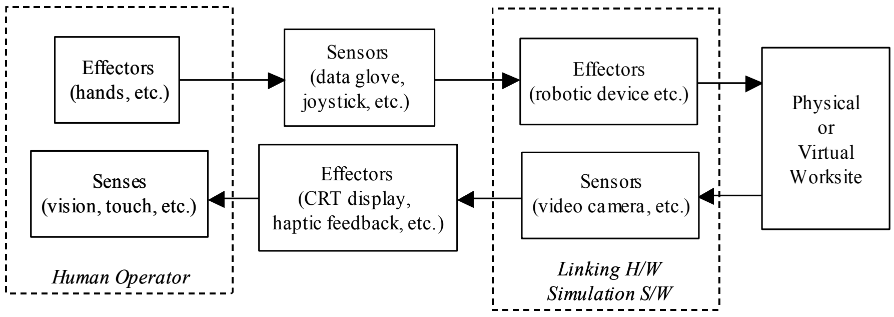


Fig. 1. General structure of VR-based human/machine systems.

2.2 Augmented Reality: Basics

Augmented Reality (AR) systems constitute in fact a category of quite specialized VR technologies, which have attracted significant interest during the last years due to the numerous applications they find in various new domains. While VR has as the main goal to immerse the human operator in a completely synthetic 3D simulation environment, the basic principle of AR systems is to enable the user to experience simultaneously parts of the actual (real) physical world. In other words, AR complements, instead of entirely substituting, the real world, with the ideal situation being to create the illusion that both virtual and real objects "coexist" in a unified (mixed) environment (Azuma, 97).

The question is then: why has this new field of AR systems found such an interest within the scientific community during the last years, and where does the usefulness of such mixed reality environments reside? The answer is that virtual environments can realistically display various data encoding complex information related to the real world, information that is not directly accessible in reality and could not be perceived differently from the human being. One can say that display of this information, which is conveyed by means of virtual objects, in fact "amplifies" the perceptual capacity of the human being, increasing the abilities to perform complex tasks on the real world.

The application domains of AR systems are various and are constantly evolving during the last years. They comprise: (a) *medical applications*, like simulation for education and training in medical (invasive or not) procedures, as well as pre-operative planning and computer-aided (image-guided) operations, (b) *CAD and manufacturing processes*, for instance architectural design of a new building and "previewing" its spatial integration, training in maintenance procedures etc., (c) applications in the *entertainment industry*, e.g. with the production of special effects, virtual actors etc. Military applications are, unfortunately, also not excluded. In the following section, we present a short overview of VR and AR applications in robotics, as introduction to the use of such technologies in the field of telerobotics, which forms the main scope of this chapter.

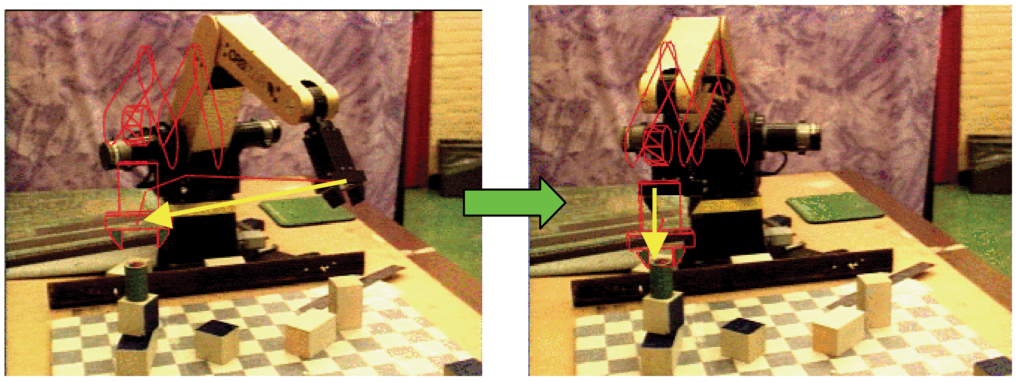

(a) (b)

Fig. 2. Programming of a robot manipulator using augmented reality (AR) techniques. Overlaying virtual model (3D wireframe graphics above) on real robot image (adapted from: Rastogi et al., 96).

2.3 Overview of VR Applications in Robotics

A very important domain where VR and AR technologies find many and interesting applications concerns the field of robotics and robot integrated manufacturing systems. It is now asserted that the use of such technologies can provide significant benefits in all the phases of a manufacturing procedure, from initial design to implementation and control, particularly when these involve the integration of robotic systems. As outlined previously, given that a VE constitutes in fact an *integrated human-machine interaction* system, VR can contribute significantly in all the processes where human intervention (and human factor in general) plays an important role, like for instance:

- The design of *virtual prototypes* and the evaluation / assessment of various characteristics (including aesthetic, ergonomic etc.) and parameters related to functionality, feasibility tests, reliability, consistency of operation etc., which can be performed by means of interactive VR simulation environments.
- The *programming and control* of automated (robotic) procedures, with the goal being to better exploit the skills of the human operator, as well as his capacities to evaluate complex situations and solve decision problems, such as task and path planning involving a robot manipulator.

One very useful related application of AR concerns programming of robot manipulation tasks. The basic idea resides on the use of virtual models representing the robot and its task environment as an intermediate representation to guide and assist the robot action planning process. Overlaying 3D graphical models, representing the robot and the planned motion (action) sequence, on real views of the actual task environment, enables the human operator to perform a "previewing" of the system operation, potentially off-line, facilitating the programming and validation of complex robotic tasks, without the need to constantly work with the real robot in an on-line programming scheme (with all the advantages that such an off-line robot programming presents in practical scenarios). Of course, VR models must be correctly superposed on the real world images, for such systems to be of any practical use. This is called *3D image registration*, which constitutes one of the basic problems that needs to be tackled in any AR system, based on camera calibration techniques, and probably making use of (image- or sensor-based) *3D tracking* methodologies (if such a system is to operate in real-time).

Fig. 2 presents an application example of such an AR-based programming of a robot manipulation task (Rastogi et al., 96). The graphical model (wireframe) shown in this figure constitutes in fact a virtual representation of the real robot, which is overlaid on real robot image views assisting the human operator to better evaluate the anticipated outcome of a programmed action sequence. In this case, the task consists of grasping an object and performing a "pick-and-place" operation. Fig. 2(a) shows the "predicted" path of the robot manipulator, as this is programmed using this AR interface. Fig. 2(b) displays a subsequent view with the robot manipulator having moved according to the planned operation, with both real and virtual robot images being registered (i.e. correctly aligned), demonstrating the accuracy of this robot planning scheme. It must be pointed out here that the additional use of stereoscopic images with 3D graphical models can significantly enhance the efficiency of the human operator in performing such robot programming tasks, by providing visual feedback information in a more intuitive way and increasing the overall performance of the system (reducing time, minimizing false maneuvers etc.).

Based on the above concepts, virtual and augmented reality techniques have evolved substantially during the last decade finding significant and interesting new applications particularly in a special field of robotics called *telerobotics*, which can be mainly characterized by the direct involvement of a human operator in the control loop. The theoretical and practical foundations of this domain form the main scope of this chapter and are thoroughly analysed in the following section.

3. Telerobotics: Historical Evolution

Telemanipulation as a scientific term describes all the methodologies and techniques enabling a human operator to perform from a distance a manipulative task, using his own hand through the use of an intermediate mechatronic system. Telemanipulation control of a remote manipulative task, besides its fascinating character related to the notion of extending human capabilities by some tool beyond usual space or time limits, it can prove extremely beneficial in cases where human intervention is indispensable to perform a task taking place in an unstructured "hostile" environment, due to the increased uncertainty and non-repetitiveness characteristics of such tasks, and the complex task/path planning required for timely and correct execution. Original master-slave telemanipulation systems consisted of a couple of mechanical or electromechanical arms (one called the master, controlled by the human operator, and the other, called the slave, performing the remote manipulation task). Bilateral exchange of energy (position and force signals) was initially ensured through a mechanical linkage and, later-on, through the use of electrical links and servo-control loops. In its infancy, telemanipulation technology found outstanding applications in the nuclear industry for the remote manipulation of radioactive materials in environments where human presence was hazardous. Typical example is the work accomplished by Raymond Goertz at Argonne National Laboratories, USA, or by Jean Vertut and the French group at the CEA (Vertut & Coiffet, 84).

Bilateral servo-controlled telemanipulation and industrial computer-controlled robotics were two technological fields developed originally in parallel and, in some extent, independently. The awareness that both these fields can benefit from development accomplished in each other has led to the fusion of these technologies and the creation of what is generally described under the term of telerobotics. Robotics was initially concerned with the development of industrial manufacturing systems performing programmable, repetitive operations in an autonomous sensor-based manner, while telemanipulation was focusing on a different class of tasks, which should clearly rely on the predominant presence of a human operator in the control loop. Telerobotics, which globally describes the fusion of these general technological fields, is a very challenging and promising research field, which aims at exploiting in a full extent both human operator skills and machine intelligence capabilities within a human/robot interaction and cooperation context.

The integration of some mobility characteristics on a remote manipulation system, has extended the workspace and, generally, the functionality of these systems in terms of space and task limitations, and has led to the creation of new application domains covered under the more broad term of teleoperation. Such application domains include the development of mobile telemanipulator vehicles for space operations (e.g. Mars Rover etc.), with typical examples being the mobile robotic systems developed by NASA, for future Mars exploration missions[1]. Underwater remotely operated vehicles (ROVs) have also been developed, such as those described in (Gracanin & Valavanis, 1999). All these systems belong to the general field of intervention and service robotics, which focuses on the development of integrated mobile robot platforms with embedded manipulation and sensing modules, operating under direct remote control or semi-autonomously under high-level human supervision. Such systems aim mainly at substituting the human being in the execution of hazardous (e.g. handling of explosives), painful (e.g. lifting heavy weights, for instance civil works), or else boring every-day tasks (e.g. vacuum cleaning etc.). In section 4.2, we will present one example of such a mobile service robot. This general field also comprises systems that aim at assisting humans when performing delicate operations, requiring increased precision, which is the case of the research performed in the field of medical robotics, dexterous telemanipulation and telesurgery.

Let's describe now the main problems encountered in general teleoperation systems, as well as some existing solutions, methodological approaches and guidelines proposed in the literature, in order to situate the current state-of-the-art of research carried out in the field of telerobotics. The major problem and certainly the most cited one is the presence of time delays in the bilateral communication loop, which is mainly due to the distance separating the master from the slave site, but may also be due to the processing time required for coding and data transmission. Such delays may be constant (e.g. in the case of direct ISDN link), but may also be varying in an unpredictable manner due to the load of the network servers (which is the case of the Internet), causing additional difficulties in coping with the problem. For instance, time delay for transcontinental teleoperation when a satellite link is used may exceed 1 second, while when teleoperating a rover on the moon, round-trip time delay approaches 3 seconds. The human operator is in such cases obliged to apply a ``move-and-wait'' strategy, that is, to make small moves while waiting for the images (and in general, the sensory feedback) to be updated. As a

[1] (see for instance: http://robotics.jpl.nasa.gov/groups/rv/ for a brief survey)

consequence, communication time delays cause certain degradation of the teleoperation system's performance; but what is even more critical, their presence may jeopardize safe operation and cause dangerous instabilities especially when force-feedback is involved in a long-distance bilateral telemanipulation system.
Degradation of sensory feedback may also be due not only to the presence of time delays and limited bandwidth, but also to noise and other sort of disturbances in the communication channel. Problems related to the quality of sensory feedback may also derive from the nature of the task itself, for instance when a slave robot operates in low visibility conditions (e.g. video feedback from an underwater remotely operated vehicle, which may, in some cases, be completely useless or extremely difficult to interpret). In all these cases, when sensory feedback is deteriorated, due to time-delays, noise or other source of signal degradation, some task-specific methodology or advanced remote control strategy has to be followed to assist the human operator to perform the task goals, and ensure safe and efficient operation of the system.
Time-delay has long been known in classical control theory as a very challenging problem, and various predictive control schemes have been proposed based on some a-priori knowledge of the delay (for instance, the predictor of Smith, proposed around 1956, see: (Laminat, 1993) for a survey). In the teleoperation field, more recently, some new control schemes have been proposed to cope with this problem, based on passivity theory (Anderson & Spong, 1992), or on the concept of adaptive impedance (Niemeyer & Slotine, 1991). All these approaches converge to the fact that, in any case, stability and transparency (defined in terms of force/trajectory tracking between the master and slave) of the teleoperation system are two contradictory objectives, and some trade-off between these characteristics has to be achieved most of the times. All these approaches in fact slow down the control system coupling the master with the slave, that is, diminish the control bandwidth of the system leading to a more compliant (less stiff) teleoperator. This ensures the stability (passivity) of the system, under some constraints related to the magnitude of the time delay, but has as a counter-effect to deteriorate the transparency of the teleoperation system (for instance, the human operator does not feel the real profile of the force generated at the slave site). The problem becomes even more difficult when time-delay is randomly varying, with no a-priori knowledge available on its order of magnitude.
Another class of techniques trying to cope with the problem of communication time-delay, is based on the use of *predictive displays*. Graphical predictors, supplying visual cues (estimations) on the evolution of the teleoperation task, are the most commonly used. Bejczy et al. (1990), for instance, have proposed the use of a wireframe graphical model of the slave robot, overlaid on the usual video feedback provided to the human operator. This combination of both synthetic and real images (that is the display of a graphical model, directly following the movements of the human operator and showing what the state of the robot will be before the actual delayed video images arrive from the slave site) greatly facilitates the task of the human operator. The paradigm of graphical predictive displays has been greatly adopted since, and extended to cope not only with problems related to the presence of time delays in the bilateral communication loop but also to perform visual feedback enhancement and assist the human operator in quickly assessing a situation and performing teleoperation tasks.

3.1 Teleoperation and Virtual Reality: Synergy

The integration of more advanced virtual reality techniques in teleoperation systems can be partly seen as a generalization of the concept of predictive displays described above, where the term display may now refer not only to the visual display of simple graphical cues, but also to other forms of sensory feedback such as haptic or auditive display. Virtual Reality is in fact a multidisciplinary scientific/technological field, which aims to enable a more natural and intuitive human/computer interaction based on the use of multimodal/multisensory interfaces. This human/machine interface technology involving various perceptuo-motor modalities of the human being (not only vision, but also haptic interaction and auditive feedback) can provide a technological solution of excellence for the human/robot interaction and communication systems constituting the field of telerobotics. Virtual environment simulations of teleoperation systems can indeed be used as predictive models performing the role of a mediator between the human operator and the remote (slave) robotic system. This means, in other words, that the human operator could be provided with realistic three-dimensional graphical images of the remote operation site, while being able to interact with these images and perform the desired teleoperation task in a natural and intuitive way (that is, for instance, by feeling the reaction forces during the execution of this virtual task model), and all that before the actual (delayed or deteriorated) real sensory-feedback signals arrive from the remote slave site. In fact, this interaction between the human operator and the virtual environment (that is, the virtual task performed by the human operator) can be used to generate the appropriate command signals that have to be sent to the slave robotic site, and guide the on-line execution of the real teleoperation task. The use of such an intermediate virtual representation of a teleoperation task is reported in (Kheddar et al., 1997), where a multi-robot long-distance teleoperation experiment is described, as will be presented more in detail in Section 4.1.1.

VR-based models of teleoperation tasks can also be used in off-line *teleprogramming* schemes, in which case the master and slave control loops are completely decoupled. The human operator performs a virtual task in a completely simulated manner, within a 3D graphic environment representing the slave site. This virtual task is analyzed and the appropriate sequence of robot commands is extracted and recorded. The sequence of command signals is then evaluated by the human operator before its subsequent transmission to the slave robotic system, where real task execution will take place. Communication time delay is generally not a problem in this approach. However, this is not applicable for all kind of teleoperation tasks, for instance when fine telemanipulation of a dextrous robotic mechanism is required, since programming such complex tasks in the form of simple sensor-based operations is very difficult. The key issue in teleprogramming schemes is the type of commands that will constitute the robot programs, which must make use in full extent of any autonomy features supported by the slave robotic system, in the form of reactive sensor-based behaviours or elementary task operations. Such approaches are especially applied in super-long-distance teleoperation systems, for instance when guiding the operation of a rover on the surface of a distant planet such as Mars. Of course, the same idea of semi-autonomous teleoperation control can also be applied in an on-line direct teleoperation scheme, where more high-level command primitives can be send in real-time to the remote robot, instead of the traditional, continuous force/position/speed signals. In this general framework, Hirzinger et al. (1993) have proposed the use of a tele-sensor-based scheme for the remote control of a robot manipulator in space. Freund and Rossmann (1999) have proposed a task deduction/action planning approach (called projective virtual reality paradigm) tested on a variety of applications, from simple teleoperated robotic assembly tasks up to the control of multirobot telemanipulation systems for space applications. In

Section 4.1.2, we will present an example of a similar telerobotic system, but with the application being that of a "remote laboratory" for education and training in robotics.
VR technology and its applications in different scientific fields have known a rapid development during the last five to ten years. We can now say with confidence that VR has the potential to become a key technology for the design of modern man-machine interfaces, as is the case of teleoperation systems. It can provide the tools and techniques to establish a multimodal, natural and intuitive human-machine interaction, increasing the feel of *telepresence* for the human operator, which constitutes the ultimate goal of any teleoperation/telerobotic system. Of course, many challenging problems have to be tackled and appropriate (generalized or task-specific) solutions must be proposed, taking into consideration not only ergonomic issues and human factors, but also more technical problems such as image calibration (Kim, 96), coping with discrepancies and modeling uncertainties, as well as control issues and stability of human-machine active interfaces.
The use of VR techniques in telerobotics can be seen as an evolution of general *computer-aided teleoperation* schemes, developed to facilitate the task of the human operator and provide assistance in one of the following ways:

- by performing the functions of an *information provider*, that is, by enhancing the sensory feedback provided to the human operator and helping him to better understand the state of the remote task execution. Typical examples are the graphical predictive displays, described above, or some form of artificial haptic (kinesthetic and/or tactile) feedback. Other VR-based techniques include the use of virtual fixtures (Rosenberg, 1993) or virtual mechanisms (Joly & Andriot, 1995).
- by performing some form of *decision support* function, that is, by providing suggestions or indications concerning the most suitable action plan and assist the human operator at the decision making process.
- by interpreting the actions of the human operator and performing a function of *substitution* or cooperation, to provide *active assistance* for the on-line control of a teleoperation task. This is the case of an active intervention of the master computer, with typical examples being a system undertaking the control of some degrees of freedom (dof), or ensuring that the commands issued by the human operator satisfy some constraints related to safety issues.

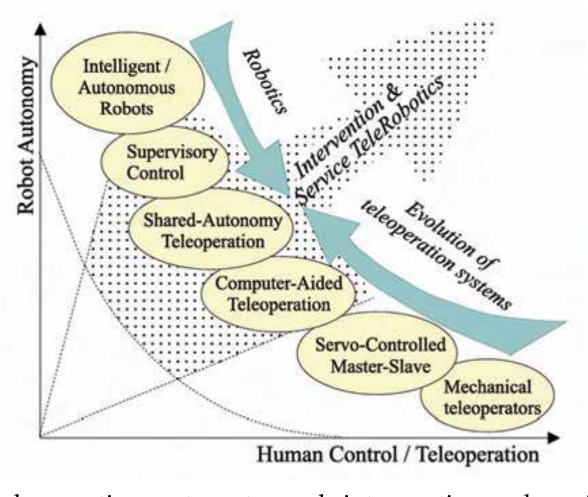

Robot Autonomy
Intelligent / Autonomous Robots
Supervisory Control
Shared-Autonomy Teleoperation
Computer-Aided Teleoperation
Servo-Controlled Master-Slave
Mechanical teleoperators
Robotics
Intervention & Service TeleRobotics
Evolution of teleoperation systems
Human Control / Teleoperation

Fig. 3. Evolution of teleoperation systems towards intervention and service telerobotics.

All these features (i.e. providing perception, decision or action assistance to the human operator) concern functions performed by the system within the master control station and are generally described by the term *computer-aided teleoperation*. Similarly, some form of computational intelligence can be embedded to the slave control system, which is for instance the case of a slave robot supporting some kind of autonomous sensor-based behaviors. In this case, we refer to a *shared-control* (or *shared-autonomy* control) mode of operation, with the slave robot executing a set of elementary (or more complex) operations in a completely autonomous mode. The commands issued by the master control station (that is, by the human operator) are described in a higher level of abstraction and include some form of implicit task representations. In an even higher level one could then think of a telerobotic system where the human operator is in charge of simply supervising the remote task execution, with active intervention only in extreme error recovery situations. All these paradigms are generally grouped under the term *supervisory teleoperation*, described in (Sheridan, 1992). A schematic representation of the evolution of these teleoperation paradigms is illustrated in Fig. 3. The interaction and merging of machine intelligence features with the human operator capacities and skills is the key issue that will lead to the creation of more advanced telerobotic systems, capable to perform more complex task such as those required in the field of intervention and service robotics. It is certainly one of the most challenging tasks for the designers of modern teleoperation systems, to find the "optimum line" between robot autonomy and human operator control, in order to exploit in a full extent the potential of such human/machine interaction and cooperation systems.

3.2 Web-Based Telerobots

Until quite recently, that is before the last five to ten years, telerobotic systems were remotely operated through dedicated fast network connections, and their use was exclusively reserved to trained specialists. The integration of teleoperation technology with new rapidly evolving media/network technologies, especially the Internet and the World Wide Web technologies, promises to open the door to a much wider audience, by creating and wide spreading new application domains. Controlling a real distant device over the Internet and performing a physical process in a remote location (as opposed to simple information processing) will extend the scope of telework applications, most probably having a significant impact in many aspects of both social and economic life. This section presents a brief survey of such web-based telerobotic systems. Situating the current state-of-the-art for this promising and challenging research area, is of particularly interest within the scope of the survey presented in this chapter.

By web robots we mean robotic devices that are accessible from any computer connected on the Internet. Remote control of these systems via the Internet is possible by any site using a standard web browser incorporating the human operator control interface. Even though there exist by now many robots available for teleoperation on the web, the development of such systems is still more or less in its infancy and consists mainly of "playing" with a distant robot over the Internet, issuing simple motion commands to perform elementary tasks. A typical example is the Australia's telerobot, developed at the University of Western Australia[2]. It

[2] http://telerobot.mech.uwa.edu.au/

consists of a six-axis robot manipulator, remotely controlled with one fixed observing camera. The initial system, originally demonstrated in 1994, required users to type in spatial coordinates to specify relative arm movements. Since then, various user interfaces have been developed and tested (Taylor & Dalton, 2000), which more recently embed Java technology to enable the human operator either to choose from a prespecified set of target positions or to click on the image and issue robot motion commands relative to the position of a cursor. The problem of course still remains to associate the position of the cursor that is being dragged on a 2D image, with the position of the robot end-effector and the other objects in the 3D world. An other very good example of a robotic manipulator being controlled over the Web is the PumaPaint system (Stein, 2000), which was on-line from June 1998 until March 2000. It consisted of a Puma 760 robot controlled over the Internet using a Java compatible web browser. The task performed by the robot was painting on an easel, reproducing in real the paintings created by the user on a virtual canvas, which was incorporated in the user interface running a Java applet. The interface forwards all commands to the robot so that almost the same image appears on the real canvas. The system also provides visual feedback in the form of periodically updated live images from the robot.
Besides these systems consisting of robot manipulators controlled through the Internet, there is another class of web robots involving teleoperation of mobile platforms over the www. Most of these systems provide exclusive remote control to a single person or provide queues to schedule user requests. One of the first mobile robots to operate in a populated office building, controlled through the web, was Xavier (Simmons, et al., 2000). This system was created by the end of 1995 to test the performance of various navigation algorithms, but has soon become very popular with more than 40,000 requests and 240 Kilometers travelled to date! The command interface of the robot provides a discrete list of destinations to send the robot and a list of simple tasks to perform there. When a user submits a task request, this task is scheduled for execution and a confirmation web page is sent back indicating when the robot will most likely carry out this task. If the user had registered using a correct e-mail address, the system will send an e-mail after completion of the requested task. In addition to the command interface page, there is a monitoring web page that includes the robot's current status, a map of the floor the robot is currently on and a picture of what it currently sees.
A very interesting application of such web-based systems involves remote control of mobile platforms moving in a museum. These are called tour-guide robots (Thrun et al., 1999), like the Rhino robot deployed in the Deutches Museum in Bonn, or its successor, Minerva (Schulz et al., 2000), installed successfully in the Smithsonian's National Museum of American History. These robots are operated either under exclusive control by remote users on the web (virtual visitors), or under shared control by both real (on-site) and remote (virtual) visitors of the museum. Under exclusive web control, the user interface is implemented as one Java applet incorporating a map of the exhibition area and two live images, one from the robot and the other from a ceiling-mounted camera. Minerva's shared control interface was on-line for 91 hours and was accessed by 2885 people. The robot travelled 38.5 Km under shared web and on-site control, providing information about 2390 exhibits.
There exist many other Web robots on the net, performing a variety of tasks such as those described in (Goldberg, 2000). The NASA Space Telerobotics program website[3] currently

[3] http://ranier.oact.hq.nasa.gov/telerobotics_page/realrobots.html

lists over 20 Real Robots on the Web. Reviewing all those web-based teleoperation systems, it is clear that the main problem is of course the unpredictable and variable time delay for communication over the Internet, which calls for the use of some form of supervisory control or off-line teleprogramming scheme to ensure stability. Most of the systems currently available on the web incorporate user interfaces, which implement basic functionalities, such as enabling the user to choose from a prespecified set of tasks (e.g. target locations). These interfaces use some combination of HTML forms or Java consoles to enter data and issue simple commands for immediate or future execution. Sensory feedback is usually limited to the display of images that are captured at the remote site, and the presentation of some status information in text form. It is obvious that this separation between the actions of the human operator (user) and the response of system fed back by the remote robot deteriorates the transparency and telepresence characteristics of the teleoperation system. More advanced "interactive telepresence" techniques need to be investigated, like for instance the integration of VR models and tools within the master control interface (including predictive displays and automatic active-assistance operations) to enable a more natural, intuitive and direct, real-time interaction between the user and the web-based teleoperation system.

4. VR in Telerobotics: Application Scenarios

As already stated, VR can be used in various ways to enhance robot teleoperation systems. Since it can be seen as constituting, in fact, a pool of advanced multimodal human/machine interaction technologies, VR can be employed at a "mediator" level between the human-operator and the remotely controlled robotic system. The performance of any telerobotic system can be measured in terms of two, often contradictory, indicators:

(a) *Transparency*, that is, the fidelity with which the human operator can perceive the remote robot environment, and the easiness by which he can perform the remote task via the telerobot, and

(b) *Stability*, particularly in the presence of large time delays in the bilateral communication and control loop that can jeopardize smoothness of operation, especially when force-reflecting bilateral telemanipulation is involved.

The goal of using VR interfaces as mediators in human-robot interactive communication systems would thus be twofold:

- to increase naturalness and intuitiveness of human operation, by: (i) enhancing information visualization via virtual and augmented reality displays, (ii) exploiting the use of multimodal sensori-motor interfaces taking into account human factors, and (iii) providing active assistance to the human operator; the goal of all these being, therefore, to improve transparency of the telerobotic system, facilitating the task from the human operator perspective.
- to cope with the presence of large time delays, (i) through the use of predictive displays by means of virtual and augmented reality models, and (ii) by applying some form of off-line teleprogramming scheme based on a virtual representation of the remote task environment; the goal being, here, to improve stability of operation and robustness for the telerobotic system.

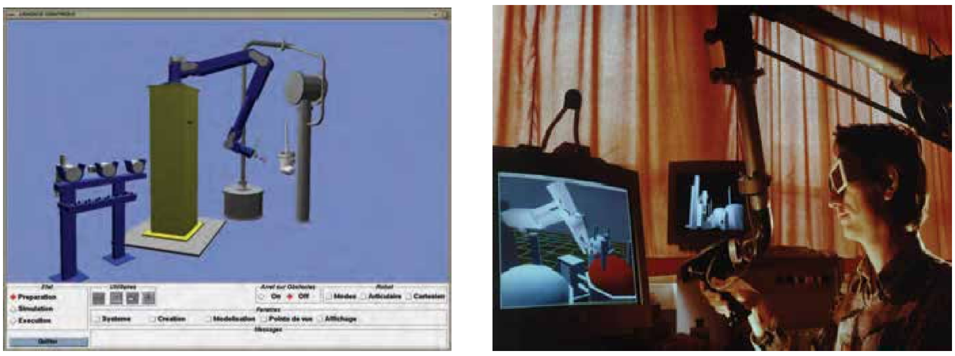


Fig. 4. TAO 2000 VR based teleorobotic interface (CEA, France).

In the rest of this section, we present some typical examples of VR-based teleoperation systems, for the two main classes of robotic systems, namely: (a) robot manipulators, and (b) mobile robotic platforms.

4.1 Robot Telemanipulators

Many experiments have been conducted, since over a decade now, in the context of robot telemanipulation, with main application fields being the nuclear industry (handling of radioactive material), and space telerobots (long-distance telemanipulation). The French Nuclear Centre (CEA – Commissariat à l'Energie Atomique) is very active in the field, since the very beginning of the teleoperation history. One of the most recent advances is the TAO 2000 system, a VR based graphical programming interface, for nuclear servicing using a master-slave robot telemanipulator system. The whole system is illustrated in Fig. 4, with the graphical tele-programming interface (left) and the master force-feedback manipulator arm (MA-23, on the right). G. Hirzinger and his team (1993) at DLR[4] have developed a multisensory telerobot and conducted the first actual space experiments on a space telerobot technology (ROTEX). The system has flown in a space-shuttle mission and worked successfully in various modes, including autonomous operation, teleoperation by astronauts, as well as telerobotic ground control using either on-line direct teleoperation or, what was termed, a telesensor programming mode.
Since then, numerous telemanipulation systems have been developed, applying various methodologies adapted from the field of virtual and augmented reality, with many different applications (from telesurgery to nano-scale telemanipulation). In the sequel, we present two typical examples of VR-based robot telemanipulation systems based on the application of VR concepts and technologies: (a) a long distance, parallel teleoperation of multirobot systems, and (b) a distance training (remote / virtual laboratory) system, for teaching robot manipulator programming using a multimodal VR-based web-enabled interface.

4.1.1 Long-Distance Multirobot Telemanipulation

On October 10th 1996, a teleoperation experiment was performed involving four robot manipulators of different kinematics and situated in different locations (respectively Poitiers, Grenoble and Nantes in France, and Tsukuba in Japan). The robots were teleoperated simultaneously (in parallel) by the master control station situated in Poitiers.

[4] German Aerospace Research Establishment, Wessling

This experiment was the first general one of a research cooperation programme named TWE (Telepresence World Experiment) linking seven research teams belonging to five countries (among them, the Laboratoire de Robotique de Paris in France, and the Mechanical Engineering Laboratory in Tsukuba, Japan).
The main challenge of this "*telework experiment*" was to demonstrate the possibility offered by VR technologies to ameliorate the human operator master control interface and enhance the capabilities of such robot teleoperation systems. With four different robots controlled in parallel to perform the same task, a common intermediate representation is imperative, to let the human operator focus on the task to be performed and 'mask' any robot manipulator kinematic dissimilarities and constraints. Fig. 5 shows an overview of the experimental setup, with the master control interfaces, and two robots in parallel operation (one in France and one in Japan). As can be seen in this figure, the task consisted of assembling a four-piece puzzle within a fence on a table. The operator performs the virtual puzzle assembly using his own hand and skill on the master control virtual environment, constituting an intermediate representation of the real remote assembly task. The visual and haptic feedback is local and concerns only the graphic representation of the remote task features without any remote robot. The operator / VE interaction parameters are sent to another workstation in order to derive robot actions (graphically represented for software validation and results visualization) and does not involve any direct operator action/perception. A video feedback was kept for safety and error recovery purposes.
The ultimate goal of such research efforts is to study the role that VR concepts and techniques can play towards the development of "efficient" interaction and communication interfaces between humans and robots. In the direction of ameliorating the transparency of the telerobot system, which constitutes a major target as has been already stated, such a human-robot interface must enable the human operator:

- to remotely perform the desired task in a natural and intuitive way, as if he was physically present at the remote (slave robot) site, without feeling obstructed or constrained by the telerobotic system, an objective that is often described by the term "*telepresence*",
- to use his manual dexterity in remotely conducting the desired manipulation task, meaning that the system must support *natural skill transfer* between the human operator and the remotely controlled (slave) robot.

To approach these key objectives, it is particularly important for the master teleoperation environment to display information to the human operator not in a static way but through a multimodal / multisensory dynamic interaction environment. VR concepts and tools play a significant role in this direction. Particularly, interacting via the (active) sense of touch is of primary important. This is termed "haptic interaction", or *haptics*, which is now a very active research field worldwide, with numerous applications, as we will see later on in this chapter.
These haptic systems are often based on the development of special purpose glove-like devices, often termed "data-gloves", or "force-feedback gloves" if application of forces on the human-operator's hand is possible. One such example is illustrated in Fig. 6, where the human-operator wearing a specially designed exoskeleton device on his hand (the LRP dexterous hand master, see for instance (Tzafestas et al., 1997)) can interact, in a direct and intuitive way, with a virtual environment representing the task to be performed. The virtual manipulation actions performed within this VR simulation environment on the master control site are transformed into an appropriate sequence of commands for a robot manipulator to execute, and are then transmitted to the slave robot site(s).

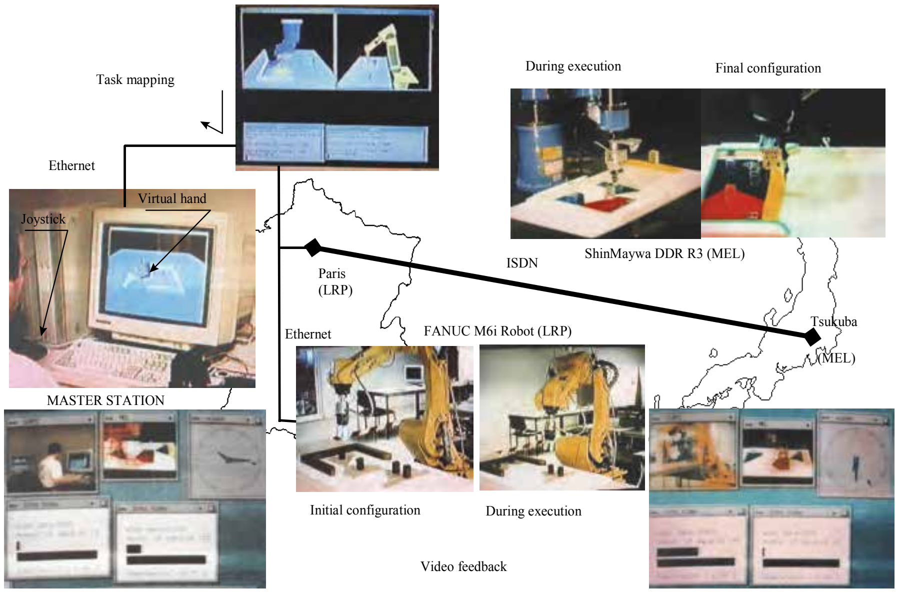


Fig. 5. Multi-robot long-distance teleoperation experiment (adapted from (Kheddar et al., 97)).

This general concept according to which, in an ideal telerobotic system, the human operator must feel as if he directly performs the task, instead of controlling the robot to perform the task, was called the *"hidden robot" concept*, and was also applied in the context of the multi-robot teleoperation experiment depicted above in Fig. 5.

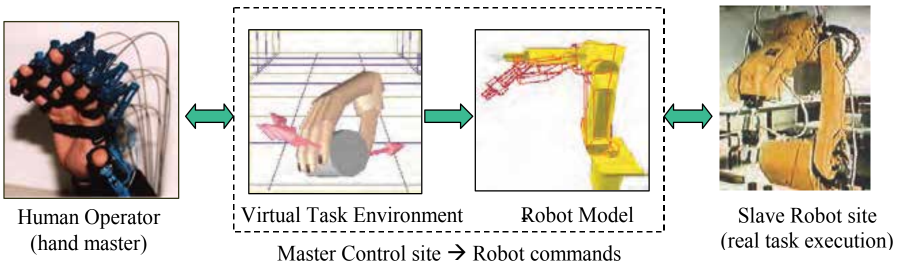


Fig. 6. Robot teleoperation by means of a virtual tele-work environment (adapted from (Kheddar et al., 97)).

The idea here is to let the human operator concentrate his awareness only on the task at hand and not on both the robot control and the task. The human operator is, thus, not concerned with the constraints imposed by the robot mechanisms (e.g. kinematical dissimilarities, etc.). Such issues are resolved by the system and are transparent to the user, giving him the opportunity to better concentrate on the task to accomplish. Of course, the system must possess adequate "intelligence" to interpret correctly the human manipulative actions performed within the master virtual environment. It must analyse these actions, extract the critical task parameters and deduce (in real-time, in case of direct teleoperation, or off-line, in case of teleprogramming) the commands that need to be sent to the slave robot for execution.

We can thus conclude that the application of VR-based concepts and tools in the control of robotic telemanipulation systems, aims principally the development of a human-robot interactive communication framework that allows to exploit in a better extent: (a) from one hand, the dexterity and skills of the human operator to conduct dexterous manipulation tasks and solve complex decision problems, and (b) on the other hand, the capacities of robot manipulators to execute, with superior speed and accuracy, a combination of sensor-based primitive tasks.

4.1.2 Distance Training in Robot Manipulator Programming

Substantial application scenarios of VR technologies can be found in the field of education. If these technologies are combined with teleoperation concepts and tools they can lead to the development of very efficient *remote and virtual laboratory* platforms, aiming to enable distance training in a number of engineering disciplines. One such application is described in (Tzafestas et al., 2006), presenting a platform that aims to enable student training in robot manipulation and control technologies from any remote location via Internet. Access to robot manipulator arms and other similar mechatronic devices and laboratory equipment is often either limited by specific time restrictions or even not provided at all. One prohibitive factor is the high cost of such equipment, which makes it very difficult for many academic institutes to provide related laboratory training courses in their educational curricula for engineers. Therefore, the benefits from providing a means for any-time/any-place (virtual and/or remote) experimentation, in a "lab facilities sharing" context, are evident from a socio-economic point of view, apart from a pedagogical point of view related to the completeness and quality of practical training possibilities offered to their students.

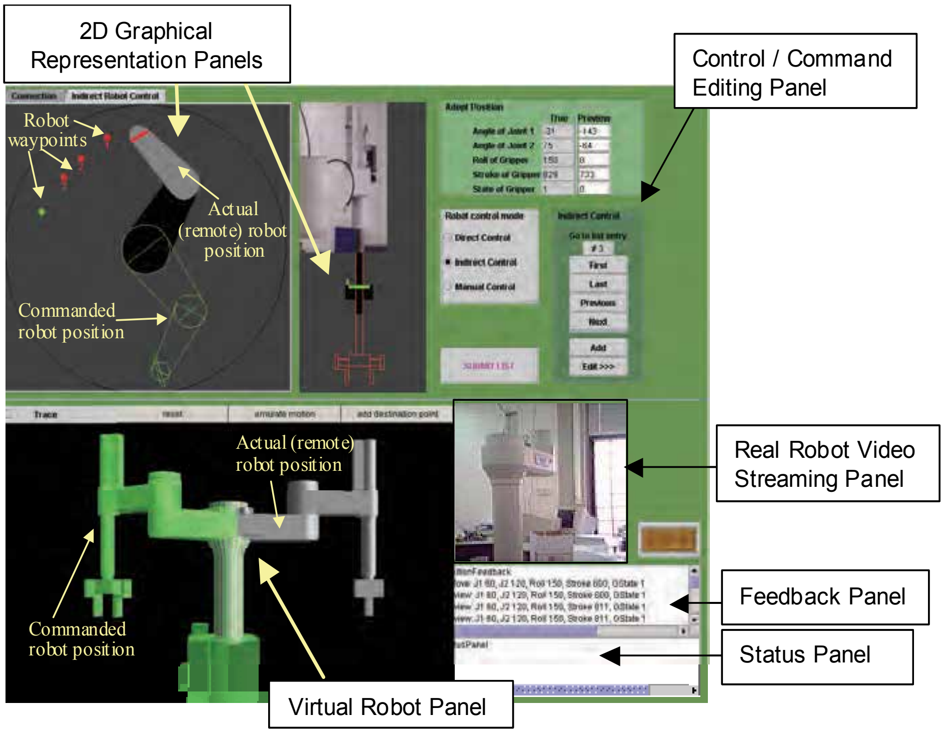


Fig. 7. The graphical user interface of the virtual robotic laboratory platform.

Taking into account these considerations, the work described in (Tzafestas et al., 2006) was directed towards the development of a virtual robot laboratory platform that will train students on how to program a robot manipulator arm, using the functionality and programming modalities provided by the real robotic system. The platform developed incorporates a robot's Teach Pendant emulator, as well as a virtual 3D robot animation panel integrated in the graphical user interface. The system enables students to create, edit and execute robot programs (i.e. complete motion sequences, such as a pick-and-place task), in exactly the same way as they would if they were using the real-robot's pendant tool. The program created can be previewed "locally" by the student/trainee in 2D and 3D animation modes, and can then be sent for execution: either (a) by the virtual robot simulation, incorporated as mentioned above in the graphical user interface, or (b) by a real, remotely located, robot manipulator (such as, in this case, a SCARA-type AdeptOne manipulator located in the premises of the robotics laboratory).
Fig. 7 shows the graphical user interface (GUI) of the virtual and remote robotic laboratory platform, which is developed based on Java technologies and integrates the following control panels:

- 2D graphical representation panels (top-view and side view), visualizing both actual and commanded robot configurations,
- a real-time video streaming panel, which is based on RTP and implemented using JMF, showing (when on-line) the real remote manipulator in motion,
- a control/command editing panel,
- an interactive panel providing an exact emulation of the robot's Teach Pendant, called Virtual Pendant,
- status and feedback panels providing real-time textual information on current robot state, and
- a virtual robot panel, implemented using Java3D API, providing 3D visualization of both the commanded (preview animation) and the current robot configuration.

The overall architecture of the remote laboratory platform is depicted in Fig. 8. The system is based on a client-server architecture, enabling users to connect via Internet (or LAN). It supports multiple connected users through the implementation of a specific protocol using TCP/IP sockets for communication and real-time data exchange with the "robot server". The robot server supports the following three remote control modes: (i) direct teleoperation control, (ii) indirect control, for robot teleprogramming via the command/ editing panel, and (iii) manual control, that is, robot manipulator programming using the Virtual Pendant functionalities. These control modes are inspired from the telerobotics field, and particularly from work proposing various "shared-autonomy" and "supervisory" remote control modalities. In *direct teleoperation*, every command issued by the user (human operator) locally, i.e. within the GUI (master control site), is immediately transferred for execution to the remote (slave) robot. At the same time two types of feedback displays are active: (a) a predictive display (both in the 2D and 3D graphical panel) immediately visualising the commanded robot motion according to the human operator issued commands, and (b) a real robot feedback display (also both in 2D and 3D animation), showing where the robot actually is (that is, visualising current remote robot configuration, information provided in real-time through continuous feedback from the remote site).

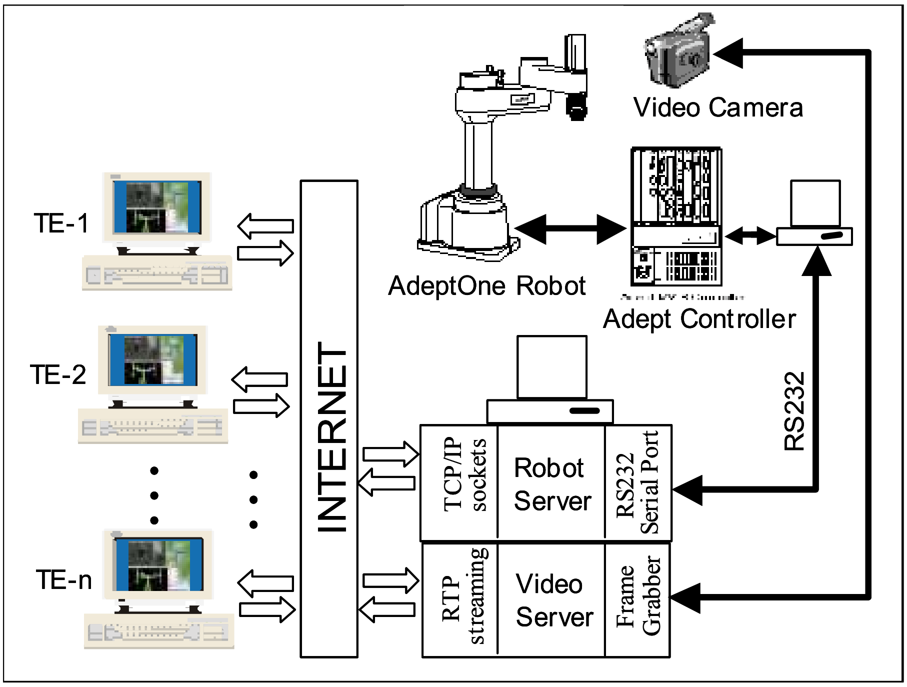


Fig. 8. Overall Architecture of the Virtual and Remote Robot Laboratory Platform.

As opposed to direct teleoperation, in the *indirect "teleprogramming"* control mode the commands are generated off-line, queued in a list and submitted to the robot in a subsequent time frame, when the human operator decides to do so. The idea is to be able to create a complete robot program off-line, test its validity and optimality, before actually sending the command sequences for execution by the real robot. Based on the functionality (robot command editing routines, waypoints list creation etc.) of this indirect teleprogramming mode, a third "*manual-control*" mode has been developed, which implements exactly the *Virtual Pendant* robot-programming scheme. This Virtual Pendant panel supports all the main functions of the real robot's pendant tool, and enables the student to learn and practice robot-programming routines locally.
A pilot study was conducted to evaluate the performance of the virtual and remote robotic laboratory platform. The objectives of this study were: (1) to explore to which extent the considered e-laboratory modalities can be efficiently implemented in practice and used by students to obtain practical training as a supplement to a theoretical course module (in this case, an introductory course on robotic manipulation), and (2) to explore the relative importance of various e-learning elements, particularly virtual vs. remote training modalities, in comparison with traditional hands-on experimentation. Experimental results show that students trained using the virtual training modality performed equally well as compared to students trained the classical way on the real robot, and were even seen to be more motivated, as revealed by the error rate related to the assimilation of high-level concepts and skills. These results are very interesting since they show that VR technologies and tools, when combined with telerobotic concepts and techniques in distance training scenarios, can prove very beneficial and contribute significantly to the development of very efficient virtual learning platforms, in many engineering (and not only) disciplines where

laboratory / hands-on training is of primary importance to complete a full educational curriculum. Of course, larger scale studies are still needed to draw more general conclusions regarding the feasibility of these goals and the acceptability of such new technologies by students in their education and training practice.

4.2 Mobile Robotics

Control of mobile robots is often considered as being identified with the field of autonomous and intelligent robotics. From the very early work in this field (for instance, the mobile robot Shakey[5] developed at Stanford University in the early 70's), mobile robots have been designed as machines that are supposed to work in autonomous fashion, according to variations of a straightforward *"sense-think-plan-act"* control architecture. Mobile robots must sense their environment, reason about it, plan their actions and move around autonomously. However, due to the complexity of such tasks and the limitations encountered as "bottlenecks" in the field of artificial intelligence, the presence of a human operator in the loop, being able to take control of the system, is considered imperative especially in unstructured and highly uncertain environments. The tendency is to design robotic systems that support some type of *"intelligent human-robot cooperation"*, in line with the principles developed in the field of telemanipulation, and related schemes such as: shared autonomy control, supervisory control, teleprogramming, etc.
The application of virtual and augmented reality techniques for the teleoperation of mobile robots follows the same guidelines as in the case of telemanipulators, with similar objectives being: (a) to assist the human operator and facilitate remote control by means of a highly interactive and intuitive interface, (b) to enhance information feedback through augmented reality models, (c) to support off-line mission planning and teleprogramming of the mobile robot by means of a VR-based graphical user interface. Mobile robot platforms can use any type of locomotion, namely: wheeled (indoor/outdoor), legged (walking robots), airborne, underwater, space robots etc. Of course, depending on the robot's environment (e.g. visibility conditions, rough terrain, distance from master station, communication bandwidth, etc.) and on the locomotion mechanisms (e.g. fast vs. slow moving robot, stability during motion etc.), the teleoperation interface should adapt accordingly to cope with task specific problems.
A typical application domain, where advanced teleoperation technologies are used in the field of mobile robotics, concerns underwater robots (Remotely Operated Vehicles - ROVs). One such example is presented in (Sayers et al., 95), where a teleprogramming concept was applied in the frame of a subsea ROV equipped with a robotic arm. The main problems in such systems concern: (i) the limited communication bandwidth (of some Kbits/sec) resulting in a round trip communication delay up to 7 seconds, which is typical of the sub-sea acoustic transmission channel; (ii) the limited visibility and distorted images received at the master control station, typical of underwater operation; and (iii) the effect of underwater currents on the mobility of the ROV platform. The use of virtual and augmented reality models as predictive displays in the master control GUI can help alleviate some of the difficulties related with this specific application environment. Figure 9 shows snapshots of the system in operation, both from the operator's interface and from the real robot performing a grasping task. However, the most exceptional instance highlighting the use of

[5] See: http://www.ai.sri.com/shakey/ for more information

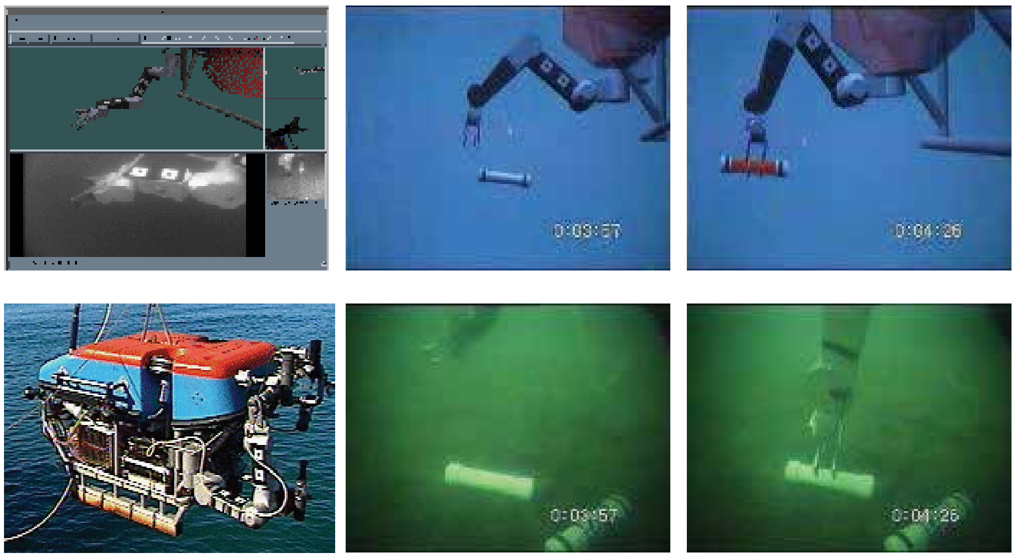


Fig. 9. Teleoperation of a subsea ROV. Up-line pictures designate the master control station, and down-line pictures show an image of the real submersible robot, as well as snapshots of the video feedback from the robot (JASON ROV) environment (adapted from (Sayers, 99), the GRASP laboratory, Pennsylvania University).

VR techniques in mobile robot teleoperation is undoubtfully the Mars Pathfinder Mission (NASA)[6]. Indeed, a VR technology based supervision and control workstation was designed to remotely command the Sojourner rover, which landed on the Mars planet by July 4th 1997. A good survey of these VR-based teleoperation technologies and their applications can be found in (Kheddar et al., 2000).

In the rest of this Section, we present two application examples regarding the use of VR techniques in mobile robot teleoperation. The first one concerns a mobile robotic assistant teleoperated in a modelled indoor navigation environment, while the second one focuses on the use of more advanced haptic interface to facilitate remote exploration of an unknown environment using a miniature mobile robot.

4.2.1 Teleoperation of a Mobile Robotic Assistant

The work presented in this paragraph was carried out in the framework of a research project called "HygioRobot" (Health-Robot), funded by the Greek General Secretariat for Research and Technology and the European Commission. The aim of the project was the development and implementation of control algorithms for a mobile service robot, consisting of an integrated robotic platform equipped with a vision system and a light manipulator (Tzafestas et al., 2000). The system was targeted towards a particular class of applications, namely to perform assistive tasks in a hospital environment.

[6] See: http://mars.jpl.nasa.gov/MPF/ and http://www-robotics.jpl.nasa.gov/ for details

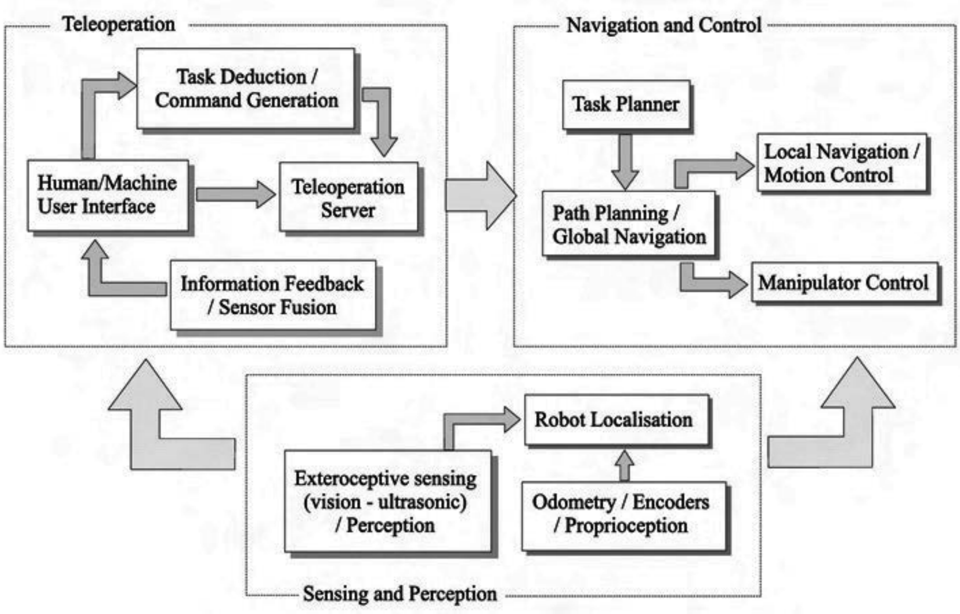

Teleoperation
Task Deduction / Command Generation
Human/Machine User Interface
Teleoperation Server
Information Feedback / Sensor Fusion
Navigation and Control
Task Planner
Local Navigation / Motion Control
Path Planning / Global Navigation
Manipulator Control
Robot Localisation
Exteroceptive sensing (vision - ultrasonic) / Perception
Odometry / Encoders / Proprioception
Sensing and Perception

Fig. 10. Teleoperated mobile robotic assistant: general architecture of the system (adapted from (Tzafestas et al., 2000))

These may include tasks such as transportation of specific items (like pharmaceuticals, lab specimens, medical records etc.), accompanying a patient from one location to another within the hospital building, or even surveillance of an area, in other words, a combination of simple displacement and manipulation (fetch-and-carry) operations in a potentially human crowded indoor environment.

The global architecture of the system, with the principal functional modules and their interconnections is shown in Fig. 10. It consists of the following main subsystems:

(a) The *navigation and control* subsystem, including the task planner, the global and local path planning and navigation modules, as well as the manipulator control module. Major issues that must be investigated include here: (i) the real-time collision avoidance to ensure safe operation of the mobile platform in a dynamic environment, (ii) the development of a number of autonomous low-level sensor-based behaviors, and (iii) the coordinated action of the mobile platform and robot manipulator to optimally exploit the redundancies offered by such an integrated mobile manipulation system,

(b) The *sensing and perception* subsystem, performing fusion and interpretation from a variety of sensory feedback information, provided by the odometry and optical encoders (proprioceptive feedback), as well as from the vision and ultrasonic sensors (exteroceptive feedback). The goal of this subsystem on its whole is to update internal representations regarding: (i) the robot's actual state (positioning, i.e. robot localization) and (ii) the external world map (i.e. dynamic moving obstacles etc.)

(c) The *teleoperation* subsystem, which aims at integrating the decision and action capacities of a human operator in the control loop, and consists of: (i) a multimodal user interface (which will be designed to enable Web-based remote control of the robotic platform), (ii) a sensory feedback acquisition and processing module, and (iii) a task deduction and command generation subsystem. All these modules are coordinated by a teleoperation server, which was designed to support various modes of operation ranging from direct on-line remote monitoring and control, to off-line teleprogramming or simple supervisory control of the system.

The work described briefly in the sequel focuses more specifically on the design and implementation of a multimodal teleoperation system for the mobile robotic assistant, integrating virtual reality techniques within a Web-based user interface, to assist the human operator and enhance the functionality and efficiency of the system. Efficiency in remote operation and control of a robotic system concerns: (a) making "good use" of the available communication bandwidth between the master and slave systems, and (b) achieving a "synergy" between the human operator and the robot, by enabling the system to best exploit and integrate (in terms of speed, precision and error recovery) both (i) the human operator capacity to take rapid decisions and intuitively indicate the most appropriate (coarse or detailed) plan for system action (e.g. robot motion) in complex situations, and (ii) the robotic system capacity to perform, with controlled speed and precision, a variety of autonomous operations.

To approach towards these general targets, a set of requirements have to be specified and fulfilled by the teleoperation system and all its sub-modules. The final system design must converge towards the merging between a number of often contradictory modalities, in search of an "optimum" compromise and increased "efficiency". By *multimodal teleoperation interface* we mean a system that supports: (a) multiple computer-mediated human/robot interaction media, including VR models and tools, or even natural gesture recognition etc., and (b) multiple modes of operation, with a varying degree of robot autonomy and, respectively, human intervention. The latter is a very important issue for the design of a telerobotic system, as has been already cited in previous sections. The modes of operation that have been considered for this teleoperation system included:

(a) *Direct teleoperation* control, based on on-line exchange of low-level commands and raw sensory signals.

(b) *Computer-aided teleoperation* of the mobile robot, with the master control system providing some form of assistance to the human operator, such as: (i) performing information feedback enhancement, like model-based predictive display, (ii) undertaking active control for some of the degrees of freedom (dof) of the system, substituting or complementing the human operator, or even (iii) providing some form of active guidance and model-based correction for the human operator's actions. Two main functions have been integrated in the system: an active anti-collision and an active motion-guide function, both based on the use of either a virtual reality model of the robotic platform and its task environment, or of a simple 2D top-view representation.

(c) *Shared-autonomy teleoperation* control of the robotic system, using a set of sensor-based autonomous behaviors of the robot. This mode of teleoperation control can be extended to incorporate a large set of intermediate-level, behavior-based, hybrid (qualitative/ quantitative) instructions, such as: move through points A, B, C while avoiding obstacles, pass through door on the left, move at distance d from wall, follow corridor etc. These commands trigger and make use of respective behavior-based control modes of the robot, incorporating automatic path generation functions. In other words, this mode of teleoperation control is based on some form of basic autonomy (local path planning and reactive sensor-based behaviors etc.) embedded on the slave

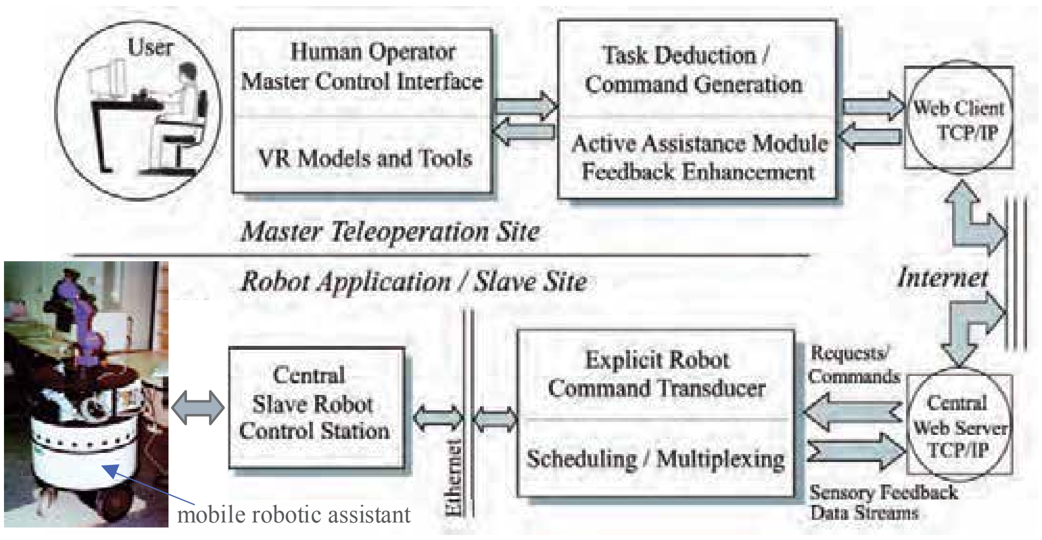


Fig. 11. Schematic representation of the overall teleoperation system for the mobile robotic assistant (adapted from (Tzafestas et al., 2000))

robot. Of course, the master control system should enable this form of remote control, by allowing the human operator to intuitively indicate the robot plan, interpreting his actions and transforming them into appropriate robot instructions that fall into this category. This natural/intuitive human/robot interaction in the context of a shared-autonomy teleoperation control architecture, are issues that will be discussed later on in this chapter.

(d) *Semi-autonomous teleoperation*, based on a set of high-level qualitative task-based instructions (such as go to location X, grasp object A on table B of room C etc.) This set of instructions must be built upon a combination of task-planning, path-generation and environment-perception modules incorporated on the robot control system.
(e) *High-level supervisory control*, that is, simple monitoring of sensory feedback, and limited human intervention on specific complex situations, requiring difficult decision making and task planning.
Depending on the specific application and the tasks to be performed, a combination of these control modes can be used for on-line monitoring and remote control of the mobile robot. The system, however, also supports a subset of these control modes in an *off-line teleprogramming* scheme, where the human operator controls the robot task in a simulated environment and checks the validity of his actions before actually sending the commands (registered action plan) to the slave robotic system for real execution. A schematic representation of the overall structure of the teleoperation platform, from the human operator (user) to the mobile robotic assistant, is shown in Fig. 11. To support the "multimodality" of operation, as defined above, the graphical user interface for the teleoperation of the mobile robotic assistant comprises four main components (Fig. 12):
(i) The *VR-panel*, where the 3D graphical models of the robotic system and its task environment are rendered. This simulation environment constitutes the first modality for inserting instructions (eg. motion commands) to the system in a natural and intuitive way.

The input devices used were: a joystick for virtual robot motion control, and a Spaceball for virtual camera navigation. Some form of sensory feedback information is also integrated in this virtual environment, like the actual robot position represented by a wireframe graphic model of the robot platform.
(ii) The *control-panel*, containing a 2D top-view graphical representation of the mobile robot environment (corridors, doors, rooms, obstacles etc.) and a command editing panel. The 2D model contains accurate map information of the whole indoor environment, where the robotic assistant is operating, allowing the user to obtain rapidly a top-view of any required region (using scrollbars or predefined region-buttons). The human operator will also have the ability, if needed, to directly edit commands that must be sent to the robot.
(iii) The *sensory-feedback panel*, where all the required sensory feedback signals are displayed (for instance a sonar map, indicating the location of obstacles detected by the robot). A *visual-feedback panel* was also integrated, displaying images obtained by the on-board robot camera. The refresh-rate of this video feedback is of course reduced, since the bandwidth available for communication through the Internet is limited and real-time access to other more critical information, such as actual robot location and sensor status, is indispensable.
(iv) The *status panel* displaying information on the actual status of the robot in textual form, as well as messages about actions and events.
Fig. 12 shows a snapshot of this prototype human operator interface, developed based on Java technology to facilitate Web-based operation of the system. The 3D graphics rendering routines for the VR panel are implemented using the Java3D API. An enhanced version of this system will constitute in the near future the Internet-based teleoperation control platform for this mobile robotic assistant.

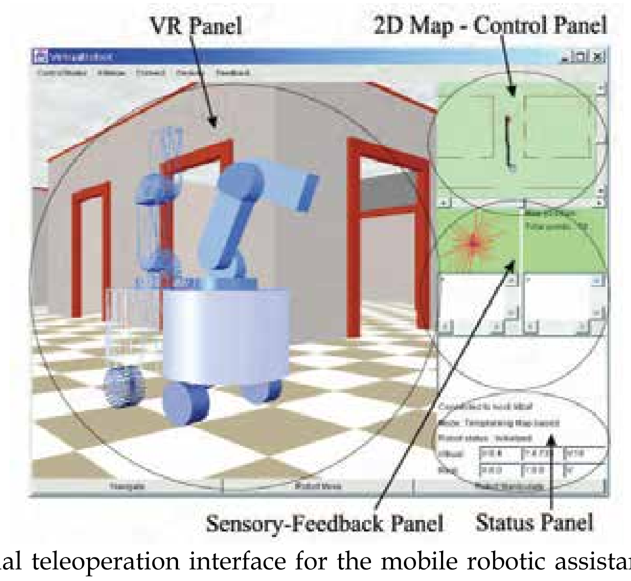


Fig. 12. Multimodal teleoperation interface for the mobile robotic assistant: First prototype implementation (adapted from (Tzafestas, 2001a))

Experiments with this system were very satisfactory, demonstrating convincingly that VR techniques and tools can be used efficiently in the teleoperation of mobile service robots. There

remain, however, several issues that need further attention and require future research efforts. One of the major difficulties is how to enable both: (a) the human operator to perform actions in a natural and intuitive manner within the master control environment, and (b) the system to interpret these actions, extract critical task-related parameters and synthesize appropriate robot commands. The first issue is related to the design of the human operator interface, where we have opted for the use of VR techniques to enable such an intuitive interaction with the system, while providing active assistance functionalities, as described above. On-line monitoring and analysis of the human operator's actions is then necessary, to deduce a correct robot task plan as specified/indicated by these actions. This means, in other words, to incorporate some form of "intelligence" in the master control environment, capable of performing these task-deduction and robot-command-extraction operations, based on observation of human actions within a simulated virtual representation of the slave site. One approach to this problem of *robot action tele-planning from observation and learning of VR-based human demonstration*, is described in (Tzafestas, 2001b).
In the same context of "*embedding intelligence*" *in telerobotic interfaces* to facilitate the human operator task and enhance system performance, an important issue concerns the development of systems integrating more advanced input/output modalities, inspired from works in the field of virtual reality (for instance, providing active assistance through haptic devices, as simple as a force feedback joystick, or more enhanced like a general-purpose haptic device such as the PHANTOM® devices from SensAble Technologies[7]). Such a research effort, involving active haptic display in a mobile robot teleoperation interface, is described in the following section.

4.2.2 Haptic Teleoperation for Remote Exploration

As discussed above, the use of VR technologies in mobile robot teleoperation is usually focused on providing mainly visual assistance to the human operator, via 3D modeling and predictive display of the remote environment, either for on-line guidance or for teleprogramming / tele-planning purposes. Since quite recently, this field is evolving towards a more fundamental pursuit of an efficient human-robot cooperation framework, where the system is designed according to schemes aiming to combine in an "optimal" way the capacities of humans and robots. Many researchers are currently concentrating on establishing such a "*synergetic and collaborative*" *telerobotic control framework*, as this was done in the telemanipulation case, like for instance the "shared intelligence" scheme proposed in (Bathia et al., 99), describing a VR-human interface for assisting human input in path planning for telerobots. For the teleoperation of a mobile robot, a simple application of this general principle could be to commit the human operator in performing the necessary global planning operations, which are more demanding in terms of complex reasoning and required "intelligence", while other more local tasks such as collision avoidance and trajectory optimisation are dedicated to the telerobotic system.
Our objective in the design of such advanced VR-based teleoperation systems is, always, to devise new methods to support efficient multimodal sensori-motor interaction and provide adequate multi-sensory information to the human operator, which will assist him (a) to more intuitively perceive important features of the remote environment (where the slave robot is moving), and (b) to indicate his intentions and issue teleoperation commands in a direct and natural way.

[7] http://www.sensable.com/products/phantom_ghost/phantom.asp

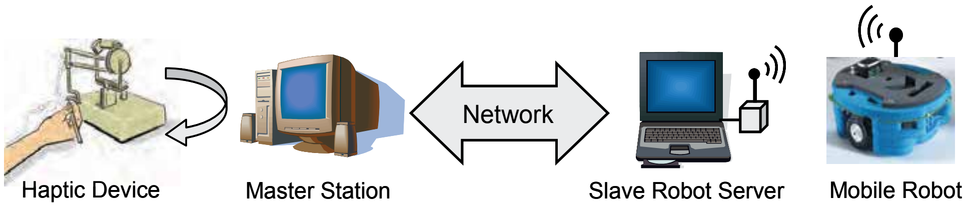


Fig. 13. Overview of a mobile robot "haptic teleoperation" system with master (left) and slave (right) side.

In this context, the application of haptic technologies is now considered as a very promising research direction, offering great potential as a means to revolutionize the way human-robot (and generally human-machine) interfaces are conceived. The term "*haptic*" comes from the Greek word "*αφή*" (touch), and usually refers to the sense of touch, and especially to the human hand that can act on the environment and perceive its physical characteristics. Exploiting human dexterity and manipulative skills within interactive VR systems constitutes a real-challenge for scientists and engineers in the field. Haptics constitutes now a rapidly evolving field of VR technologies, with new research communities active worldwide[8].

With the spread of low-cost haptic devices, haptic interfaces appear in many areas in the field of robotics. In telerobotics, the integration of haptic technologies and tools has given rise to a new research field that is now often termed as "*haptic teleoperation*". The main motivation behind this new field is the constant search to enhance transparency and naturalness of teleoperation systems, and promote performance a step further by enriching the human-telerobot interface with new sensori-motor interaction modalities, for *efficient human-robot skill transfer and cooperation*. Recently, haptic devices have been also used in the field of mobile robot teleoperation, particularly where mobile robots operate in unknown and dangerous environments performing specific tasks. Haptic feedback is shown to improve operator perception of the environment without, however, improving exploration time.

Teleoperated mobile robots are a major tool in the exploration of unknown and risky environments. Mines removal (Smith et al., 1992) and exploration of underwater structures (Lin et al., 97) are two common applications carried out through mobile robots. Robot motion is usually controlled by system operators with the help of a camera mounted on robot or inspecting the area from above. However, although vision systems provide much information of the environment, they require network bandwidth and much attention from the operator. To overcome this problem, haptic devices have been recently introduced as a way of enhancing operators perception of the robot environment. They provide operators with the additional sense of “feeling” the robot workspace, thus making it easier to avoid obstacles and reducing the average number of collisions.

However, the force rendering process yields a problem regarding how the haptic feedback affects the exploration time. Ideally, we would like the presence of force feedback to reduce the exploration time or at least not to increase it. In practice, though, this additional sense often adds more information for operators to interpret and leads to an increase in the navigation time, as for instance in (Palafox et al., 2006). Another important issue in mobile robot teleoperation is the selection of a proper driving mechanism. Usually, operators have to manually drive the mobile robot through obstacles by explicitly specifying the robot

[8] see, for instance: [www.eurohaptics.net] and [www.worldhaptics.org]

angular and linear velocity. By doing so, they are fully in charge of the robot motion and, as a clear viewpoint of the robot environment may sometimes not be available, they could accidentally drive the robot to collisions or choose longer paths than optimal ones.
In a recent work described in (Mitsou et al., 2006), a teleoperation scheme is presented for the case of remote exploration of a structured polygonal environment by a miniature mobile robot, with the use of a haptic device. During robot exploration, robot sensor measurements are used to build an occupancy grid map of the environment, which is displayed to the operator as a substitute for camera information. The operator can simultaneously exert two different types of commands: an "active" and a "guarded" motion command. Each command receives force feedback independently (without influencing one another) making force origin clear. A behavior-based system is then responsible for controlling the overall motion performed by the slave mobile robot. The commands received from the haptic device act as a general policy that the robot must follow.
An overview of the mobile robot haptic teleoperation system is shown in Fig 13. It consists of two sides: the master side, which contains the haptic device and the master station with the map-building module, and the slave side, which contains the mobile robot and a slave robot server with the behavior and the localization module. A more detailed view of the main system modules and their interconnection is schematically shown in Fig. 14. To develop a teleoperation interface that will facilitate intuitive teleguidance of a mobile robot exploring unknown environments (such as a rectangular maze), the first step was to implement a specific "*driving scheme*". According to this scheme, the haptic workspace is divided into three types of areas. These areas are not relative to the local coordinate system of the mobile robot, but rather absolute and relative to the local coordinate system of the master workstation monitor. There is a "neutral" area, corresponding to the area in the centre of the haptic workspace and implying a stop command. When selected, the robot immediately stops its motion. When the haptic end-point enters one of the Up, Down, Left or Right areas, the robot changes its orientation and moves according to the operator's command. For instance, if the Up area is selected, the robot will start moving upwards, as watched in the operator monitor. Finally, when the haptic end-point enters one of the "bi-directional" areas, a "combined command" will be issued and sent to the slave side. The robot will be instructed to move towards one direction and simultaneously "wall-follow" a wall. A combined motion command execution is illustrated in Fig. 15 (upper row).

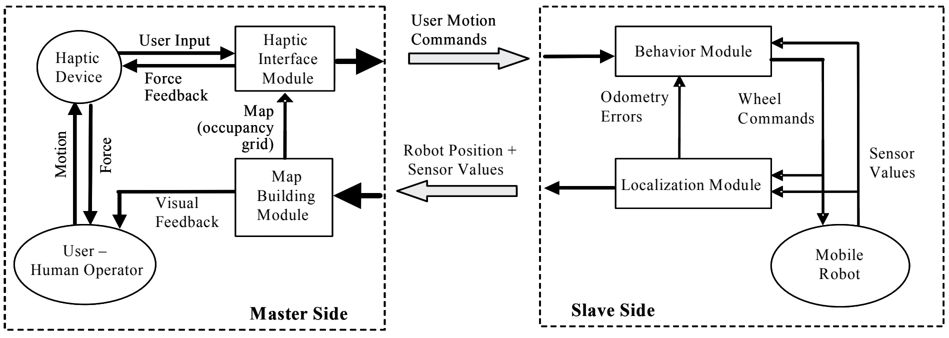


Fig. 14. Mobile Robot Teleoperation System Architecture.

The *Map-Building Module* is responsible for the creation of an "*occupancy grid map*" representation of the environment under exploration. It receives robot position and sensors values from the Localization Module of the slave robot server, and updates all affected cells. The map is constructed on-line and is displayed on the Master Computer. Cells that hold

value greater than a threshold are considered occupied (painted white), cells with value less than this threshold are considered empty (painted black) and cells with no value are considered unvisited (uncertain - painted gray). The constructed map is thus sequentially created and continuously updated on the master control monitor along with the robot position and orientation (Fig. 15). It forms the visual part of the information feedback provided by the interface to assist the human operator in monitoring the task execution and issuing proper navigation commands.
The second principal information feedback channel of the teleoperation interface concerns haptics. As already mentioned, *haptic feedback* is added to the system to enhance operator's perception of the remote environment, and assist teleoperation of the robot exploration task. The force fed back to the operator is generated based on a 2D virtual joystick model. When the haptic control point exits the "neutral area", a spring force is exerted to the operator attempting to restore end-point position inside this neutral area. The stiffness coefficient of this virtual spring depends on the absence or presence of an obstacle in this direction, that is, on whether the respective motion command is permissible or not. In case that no obstacle hinders the execution of a specific motion command (move forward/backward, turn left/right), the respective spring coefficient is set to a minimum value. This feature enables the operator to feel if he/she is actually exerting an "active" command. In case that an obstacle (e.g. a wall) is detected in the direction of a motion command, the respective spring coefficient is gradually switched to a much larger (max) value. The presence of a wall can thus be viewed as the origin of a virtual repulsive spring force that is applied to the mobile robot. In this case, the operator is feeling a force on his/her hand as if the wall is pushing the robot away, thus conveying the important information that the respective motion command is not permissible.

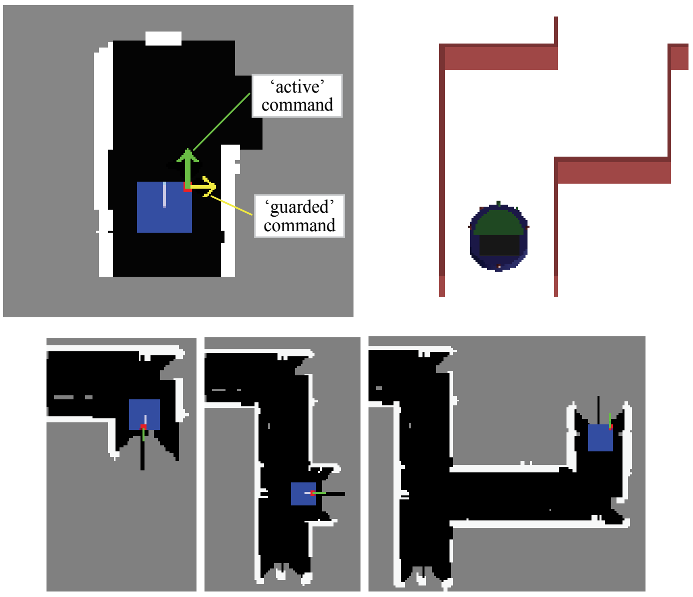


Fig. 15. Example of combined command execution (upper row, in the left, the map on the master station, in the right the VR simulator view), and Sequential Map Creation in the Master Teleoperation Interface (lower row) (adapted from Mitsou et al. 2006).

It is important to note that, in order to calculate the environmental force that is applied to the human operator via the haptic device, we do not directly use the current sensor values transmitted by the Localization Module. The force is generated according to the values of the cells of the occupancy grid map. If more than *n* (a threshold we use to deal with noisy sensor values) occupied cells (e.g. forming a virtual wall structure) are found in one direction, then the respective motion command is considered as not permissible. The virtual interaction force is then continuously computed according to the virtual joystick model described above, and sent to the Haptic Device for application. Apart from a spring model, a virtual damper model is also used to smooth the force and avoid large fluctuations in its value.

Additionally, a local robot model is implemented in the master station, to deal with delays in the communication between the master and the slave side. The robot position received from the *Localization Module* is used to update the local model estimation and the *Haptic Interface Module* considers the new updated position as the real robot position. In case of great latency in master-slave communication, if no local model is used the operator may not feel a wall coming closer to the robot. However, as the local model estimates the current robot position, the generated force is going to increase and the operator will feel the real distance from the wall.

In the *slave side*, the Mobile Robot is responsible for the exploration of the unknown environment. The robot used in these experiments was a Hemisson mobile robot[9]. It has a differential drive system, is equipped with six low-range infrared sensors and has no wheel encoders. It sends sensors measurements to a server computer (slave robot server) and receives from it the new speed commands for each wheel. The communication between robot and server computer was performed via a wireless link.

The *Behavior Module* in the master computer is responsible for the robot motion. Given a policy from the master side, the Behavior Module makes sure that the robot will follow it except for cases of possible collisions. Two different behaviors are implemented: a *collision detection* behavior and a *wall-follow* behavior. The first is activated when an obstacle is detected from the three front sensors. In this case, the robot stops when operators' command drives it towards the obstacle but accepts the commands that guide it away from the obstacle. The wall-follow behavior is activated by the operator policy. Under specific policies, the robot can follow walls and automatically turn on corners so as not to lose contact with them.

To test the effectiveness of this approach, two different kinds of experiments were performed. The first one was a comparison between this approach and a common teleoperation method, while the second one constituted a "drive-by-feel" experiment. Both experiments were performed on a simulated mobile robot, with a virtual world (the slave robot environment) on a separate remote computer. Also, a PHANTOM Omni™ haptic device was used for force-feedback generation in the master teleoperation interface. Results show that exploration of unknown environments can be completed in less time when haptic feedback is enabled. The proposed visuo-haptic interface is, thus, found to improve navigation time and decrease operator effort. Comparative evaluation experiments show that the proposed system, with the addition of haptic display and autonomous behaviors in the slave robot system, automates the task of exploration making it much easier for the operators to remotely navigate the robot through an unknown world. Moreover, with no visual information available to the operator, exploration tasks

[9] The "Hemisson" webpage: [http://www.hemisson.com]

consisting of finding the exit in an unknown maze can be completed successfully when force feedback is enabled ('drive by feel' experiment).
Conclusively, the use of VR technologies in such teleoperation interfaces is seen to improve operators perception of the environment, and makes a step towards facilitating and automating teleoperated mobile robot exploration tasks. In the future, work is still needed to extensively test such approaches in practical scenarios, particularly generalizing related system architectures so as to successfully cope with applications involving any type of unstructured environment, as well as to examine system behavior under time delay conditions and develop algorithms to cope with such latency in master-slave round-trip communication.

5. Conclusion – Future Research Directions

This chapter has reviewed fundamental concepts and technologies of the general interdisciplinary field described usually by a combination of the terms: Virtual, Augmented or Mixed Reality systems, with the emphasis being on their applications in robot teleoperation. We have first analysed the basics of VR and AR systems, which have shown a great progress of research and development activities during the last decade, demonstrating a constantly increasing repertoire of useful practical applications in diverse domains of human activity. We have then described application scenarios of such VR technologies in the general field of robotics, with the particular focus on telerobotic applications.
We have started by presenting a brief historical survey of the field of telerobotics, and identified the major profits that are related to the integration of VR and AR techniques. Virtual environments can be seen as a means to achieve natural, intuitive, multimodal human/computer (and generally human/machine) interaction; in this sense, a VE can function as an efficient mediator between a human operator and a telerobot, with the main objectives being:
(a) to *enhance human perception* of the remote task environment and therefore improve *transparency* of the telerobotic system, by enriching the visual information (complemented by other form of sensory and sensori-motor stimuli) provided to the user, thus conveying complex data in a more natural and easier way;
(b) to contribute to the solution of the *time-delay problem* in bilateral teleoperation and improve *stability* of the telerobotic system, by extending the concept of predictive displays and offering a range of control metaphors for both operator assistance and robot autonomy sharing.
We have presented a number of successful case studies, where VR techniques have been effectively applied in telerobotics, for the two main robotic systems categories, namely (i) robot manipulators and (ii) mobile robotic vehicles. A long-distance parallel telemanipulation experiment was described, where an intermediate virtual task representation was used involving direct hand actions by means of a VR glove device. The use of telerobotic technologies in a distance training (virtual and remote laboratory) application has been also demonstrated, with very promising results in this important domain. As related to the field of mobile service robotics, two application scenarios have been described, to highlight the benefits that can result from the integration of VR-based interfaces for the teleoperation of robotic vehicles for a variety of tasks, including service / intervention tasks and remote exploration. The link with the field of haptics is also

discussed, as an introduction to the new field of haptic teleoperation systems, combining concepts, methods and techniques from the field of haptics as a means to further enhance performance (in terms of a trade-off between transparency and stability), and approach the aforementioned objectives for general telerobotic systems.
In this context, we can assess that the great challenges related to the development of efficient telerobotic systems still remain, as can be described by the long-term vision of creating a framework for an effective cooperation between humans and robots (or telerobots). According to such a visionary objective, an ideal intuitive and natural interaction will enable the system to exploit in a full extent: (a) from one hand, the abilities and skills of the human operator in performing specified operations (in terms of dexterity, as well as rapid decision-making and planning skills in complex environments), and (b) on the other hand, the capacities of automated robotic systems to perform autonomously (with superior speed and accuracy) specific control and manipulation physical tasks. In this direction, VR technologies in conjunction with methods and tools from other related fields (computer vision, artificial intelligence) seem to provide an excellent framework for realising such advances, by: (a) inspiring the development of new human-centered teleoperation (and, generally, human-robot communication) schemes, and (b) supporting the development of advanced user-friendly telerobotic interfaces. VR is now considered as a key technology that will contribute significantly towards such long-term goals, and revolutionize the way human-machine systems are conceived in the years to come.

6. References

Anderson, R. J. and Spong, M. W. (1992). "Asymptotic Stability for Force Reflecting Teleoperators", *Intern. Journal of Robotics Research*, vol. 11, no. 2, pp.135-149, 1992.

Azuma, R. T. (1997). "A Survey of Augmented Reality," *Presence*, 6 (4), August 1997.

Bathia, P., and Uchiyama, M. (1999). "A VR-Human Interface for Assisting Human Input in Path Planning for Telerobots", *Presence*, vol.8, no.3, pp.332-354, June 1999.

Bejczy, A. K., Kim, W. S., and Venema, S. (1990). "The Phantom Robot: Predictive Displays for Teleoperation with Time Delay", *1990 IEEE Int. Conf. on Robotics and Automation*, pp. 546-551, 1990.

Burdea, G. and Coiffet, P. (1994). *Virtual Reality Technology*, John Wiley, 1994.

Ellis, S.R. (1995). "Virtual environments and environmental instruments," Chapter 2, in: *Simulated and Virtual Realities: Elements of Perception*, K. Carr and R. England (eds.), Taylor & Francis, London, pp. 11-51, 1995.

Freund, E., and Rossmann, J. (1999). "Projective Virtual Reality: Bridging the Gap between Virtual Reality and Robotics", *IEEE Transactions on Robotics and Automation*, vol. 15, no. 3, pp. 411-422, June 1999.

Goldberg, K. (2000). "Introduction: The Unique Phenomenon of a Distance", in: *The Robot in the Garden. Telerobotics and Telepistemology in the Age of the Internet*. K. Goldberg (ed.), MIT Press 2000.

Gracanin, D. and Valavanis, K. P. (1999). "Autonomous Underwater Vehicles", Guest Editorial, *IEEE Robotics and Automation Magazine*: Special Issue on Design and Navigation of Autonomous Underwater Vehicles, vol.6, no.2, June 1999.

Hirzinger, G., Brunner, B., Dietrich J., and Heindl, J. (1993). "Sensor-Based Space Robotics-ROTEX and Its Telerobotic Features", *IEEE Transactions on Robotics and Automation*, vol. 9, no. 5, pp. 649-663, 1993.

Joly, L. D. and Andriot, C. (1995). “Imposing Motion Constraints to a Force Reflecting Telerobot through Real-Time Simulation of a Virtual Mechanism”, *IEEE Intern. Conf. on Robotics and Automation* (ICRA'95)}, pp.357-362, 1995.

Kheddar, A., Tzafestas, C., Coiffet, P., Kotoku, T., and Tanie, K. (1997). “Multi-Robot Teleoperation Using Direct Human Hand Actions”, *International Journal of Advanced Robotics*, Vol. 11, No. 8, pp. 799-825, 1997.

Kheddar, A., Chellali, R., and Coiffet, P. (2000). "*Virtual Reality Assisted Teleoperation*", in: *VE Handbook*, K. M Stanney Edts, December 2000.
(See also: http://vehand.engr.ucf.edu/revised2.htm)

Kim, W. S. (1996). “Virtual Reality Calibration and Preview/Predictive Displays for Telerobotics”, *Presence*, vol. 5, no. 2, pp.173-190, 1996.

De Laminat, P. (1993). *Automatique - Commande des systèmes linéaires*. Edition Hermès, Paris, 1993.

Lin, Q. and Kuo, C. (1997). “Virtual tele-operation of underwater robots,” in Proceedings of the *1997 IEEE International Conference on Robotics and Automation* (ICRA'97).

Mitsou, N., Velanas, S., and Tzafestas, C.S. (2006). "Visuo-Haptic Interface for Teleoperation of Mobile Robot Exploration Tasks," in Proc: *The 15th IEEE International Symposium on Robot and Human Interactive Communication* (RO-MAN'06), University of Hertfordshire, Hatfield, United Kingdom, pp. 157-163, 6-8 September 2006.

Niemeyer, G., and Slotine, J. J. (1991). “Stable Adaptive Teleoperation”, *The IEEE Journal of Oceanic Engineering*, vol. 16, no.1, pp.152-162, 1991.

Palafox, O.M., Lee, D., Spong, M. W., Lopez, I., and Abdallah, C. (2006). "Bilateral Teleoperation of Mobile Robots over Delayed Communication Network: Implementation", *IROS'2006*, Beijing, China, October, 2006.
[http://decision.csl.uiuc.edu/~robotics/index.htm]

Rastogi, A., Milgram, P., and Drascic, D. (1996). “Telerobotic Control with Stereoscopic Augmented Reality”, *SPIE Proceedings*, Volume 2653: Stereoscopic Displays and Virtual Reality Systems III, pp 135-146, San Jose, Feb. 1996.

Rosenberg, L. B. (1993). “The use of virtual fixtures to enhance telemanipulation with time delay”, *ASME Winter Annual Meeting on Haptic Interfaces for Virtual Environments and Teleoperator Systems*, New Orleans, Louisiana, 1993.

Sayers, C.P., Lai, A. and Paul, R.P. (1995). “Visual Imagery for Sub-sea Teleprogramming”, *IEEE International Conf. on Robotics and Automation*, May, 1995.

Sayers, C. (1999). *Remote control robotics*, Springer Verlag, 1999.

Schulz, D., et al. (2000). “Web Interfaces for Mobile Robots in Public Places”, *IEEE Robotics and Automation Magazine*, vol.7, no.1, pp.48-56, March 2000.

Sheridan, T. B. (1992). *Telerobotics, Automation and Human Supervisory Control*, The MIT Press, 1992.

Simmons, R., et al. (2000). “Lessons Learned from Xavier”, *IEEE Robotics and Automation Magazine*, 7(2), 2000.

Smith, F., Backman, D., and Jacobsen, S. (1992). “Telerobotic manipulator for hazardous environments,” *Journal of Robotic Systems*, 1992

Stein, M. R., (2000). “Interactive Internet Artistry. Painting on the World Wide Web with the PumaPaint Project”, *IEEE Robotics and Automation Magazine*, 7(2), June 2000.

Taylor, K., Dalton, B. (2000). “Internet Robots: A New Robotics Nich”, *IEEE Robotics and Autom. Magazine*, 7(1), (special issue: Robots on the Web), March 2000.

Thrun, S., et al. (1999). “MINERVA: A Second-Generation Museum Tour-Guide Robot”, *1999 IEEE Intern. Conf. on Robotics and Automation* (ICRA'99).

Tzafestas, C. S., and Coiffet, P. (1997). “Computing Optimal Forces for Generalized Kinesthetic Feedback on the Human Hand during Virtual Grasping and Manipulation”, *IEEE Int. Conf. on Robotics and Automation* (ICRA'97), pp. 118-123, Albuquerque, New Mexico, 20-25 Avril, 1997.

Tzafestas, C. S., and Valatsos, D. (2000). "VR-based Teleoperation of a Mobile Robotic Assistant: Progress Report," *Technical Report* DEMO 00/13, National Center for Scientific Research "Demokritos", Inst. of Informatics and Telecommunications, October 2000.

Tzafestas, C. S. (2001a). “Multimodal Teleoperation Interface integrating VR Models for a Mobile Robotic Assistant”, in: *Proc. 10th International Workshop on Robotics in Alpe-Adria-Danube Region* (RAAD'2001), Vienna, Austria, May 16-18, 2001.

Tzafestas, C. S. (2001b). “Teleplanning by Human Demonstration for VR-based Teleoperation of a Mobile Robotic Assistant”, in: *Proc. 10th IEEE International Workshop on Robot-Human Interactive Communication* (ROMAN’2001), Bordeaux and Paris, Sept. 18-21, 2001.

Tzafestas, C. S., Palaiologou, N., Alifragis, M. (2006). "Virtual and Remote Robotic Laboratory: Comparative Experimental Evaluation", *IEEE Transactions on Education*, vol. 49, no. 3, pp. 360- 369, 2006.

Vertut, J., and Coiffet, P. (1984). *Les Robots: Téléopération. Tome 3A: Evolution des technologies. Tome 3B: Téléopération assistée par ordinateur*. Edition Hermes, Paris, 1984.

Simulation and Experimental Tests of Robot Using Feature-Based and Position-Based Visual Servoing Approaches

M. H. Korayem, F. Heidari, H. Aliakbarpour
Robotic Research Laboratory, College of Mechanical Engineering, Iran University of Science & Technology, Tehran, Iran

1. Introduction

Discussion of visual control of robots has been introduced since many years ago. Related applications are extensive, encompassing manufacturing, teleoperation and missile tracking cameras as well as robotic ping-pong, juggling and etc. Early work fails to meet strict definition of visual servoing and now would be classed as look-then-move robot control (Corke, 1996). Gilbert describes an automatic rocket-tracking camera which keeps the target centered in the camera's image plane by means of pan/tilt controls (Gilbert et al., 1983). Weiss proposed the use of adaptive control for the non-linear time varying relationship between robot pose and image features in image-based servoing. Detailed simulations of image-based visual servoing are described for a variety of manipulator structures of 3-DOF (Webber &.Hollis, 1988). Mana Saedan and Marcelo H. Ang worked on relative target-object (rigid body) pose estimation for vision-based control of industrial robots. They developed and implemented a closedform target pose estimation algorithm (Saedan & Marcelo, 2001). Skaar et al. use a 1-DOF robot to catch a ball. Lin et al. propose a two-stage algorithm for catching moving targets; coarse positioning to approach the target in near-minimum time and `fine tuning' to match robot acceleration and velocity with the target.
Image based visual controlling of robots have been considered by many researchers. They used a closed loop to control robot joints. Feddema uses an explicit feature-space trajectory generator and closed-loop joint control to overcome problems due to low visual sampling rate. Experimental work demonstrates image-based visual servoing for 4-DOF (Kelly & Shirkey, 2001). Haushangi describes a similar approach using the task function method and show experimental results for robot positioning using a target with four circle features (Haushangi, 1990). Hashimoto et al. present simulations to compare position-based and image-based approaches (Hashimoto et al., 1991).
In simulating behavior and environment of robots many researches have been done. Korayem et al. designed and simulated vision based control and performance tests for a 3P robot by visual C++ software. They minimized error in positioning of end effector and also they analyzed the error using ISO9283 and ANSI-RIAR15.05-2 standards and suggested ways to improve error (Korayem et al., 2005, 2006). They used a camera which was installed on end effector of robot to find a target and with feature-based-visual servoing controlled end effector of robot to reach the target. But the vision-based control in this work is

implemented on 6R robot. The two cameras are mounted on the earth, i.e., the cameras observe the robot we can call the system “out-hand" (the term “stand-alone" is generally used in the literature). The closed-form target pose estimation is discussed and used in the position-based visual control. The advantage of this approach is that the servo control structure is independent from the target pose coordinates and to construct the pose of a target-object from 2 dimension image plane, two cameras are used. Image feature points in each of the two images are to be matched and 3-D information of the coordinates of the target-object and its feature points are computed by this system.
This chapter starts by introducing the 6R and 3P robots which are designed and constructed in IUST robotic research Lab. (Figs. 1, 2). Modeling and simulation process to solve direct and inverse kinematics of the robot are prescribed. After discussing simulation software of 6R and 3P robots, we simulated control and performance tests of robots and at last the results of tests according to ISO9283 and ANSI-RIAR15.05-2 standards and MATLAB are analyzed.

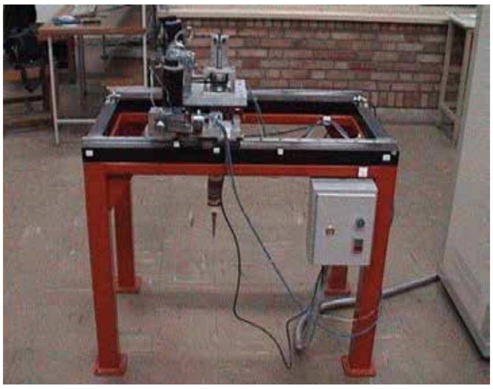


Fig. 1. 3P robot configuration.

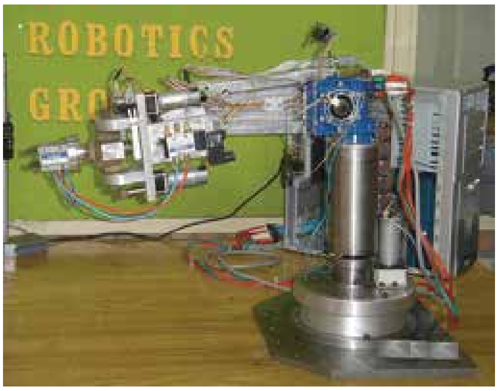


Fig. 2. 6R robot configuration.

The robots were designed on the assumption that each joint has an independent DC motor actuator with gear reduction and measuring angular joint position sensor. In keeping with this, mechanical design of the robot was done using Mechanical Desktop and manufacturing process of the mechanical parts of the robot was developed. Kinematics and dynamics equations of the robots have been derived.

2. Kinematics Equations

Within kinematics one studies the position, velocity and acceleration, and all higher order derivatives of the position variables. The kinematics of manipulators involves the study of the geometric and time based properties of the motion, and in particular how the various links move with respect to one another and with time.

2.1 Direct kinematics of 3P robot

For an n-axis rigid-link manipulator, the *direct kinematics solution* gives the coordinate frame, or pose, of the last link. For 3P robot direct kinematics equations will be as follow:

$$T = A_1^0(d_1)A_2^1(d_2)A_3^2(d_3) \tag{1}$$

Where, d_1, d_2, d_3 are generalized coordinates. After expanding right side of the equation (1), we have:

$$T = \begin{bmatrix} 0 & 1 & 0 & w_1 \\ 0 & 0 & 1 & w_2 \\ 1 & 0 & 0 & w_3 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2}$$

Where w1, w2, w3 are prismatic joints displacement in x, y and z directions, respectively.

2.2 Inverse kinematics of 3P robot

Given a desired position and orientation for the end effector of robot, for finding values of the joint parameters which satisfy the direct equations (2), we need to solve inverse kinematics of robot. For 3P robot solving inverse kinematics equation, leads to:

$$\begin{cases} w_2 = d_3 \\ w_3 - d_1 = 0 \\ w_1 = d_2 \end{cases} \rightarrow \begin{cases} d_1 = w_3 \\ d_2 = w_1 \\ d_3 = w_2 \end{cases} \tag{3}$$

2.3 Direct kinematics of 6R robot

According to the Denavit-Hartenberg notation for the 6R robot, Table of the D-H parameters will be as follow:

AXIS	θ	d	a	α
1	θ_1	d_1	0	$-\pi/2$
2	θ_2	0	a_2	0
3	θ_3	0	a_3	0
4	θ_4	0	a_4	$\pi/2$
5	θ_5	0	0	$-\pi/2$
6	θ_6	d_6	0	0

Table 1. Kinematics parameters for the 6R robot

According to Table 1 by multiplying the link transform matrices for the robot, the total transformation matrix will be:

$${}^{0}_{6}T = {}^{0}_{1}T\ {}^{1}_{2}T\ {}^{2}_{3}T \ \ldots\ {}^{5}_{6}T \tag{3}$$

${}^{0}_{6}T$ determines position and orientation of end effector with respect to base coordinate. If positions of joints are determined by position sensors, the pose of end effector will be determined according to ${}^{0}_{6}T$. For 6R robot this matrix will be as follow:

$$T_0^6 = \begin{bmatrix} n_x & o_x & a_x & P_x \\ n_y & o_y & a_y & P_y \\ n_z & o_z & a_z & P_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4}$$

Where:

$n_x = C_1(C_5C_6C_{234} - S_6S_{234}) - S_1S_5C_6$ $\quad n_y = S_1(C_5C_6C_{234} - S_6S_{234}) + C_1S_5C_6$ $\quad n_z = -C_5C_6S_{234} - S_6C_{234}$

$P_x = -C_1(d_6S_5C_{234} - a_4C_{234} - a_3C_{23} - a_2C_2) - d_6S_1C_5$ $o_y = -S_6(C_5S_1C_{234} + C_1S_5) - C_6S_1S_{234}$ $o_z = S_{234}C_5S_6 - C_6C_{234}$

$P_z = d_6S_5S_{234} - a_4S_{234} - a_3S_{23} - a_2S_2 + d_1$ $o_x = -C_1(C_5S_6C_{234} + C_6S_{234}) + S_1S_5S_6$ $a_x = -C_1S_5C_{234} - S_1C_5$

$P_y = -S_1(d_6S_5C_{234} - a_4C_{234} - a_3C_{23} - a_2C_2) + d_6C_1C_5$ $a_y = -S_1S_5C_{234} + C_1C_5$ $a_z = S_{234}S_5$

Using the notational shorthand as follow:

$$Cos\theta_i = C_i,\quad Sin\theta_i = S_i, Cos(\theta_i + \theta_j) = C_{ij},\quad Sin(\theta_i + \theta_j) = S_{ij}$$

2.4 Inverse kinematics of 6R robot

For a given desired position and orientation of the end effector in order to find values for the joint angles which satisfy the direct equations, we need to solve inverse kinematics of robot. Inverse kinematics equations for the 6R robot will be as follow:

$$\theta_1 = \tan^{-1}\left(\frac{P_y - d_6a_y}{P_x - d_6a_x}\right) \quad and \quad \theta_1 = \theta_1 + \pi \tag{5}$$

$$\theta_5 = \tan^{-1}\left(\frac{\pm d_6[1 - (a_yC_1 - a_xS_1)^2]^{1/2}}{P_yC_1 - P_xS_1}\right) \tag{6}$$

$$\theta_6 = \tan^{-1}\left(\frac{o_xS_1 - o_yC_1}{n_yC_1 - n_xS_1}\right) \; for\; \theta_5 > 0 \; and \; \theta_6 = \theta_6 + \pi \; for \; \theta_5 < 0 \tag{7}$$

$$\theta_{234} = \tan^{-1}\left(\frac{-a_z}{a_xC_1 + a_yS_1}\right),\; if\; \theta_5 > 0, else\; \theta_{234} = \theta_{234} + \pi \tag{8}$$

$$\theta_2 = -\tan^{-1}\left(\frac{\pm[1 - (w/q)^2]^{1/2}}{w/q}\right) + \tan^{-1}\left(\frac{u}{t}\right) \tag{9}$$

$$\theta_3 = \tan^{-1}\left(\frac{u - a_2S_2}{t - a_2C_2}\right) - \theta_2\,, \qquad \theta_4 = \theta_{234} - \theta_2 - \theta_3 \tag{10,11}$$

Where:

$$t = P_xC_1 + P_yS_1 + d_6S_5C_{234} - a_4C_{234} \quad , u = -p_z + d_1 - a_4S_{234} + d_6S_5S_{234}$$

$$w = \frac{t^2 + u^2 + a_2^{\,2} - a_3^{\,2}}{2a_2} \quad , q = \sqrt{t^2 + u^2}$$

Above equations are used to simulate robot motion.

3. Dynamic Equations

Manipulator dynamics is concerned with the equations of motion, the way in which the manipulator moves in response to torques applied by the actuators, or external forces. The equations of motion for an *n*-axis manipulator are given by:

$$\vec{Q} = M(\vec{q}).\ddot{\vec{q}} + C(\vec{q},\dot{\vec{q}})\dot{\vec{q}} + F(\dot{\vec{q}}) + G(\vec{q}) \tag{12}$$

Where $\vec{q}$ is the generalized joint coordinates vector; M is the symmetric joint-space inertia matrix; C describes Coriolis and centripetal effects.

F describes viscous and Coulomb friction and is not generally considered part of the rigid-body dynamics; G is the gravity loading and *Q* is the vector of generalized forces associated with the generalized coordinates *q*.

3.1 Dynamic equations of 3P robot

$$
\begin{aligned}
F_1 &= (m_1 + m_2 + m_3 + m_p)\ddot{d}_1 - g(m_1 + m_2 + m_3 + m_p);\\
F_2 &= (m_2 + m_3 + m_p)\ddot{d}_2\\
F_3 &= (m_3 + m_p)\ddot{d}
\end{aligned}
\tag{13}
$$

Where m_1, m_2, m_3 and m_p are masses, F_1, F_2, F_3 are input torques to motors No. 1, No. 2 and No. 3, respectively.

3.2 Dynamic equations of 6R robot

Dynamic equations of the 6R robot, based on equation 12, are derived by means of Lagrange-Euler method and also Newton-Euler method, using Mathematica software. Dynamic equations which derived by these two individual methods, are compared with each other. The simplified dynamics equation for 5th joint which is simplified by assuming the angular velocity of all the six joints are approximately zero is as follows:

$$
\begin{aligned}
\tau_5 = {} & -\frac{1}{2} g \left(C_{234} C_5 S_1 + C_1 S_5\right) d_6 m_6 + \frac{1}{4}\Big[4C_5 (S_{234} S_6 C_6 (I_{xx_6} - I_{yy_6})\ddot{\theta}_1\\
& - d_6 m_6 (S_{34} a_2 \ddot{\theta}_2 + S_4 a_3 (\ddot{\theta}_2 + \ddot{\theta}_3))) + 2S_5((-a_1 - C_2 a_2 - C_{23} a_3 - C_{234} a_4)\\
& d_6 m_6 \ddot{\theta}_1 - 2S_6 C_6 (I_{xx_6} - I_{yy_6})(\ddot{\theta}_2 + \ddot{\theta}_3 + \ddot{\theta}_4) + (4(S_6{}^2 I_{xx_6} + I_{yy_5} + C_6{}^2 I_{yy_6})\\
& + d_6{}^2 m_6)(C234\ \ddot{\theta}_1 + \ddot{\theta}_5)\Big]
\end{aligned}
\tag{14}
$$

4. Simulator software

For increasing the efficiency of designed robots packages for simulation of control and movement of robots are developed. In these packages by using a designed interface board, movement signals to control the robots are sent to robot.

4.1 simulation package for 6R robot

To improve the applications of 6R robot a software operating in windows system was written in Visual Basic programming language. In this program kinematics equations for robot are solved and simulated graphically. This simulator software has different parts such as *"Direct Kinematics, Inverse Kinematics, Direct Dynamics, Inverse Dynamics, Path Planning Control and Experiments"*. Main menu of program is shown in Fig. 3.

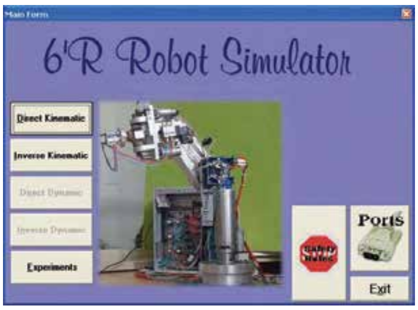
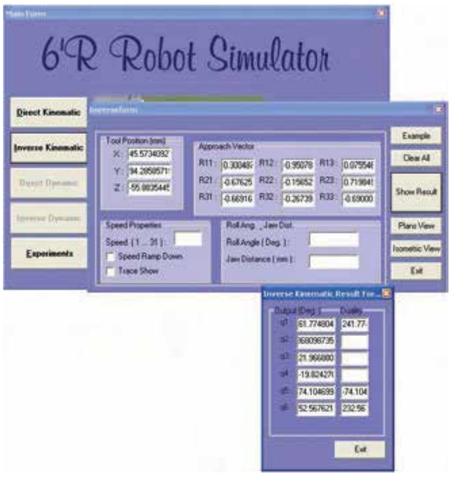


Fig. 3. Main menu of 6R robot program.



Tool Position (mm)
X: 45.5734092
Y: 94.2858571
Z: -55.8835445

Approach Vector

R11:	0.30048	R12:	-0.95078	R13:	0.075546
R21:	-0.67625	R22:	-0.15652	R23:	0.71984
R31:	-0.66916	R32:	0.26739	R33:	-0.69000

Speed (1 ... 31):
Roll Angle (Deg.):
Jaw Distance (mm):

	Output (Deg.)	Duality
q1	61.774804	241.77
q2	368096735	
q3	21.966800	
q4	-19.824271	
q5	74.104699	-74.104
q6	52.567621	232.56

Fig. 4. Inverse kinematics and graphical simulation menu of 6R robot.

In direct kinematics Section, user by entering joint parameters can obtain the position and orientation of end effector. In inverse kinematics Section for a given position and approach vector joint parameters are computed (Fig. 4). In point to point Section, user can define pose of tool for more than one point. Also a smooth curve for path of the end effector, path trajectory and torque curves can be obtained. Joints torques is computed using sliding mode control algorithm.

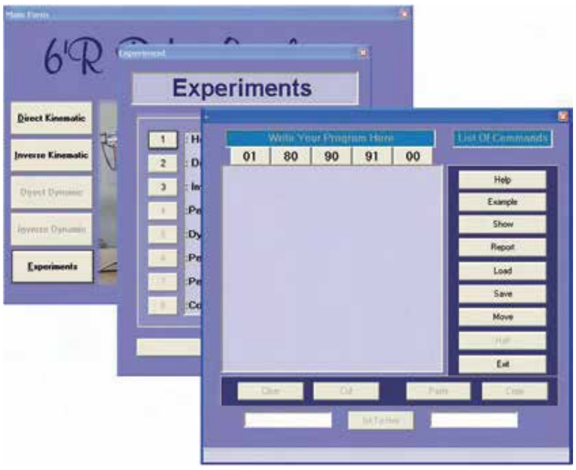


Fig. 5. Experiment section menu of 6R program.

In experiments part after selecting one of designed experiments, user can define codes for robot motion according to experiments instruction and simulation results and data can be reported (Fig. 5).

4.2 simulation package for 3P and 6R robots

To simulate control and test of 6R and 3P robots, we have used object oriented software Visual C++6. This programming language is used to accomplish this plan because of its rapidity and easily changed for real situation. In this software, the picture is taken in bitmap format through two fixed cameras which are mounted on the earth or by a camera which is installed over the end effector in the capture frame module and it is returned in form of array of pixels. Image capturing is switched between two cameras. After image processing, objects in pictures are saved separately, features are extracted and target-object and end effector will be recognized among them according to their features. Then position coordinates of target-object and end effector are estimated. After each motion of joints new picture is taken from end effector and this procedure is repeated until end effector reach to target-object.

With images from these two fixed cameras or camera on the end effector, positions of objects are estimated in image plane coordinate, usually, to transform from image plan coordinates to the reference coordinates system, mapping and calibrating will be used. In this program, in order to do this mapping which is a non-linear mapping, we have used a neural network to transform these coordinates to global reference coordinate. Mapping system needs extra work and is complicated compared to neural network. Neural networks are used as nonlinear estimating functions. To compute processing matrix, we have used a set of points to train the neural system. This collection of points are achieved by moving end effector of robot through different points which their coordinates in global reference system are known and their coordinates in image plane are computed by vision module in simulator software. The position of the end effector is recognized at any time by two cameras which are fixed with a certain distance from each other. The camera No.1 determines the target coordinates in a 2-D image plan. The depth of the picture also is computed by the second camera.

The used neural network, is a backpropagation perception kind network with 5 layers which in the first layer a 4 node entrance including picture plan coordination, layers 2, 3 and 4 take 20 node and finally output layer, 3 node including coordinates x, y and z in the earth-reference system.

4.2.1 Labeling process

The main aim of this process is allocated an exclusive index to any district in binary image through which all pixels having any quality except zero will be related with each other. Iterative labeling algorithm will analyze entrance on the image and each time a free and no related pixel is found, a new label is allocated to it. If any related, but having different labels district is seen, a series of iterative label will be published and both districts get new label, so that the least number of the labels are used. Iterative algorithm is very simple but too much slow, significantly when there are some U form object in the picture, labeling algorithm of equivalence table uses a table which registers the equivalences between the labels. Whenever the two related districts are seen with different labels, new entrance will be opened in equivalences schedule. The last process has been done by reallocation of the labels using research techniques in the graph. There will be an excessive difference, computing above mentioned approaches expenses costs. For instance, a picture in 700*356 sizes, which is labeled by iterative algorithm, takes about 50 seconds, while the necessary period for the algorithm using equivalences table is about 12 seconds.

Two processes are needed in equivalences table algorithm. The first process is the primary labeling and the second one, is to merge neighborhood labels by equivalences table. Because of this research being on-line, the 12-second period seems to be too much for the labeling task. So, another algorithm was created which could check for every non-free pixel and calculate to find if there. If the response is positive, the act of merging will be done in the same place and after merging, the task of labeling will be continued. The period spent for this algorithm will be decreased to two seconds (Korayem et al., 2005).

5. Simulation of controlling the robots

We have used feature based and position based visual servoing systems to control 3P and 6R robots. In this part of simulator software, position and orientation of end effector is controlled by using two fixed cameras for 6R robot and by one camera mounted on end effector of 3P robot. By estimating pose of end effector it will be moved to reach the target-object. In each motion of end effector the camera(s) will take pictures from target-object and end effector and its environment, then they will be recognized according to feature based visual servoing for 3P robot or position based visual servoing for 6R robot.

5.1 3P robot control

To simulate control of 3P robot, two cameras have been used. The user can observe the environment by switching between two cameras at any moment. The task of the processor camera which has been installed on the end effector is getting pictures from workspace and sending them to the vision system for information process. The second camera is the observer camera. The aim of installation of such camera is observing the whole space where the robot is moving and working and also its reaction will be observed by user. This camera is moveable and the user can change its position and orientation arbitrarily.

Picture frames taken by cameras are kept in a buffer in pixel format. Series of frames are also kept in buffer series. After segmentation, objects in the collection of the frames are labeled and separated. After threshold operations, segmentation, and labeling, the present objects in the *buf1* picture will be extracted and each single frame is conserved in different levels of the *obj* frames collection with its number. The target-object should be recognized among them after separation of the objects. Target will be recognized according to its features and properties. The approach used to control the end effector reaching to the target-object is the *feature-based visual control servoing*. Here, only one camera has been used for controlling the process.

In this controller process the displacements between target-object and the end effector is computed to correct delta X and delta Y, distance between end effector and target-object in x and y directions. Then necessary commands will be given to X and Y axis of the robot to move the end effector. After the above-mentioned displacements, another image will be taken from the workspace and the operations will be repeated. This process will be repeated until the robot's X and Y axis' error are getting the least amount possible. This algorithm tries to observe the target-object in the center of the image plane each moment and the distance errors between the end effector and target-object is computed according to the distance of center of object and the center of the image plane of camera mounted on the end effector. Then, delta Z is computed and rectified. This duty will be done just like the former case, but this axis error, is corrected based on the size of the object, observed in the image plane, the real size of object is known and it will be compared to the size observed in the image plane and the end effector will move till this difference reach to the least possible amount.

There are different kinds of objects in the robot workspace. Robot identifies the desired object which is a spherical shape by its feature in visual system and leads the end

effector to it. Visual system carries out this process in 34 steps. Figs 6, 7 show the robot in the first step and Figures 7 shows the robot in the last step after reaching to the target-object (34th step).
In this part of simulator software standard performance tests of robot also can be done without any limitation and the results can be observed quite naturally as well as controlling the end effector to find and reach the target-object automatically through the vision system.

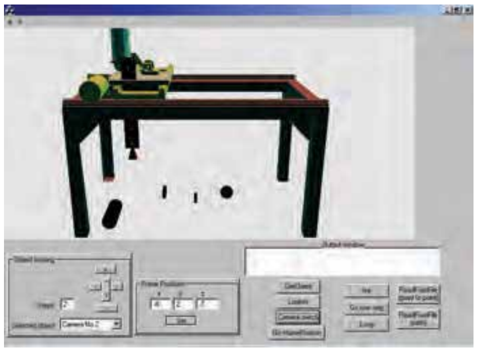


Fig. 6. Robot picture after the first step

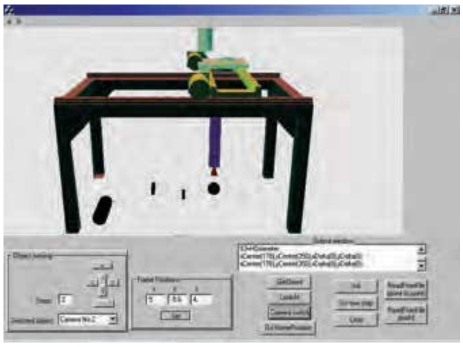


Fig. 7. Robot picture after reaching to the target

The robot can read and perform its motion in two approaches; the first process is point to point. In this approach, the rate of moving end effector in the space can be determined in the file that has been loaded before by the user in such a way of line after line and will be read and done through the program. The second approach is a continuous path and the user can determine paths such as circle (parameters: center and radius), rectangle (parameters: the length of the lines and coordination of corner), line (parameters: coordination of start and end points) for the robot motion.

5.2 6R robot control

To simulate control of 6R robot, the picture is taken in bitmap format by two fixed cameras which are mounted on the earth i.e., the cameras observe the robot we can call the system “out-hand" (the term “stand-alone" is generally used in the literature) then it is returned in form of array of pixels in a buffer. Image capturing is switched through two cameras it means that both cameras take photograph from robot and its surrounding and their images will be processed. After image processing objects in pictures are saved separately and their features are extracted and target-object and the end effector will be recognized among them according to their features and properties. Then position coordinates of target-object and end effector are estimated. After each motion of joints new picture is taken from end effector and this procedure is repeated until end effector reach to target-object.
With images from these two fixed cameras positions of objects are estimated in image plane coordinate, to transform these coordinates to global reference coordinate, we have used a neural network as described before. Computing 3D position of the end effector and target-object is possible by using images from these two fixed cameras. This approach used to control the end effector reaching to the target-object is the *position-based visual servoing control.* The advantage of this approach is that the servo control structure is independent from the target pose coordinates and to construct the pose of a target-object from two dimension image plane, two cameras are used. Image feature points in each of the two

images are to be matched and 3-D information of the coordinates of the target-object and its feature points are computed by this system.
In this algorithm, pictures taken by two cameras are saved in arrays of pixels and after threshold operations, segmentation, and labeling, the objects in the pictures will be extracted and each single frame is conserved separately with its number. The target-object and end effector should be recognized among them after separation of the objects according to their features and properties. Distance between end effector and target-object will be estimated, by using inverse kinematics equations of 6R robot, each joint angle will be computed then by revolution of joints end effector will approach to target. In each step cameras take image and the process will repeat until end effector reach to target and its distance to target-object gets the least possible amount. Control procedure of robot to reach to target-object is briefly shown in Figs. 8-11.

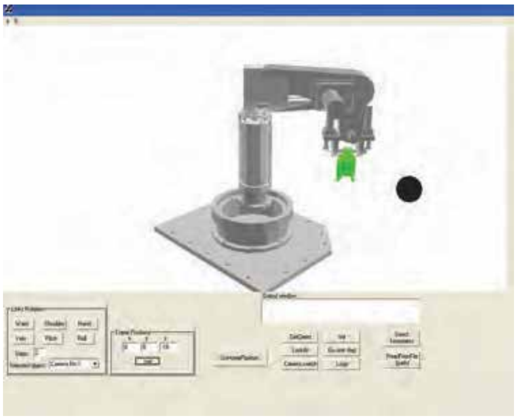


Fig. 8. 6R robot in home position (view camera1)

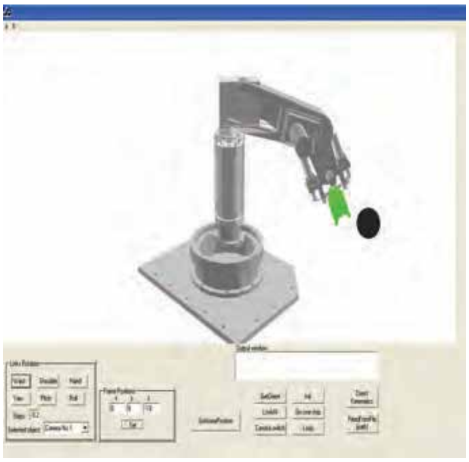


Fig. 9. Robot picture after some steps to reach to target.

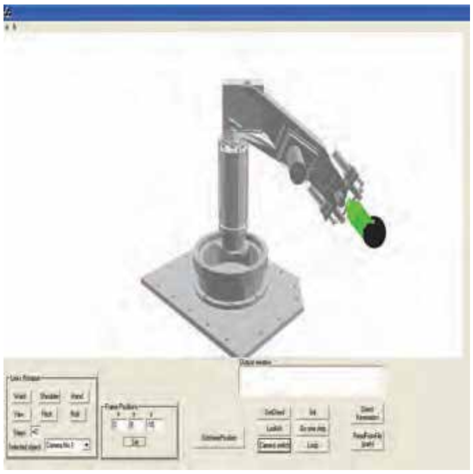


Fig. 10. Robot picture after reaching the end effector to target.

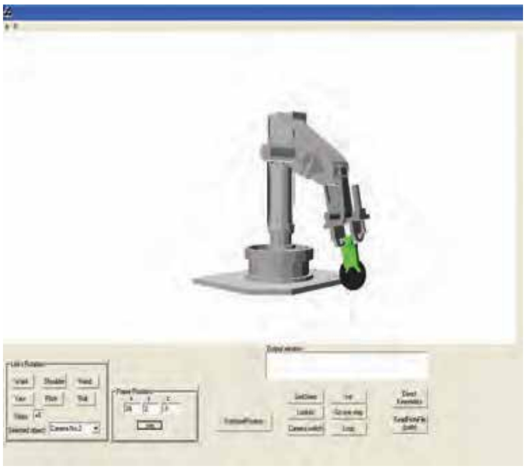


Fig. 11. The robot picture after reaching the end effector to target (view camera2).

6. Simulation of performance tests of robots

The designed robots should accomplish the given commands accurately and smoothly. This is possible in the case that the motion of the end effector of the robot is accurate enough relative to the target-object that is the point that the end effector of the manipulator has reached to. The accuracy of actual robot is under the effect of the following factors:

1) The accuracy of manufacturing mechanical parts of the robot.
2) The accuracy of assembling the constituting part of robot.
3) Accuracy during the robot operation that is influenced by external forces.
4) Electronics system accuracy and motors motion
5) The clearance existing in the system
6) Wear behaviours, that is change in accuracy of the robot in long duration
7) Change in accuracy of system after assembling the disassembled parts due to repair

So in simulator program these errors are figuratively inserted. Despite the recent international efforts by many of the standard committees, research and industrial labs, many of the robots users still sustain a loss, as a result of the lack of the standard, technical approaches and necessary determinations of the robot examinations. This matter is caused by complexity of the most robot designations and their vast limitations. In this research we try to do some of these approaches by using camera and visual system according to the standards such as ISO-9283, and ANSI-RIA.

The aim of these standards is providing technical information to help users to select the most convenient robot for their purposes. These standards define important principles based on the path, and then different appearances will be seen to evaluate them. These principles are: approximate accuracy of the path, absolute accuracy of the path repetition ability of the path rapidness specifications, and corner variable. Evaluation of the mentioned principles is one of the most convenient ways for evaluation of the whole activity based on industrial robots path. Also, the measurement of these principles brings the opportunity of comparing similar robots operations. To make tests more applicable, statistic analyzes according to ANSI-RIA or ISO9283 standards are performed.

6.1 Performance test of 3P robot

In second version of software the performance tests of the robot according to the international standards known as ISO 9283 and ANSI/RIA R15.05-2 have been accomplished. Two cameras with a certain distance form each other are looking at the end effector. One of these two cameras is fixed and zooms at the end effector. Position of end effector is determined in image plane and then is transferred to global coordinate by using neural network. For 3P robot performance tests which are implemented by camera and visual system are accomplished and performance parameters of robot are estimated. The end effector of 3P robot is moved on a special direction in different paths like circular and rectangular paths. The position of the end effector is recognized at any moment by two cameras which are fixed with a certain distance from each other. In this approach contrary to the former one, the camera is not on the end effector. The camera No.1 determines the target coordination in a 2-D image plan. The depth of the picture also is computed with the help of the second camera. Results of these tests are shown in Figs. 12, 13. (Korayem et al., 2005).

6.2 Performance test of 6R robot

In this research, by two fixed cameras on the ground we have simulated performance tests of 6R robot. Monitoring is possible with each of cameras. After image processing and recognition of the end effector and estimating its coordinate in image plane, by neural network this coordinate are transformed to global reference coordinate.

In this version of software, performance tests of robot including direct kinematics, inverse kinematics and motion of end effector in continues paths like circle, rectangle and line is possible. For point to point moving of end effector, each joint angle is determined and robot will move with joints rotation. In inverse kinematics test, desired position and orientation of end effector is determined in transformation matrix T. amount of joint angles which satisfy inverse equations will be found and wrist will be in desired pose. Two observer cameras take pictures and pose of end effector will be estimated to determine positioning error of robot.

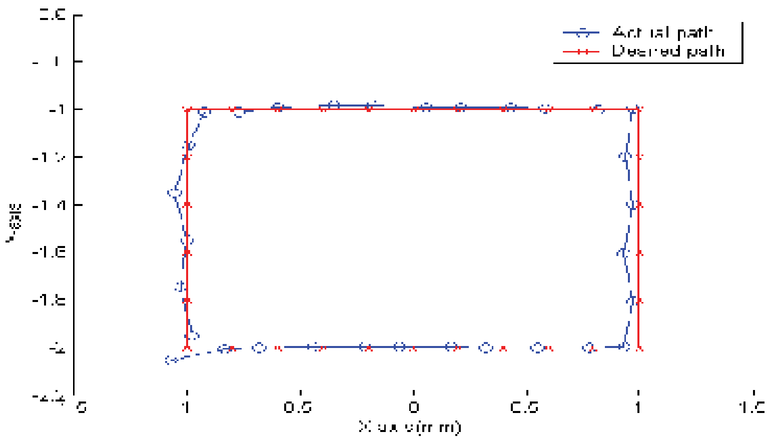


Fig. 12. Error investigation in variant rectangular path (by two cameras).

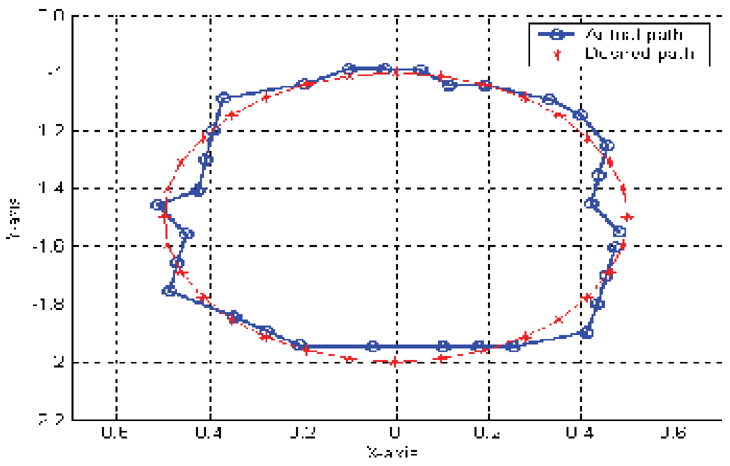


Fig. 13. Error investigation in circular path.

Then using ISO9283, ANSI-RIA standards, these errors will be analyzed and performance characters and accuracy of the robot will be determined. Results of these standard tests are used to compare different robots and their performance. In this research we try to do some of these tests by using camera and visual system according to the standards such as ISO-9283, and ANSI-RIA that belong to the robot specifically.

6.2.1 Performance test of 6R robot according to ISO9283 standard

The test operating approaches to specify robot parameters in this standard are divided into eight categories. In this research, two approaches among them which are done with camera and visual system are implemented to measure the path related parameters of 6R robot.

6.2.1.1 Direct kinematics test of 6R robot (point to point motion)

In this part of test position accuracy and repeatability of robot is determined. With rotation of joints end effector will move to desired pose. By taking pictures with two fixed cameras and trained neural network, we will have position of end effector in global reference frame. To determine pose error these positions and ideal amounts will be compared. Positioning error in directions x, y, z for 10 series of direct kinematics tests is shown in Fig. 14.

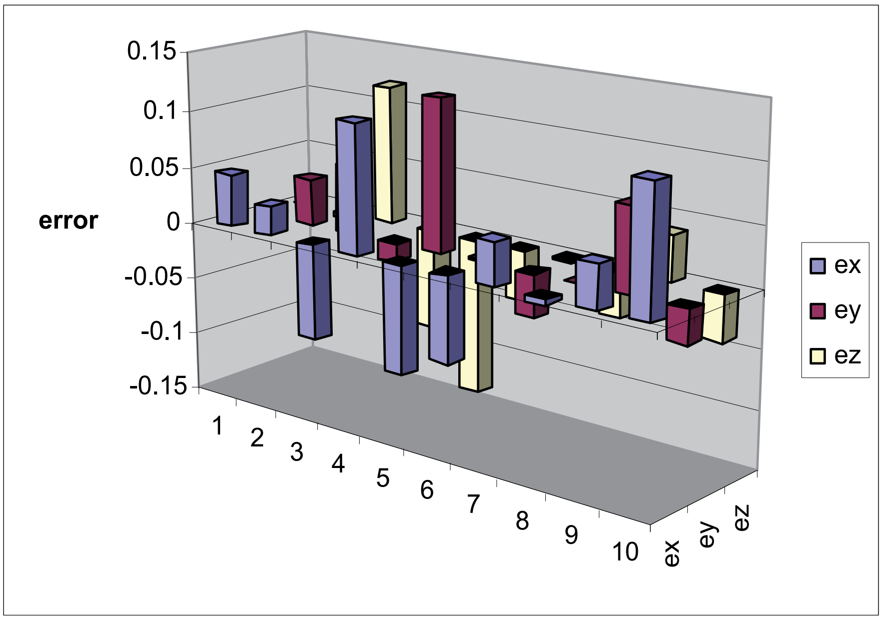


Fig. 14. The error schematics in x,y,z directions for direct kinematics tests.

6.2.1.2 Inverse kinematics test

In this stage, desired pose of end effector is given to robot to go there. By computing joint angles from inverse kinematics equations and rotation of joints, end effector will go there. By taking pictures with two fixed cameras and trained neural network, we will have position coordinates of end effector in global reference frame. By comparing the ideal amounts of pose and real one, the positioning error will be determined. Positioning error in directions x, y, z for 10 series of inverse kinematics tests is shown in Fig.15.

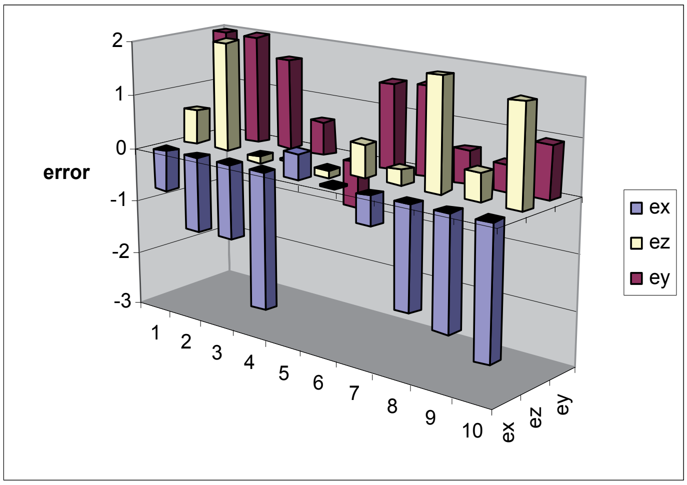


Fig. 15. The error schematics in x, y, z directions for inverse kinematics tests.

6.2.1.3 Continues path test

To determine accuracy of robot in traversing continues paths wrist of robot is guided along different paths. In simulator software, 3 standard paths are tested.

a) Direct line

To move end effector along a direct line its start and end points must be determined. Approach vector direction is normal to direction of line path i.e. wrist is always normal to its path. With pose of end effector and inverse kinematics equations of robot joint angles will be computed. Joints rotate and end effector will be positioned along its path. Coordinates of end effector in global reference frame is determined by taking pictures with two fixed cameras and trained neural network.

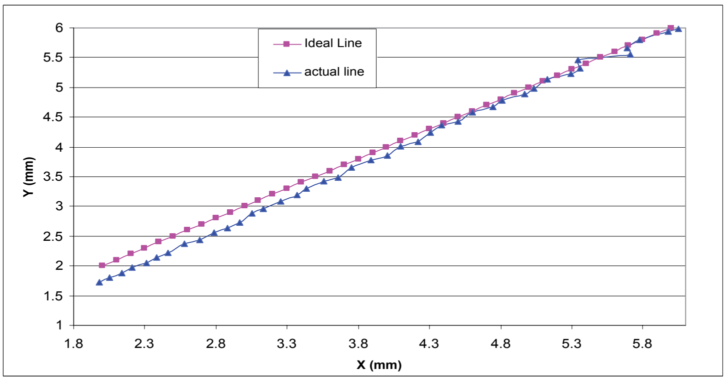


Fig. 16. The error investigated in line path.

The positioning error is determined by comparing the ideal pose and real one. Error of robot in traversing direct line path is shown in Fig.16.

b) Circular path

We investigate the accuracy, repeatability and error of robot on the circular continuous path traversing. Circle is in horizontal plane i.e. height of end effector is constant from earth level. Orientation of wrist is so that end effector is always in horizontal plane and normal to circular path and wrist slides along perimeter of circle. In this way sliding, approach and normal vectors are determined and inverse kinematics equations can be solved. During motion of wrist on the path, 32 images have been taken from end effector using two cameras. In this way, end effector coordinates in image plan will be collected. Using neural network, image plan coordinates of points will be transformed to the reference frame. The desired path and actual path traversed by robot is shown in Fig. 17.

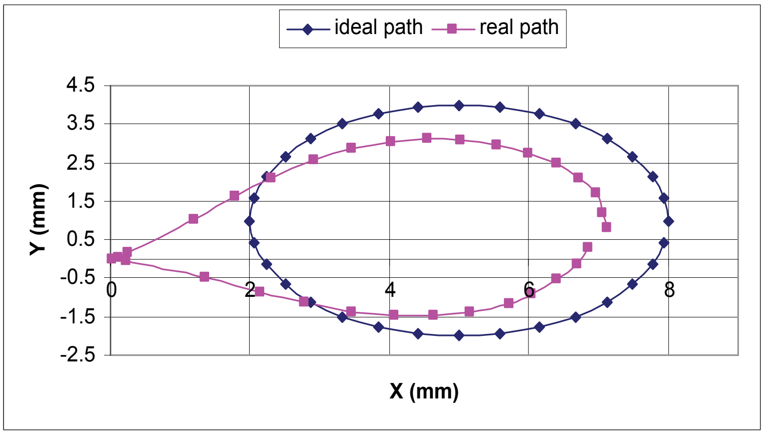


Fig. 17. The error investigated in circular path.

c) Rectangular path

Error of moving the wrist of robot along rectangular path is also considered. In order to do this, we define vertex coordinate, length and width of rectangle for the robot. Orientation of end effector is tangent to path. In this way transformation matrix of end effector is determined, then inverse kinematics equations are solved and end effector moves in this path, we take image from the end effector by the two cameras fixed on the earth. The desired path and actual path for rectangle r -3, -4, 10, 8 have been drawn in Fig. 18.

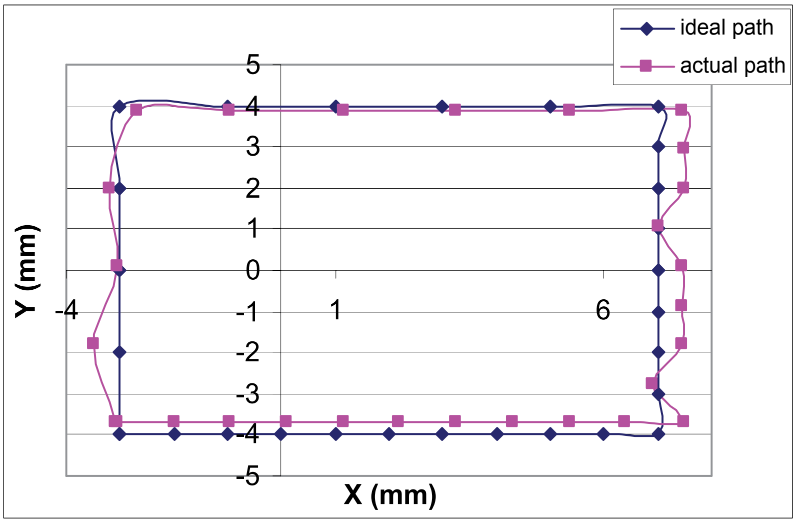


Fig. 18. The error investigated in rectangular path.

6.2.2 Robot performance test based on the American standard ANSI-RIA R15.05-2

The aim of this standard is providing technical information to help users to select the most convenient case for the robot. This standard defines important principles based on the path, and then different methods will be introduced to evaluate them. These principles are: approximate accuracy of the path, absolute accuracy of the path repetition ability of the path rapidness specifications, and corner variable. Evaluation of the mentioned principles is one of the most convenient ways for evaluation of the whole activity based on industrial robots path. Also, the measurement of these principles brings the opportunity of contrasting similar robots operations. To make tests more applicable, statistic analyzes according to ANSI-RIA standard are performed on the achieved out puts in the former sections (Korayem et al., 2005).

6.2.2.1 Circular and rectangular path

In this standard for rectangular paths at least 20 evaluating points and for circular path least 12 evaluating points must be used, that we have used 32 evaluating points for tests.

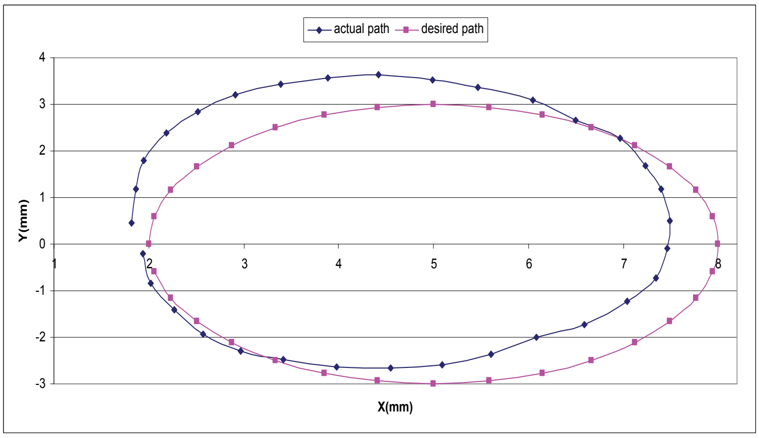


Fig. 19. The error investigated in circular path according to ANSI standard.

7. Error analysis of the 6R robot

Now we analyze results of previous tests according to different standards and we determine performance parameters and accuracy of 6R robot.

7.1 Error analysis according to ISO9283

In this standard some performance parameters of robot to position and path traversing such as pose accuracy and repeatability are determined. For direct and inverse kinematics test of 6R robot results are as follow. Bary center and mid point for test one is:

$$\bar{x} = -0.04, \bar{y} = 7.4, \bar{z} = 2.01$$

Pose accuracy for the robot which means error in positioning of end effector is computed as follow:

$$AP_P = \sqrt{(\bar{x} - x_c)^2 + (\bar{y} - y_c)^2 + (\bar{z} - z_c)^2}$$
$$AP_x = (\bar{x} - x_c), \quad AP_y = (\bar{y} - y_c), \quad AP_z = (\bar{z} - z_c) \tag{15}$$

Orientation accuracy is:

$$AP_a = (\bar{a} - a_c), \quad AP_b = (\bar{b} - b_c), \quad AP_c = (\bar{c} - c_c) \tag{16}$$

Pose repeatability is accuracy and error between attained pose and command pose after n repetition of test. Repetition n in these tests is 30. For a given pose repeatability is defined in (ISO9283, 1998).

Path accuracy, maximum drift of position and orientation AT$_p$ is maximum deviation between command positioning of path and Bray center Gi in n test cycles. Path accuracy AT is computed as follow:

$$AT_p = \max \sqrt{(x_i - x_{ci})^2 + (y_i - y_{ci})^2 + (z_i - z_{ci})^2} \qquad i = 1,2,....,m \tag{17}$$

Where x_{ij}, y_{ji}, z_{ij} are coordinate of intersection point between jth path and ith normal plane, and x_{ci}, y_{ci}, z_{ci} are coordinate of ith point of command path. Pose accuracy and path accuracy for the 6R robot in our tests are listed in Table 2.

	AP	RP	AT	RT
dir. kin	0.42	0.4	-	-
inv. Kin	0.98	1.05	-	-
line	-	-	0.65	0.85
circle	-	-	1.82	2.04
rectangle	-	-	0.78	0.92

Table 2. Pose accuracy & repeatability according to ISO9283 standard

7.2 Error analysis according to standard ANSI-RIA

Results of simulated tests in previous sections are analyzed with standard ANSI-RIA to compare with results of ISO9283.

$(v_{a_{ij}}, u_{a_{ij}})$ correspond to the coordinates of the Bary center path on evaluating plane. m is number of points in path and n is number of repetition. Their centers are $\bar{u}_{a_j}, \bar{v}_{a_j}$.

The mean reference path is used in the evaluation of the path repeatability FOM. This process involves the calculation of the deviation of D_{ij} between an evaluated point and its corresponding Bary center point (Korayem et al., 2005).

Cornering round off CR is defined as the minimum distance between the corner point and any point on the attained path. For each of the three corner points, the value for CR is calculated as follows:

$$CR = \min_{k=1}^{k} \sqrt{(X_e - X_{a_k})^2 + (Y_e - Y_{a_k})^2 + (Z_e - Z_{a_k})^2} \tag{18}$$

Where X_e, Y_e, Z_e are the position coordinates for each of the reference corner points and X_{ak}, Y_{ak}, Z_{ak}, are the coordinate of points along the attained path.

Cornering overshoot CO is defined as the largest deviation outside of the reference path after the robot has passed the corner. Its value is the maximum distance between the reference path and the attained path:

$$CO = \max_{k=1}^{k} \sqrt{(X_e - X_{a_k})^2 + (Y_e - Y_{a_k})^2 + (Z_e - Z_{a_k})^2} \tag{19}$$

Where X_{ak}, Y_{ak} and Z_{ak} are the position coordinates for discrete data points on the attained path. X_{rk} , Y_{rk} and Z_{rk} are the coordinates of the sample data points along reference path traversed by the robot.
To compute performance criteria of robot, end effector is guided along rectangular path based on standard platform (Korayem et al., 2005). The number of points for evaluation would be $m = 34$ and this will be repeated 10 times ($n = 10$). Two cameras, observing the end effector with at fixed distance in specified periods, take picture from end effector and its environment and its coordinates are achieved from image plan with position based visual system. To transform coordinates of wrist of robot to the reference frame as mentioned before, in this work we have used neural networks. Using neural networks we map coordinates from image plan into reference system, in order to have real distances.

Maximum repeatability is $PR = 0.375$ and average of repetition is $\overline{PR} = 0.264$ also CR corner deviation error and CO cornering over shoot for the tests simulated are listed in Table 3.

Reference	CR	CO
P1	0.125	3.251
P2	0.098	3.320
P3	0.178	3.78

Table 3. Repeatability & cornering overshoot according to ANSI standard

Where corner coordinates are:

$$P_1 = (2,3),\quad P_2 = (1,1),\quad P_3 = (-2,-3)$$

7.3 Error analysis with software MATLAB

Considering and analysis of direct and inverse kinematics tests data also has been done by software MATLAB. Error histograms in x and y direction for direct kinematics test is shown in Figs. 20, 21.

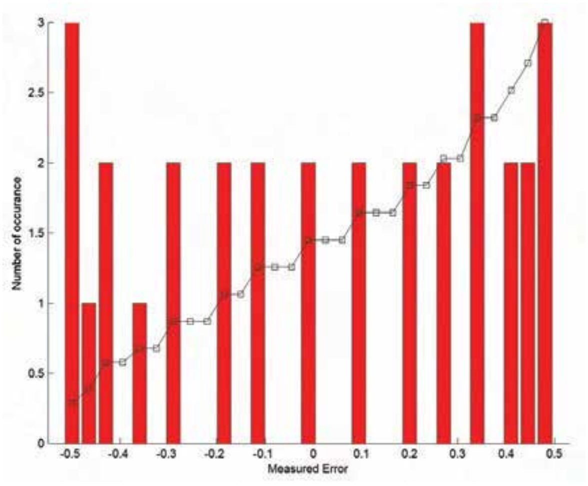


Fig. 20. The error histogram in x direction.

Mean value of error in x direction is 0.0376 with standard deviation equal to 0.353 and error skewness of -0.184.

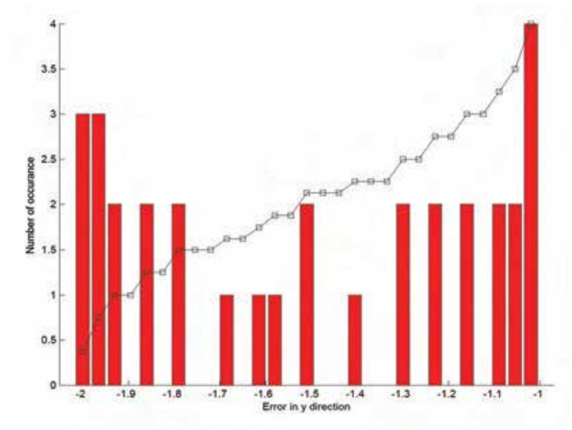


Fig. 21. The error histogram in y direction.

Mean value of error in y direction is -1.49 with standard deviation equal to 0.337 and error skewness of -0.0157.

For inverse kinematics tests, also error histograms in x and y directions are shown in Figs. 22, 23.

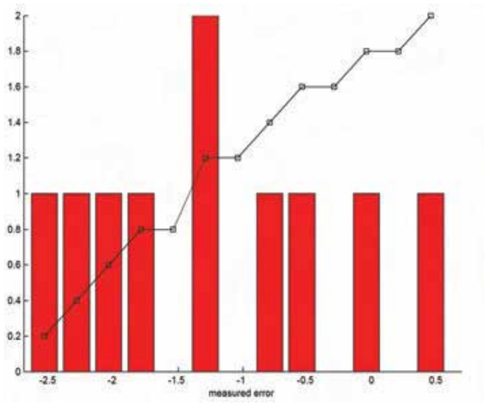


Fig. 22. The error histogram in x direction for inverse kinematics tests.

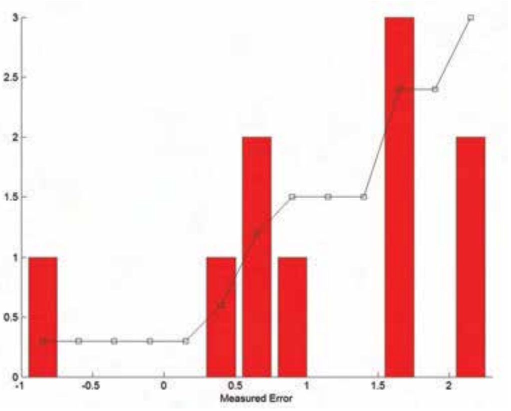


Fig. 23. The error histogram in y direction for inverse kinematics tests.

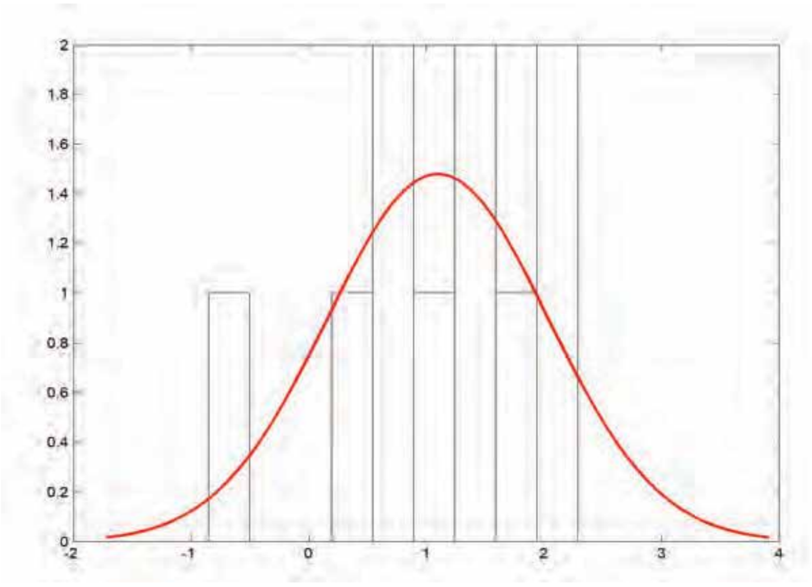


Fig. 24. Histogram with superimposed normal distribution for error in x direction.

To show normality of tests error we superimposed histogram of errors with corresponding normal plot which results in Figs. 24, 25.
To see whether or not the data are normally distributed we plot the graphs in Figs. 26, 27. It is seen that a fairly large portion of our data are close to the straight line, leading one to conclude that normal distribution is a reasonable approximation to these data.

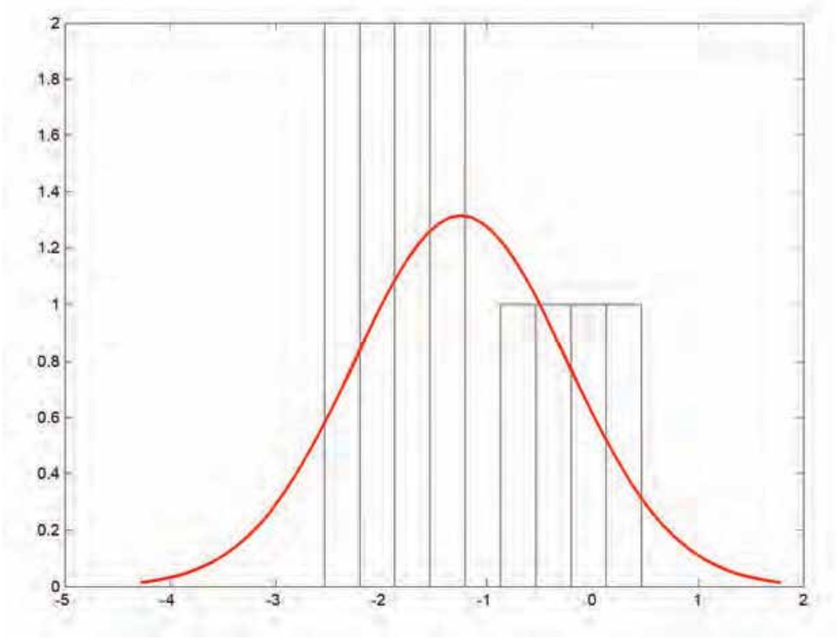


Fig. 25. Histogram with superimposed normal distribution for error in y direction.

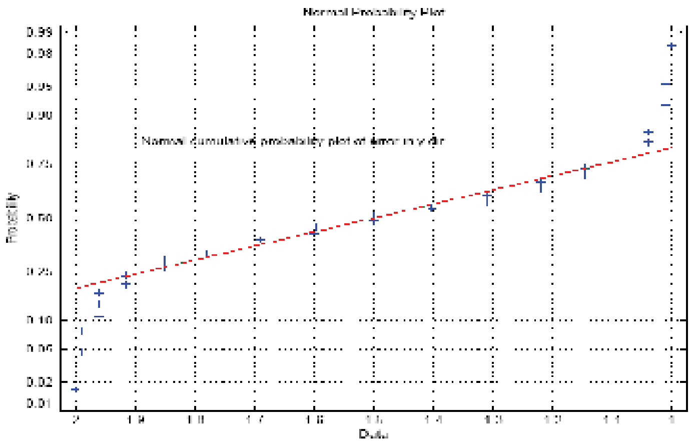


Fig. 26. Normal cumulative probability plot of error in x direction.

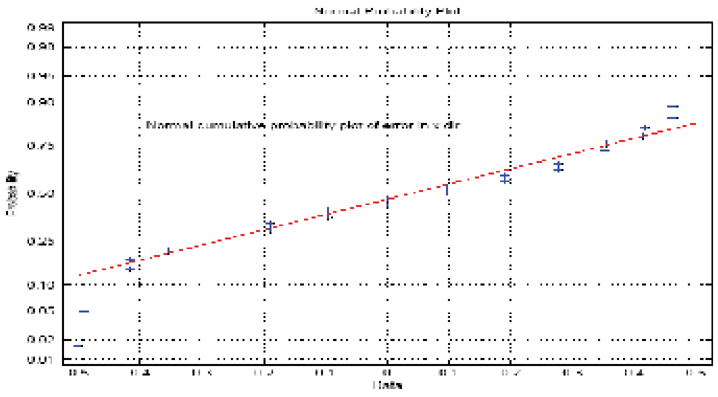


Fig. 27. Normal cumulative probability plot of error in y direction.

8. Conclusions

It has been shown how to use a vision system to control industrial 3P and 6R robots. It was observed that before using this vision system for controlling a robot, we need to know general information of the robot and the camera function and affects of light conditions on taken images. We can assume these results from the system as a better and more accurate control on robot around. In this system there is no need to know the first location of robot to calculate the required movement toward the goal, because taken images will help us to know the distance of the object to the end effector and this is one of the advantages of this system. Vision system can be used for path-related characteristics of robot. In this article, by applying vision system, the path-related parameters are found for industrial robot. The calculation of the path accuracy is simplified by finding the intersection of the attained path. Also simulator package for 6R robot, its different Sections and its capabilities are described.
Control and performance tests for 6R and 3P robot have been simulated by using position based and feature based visual system. In position based visual system it is not necessary to know the initial position of target-object and end effector due to capability of the target pose estimation in this method. Direct and inverse kinematics equations of the 6R robot have

been simulated and three-dimensional information of the target and end effector by using two cameras with acceptable accuracy have been implemented.
Errors have been analyzed by using different standards and also MATLAB to compute performance parameters of 6R robot such as accuracy, repeatability, and cornering overshoot. According to ANSI standard maximum repeatability is $PR = 0.375$ and average repeatability is $\overline{PR} = 0.264$ and according to standard ISO9283 we had average repeatability equal to $\overline{PR} = 0.42$, they are fairly close to each other. Also MATLAB showed error data were fairly normally distributed with standard deviation equal to 0.353 and skewness of -0.184.

9. References

American National Standard for Industrial Robots and Robot Systems Path-Related and Dynamic Performance Characteristics Evaluation. ANSI/RIA R15.05-2. 2002.

Gilbert, A, Giles, M, Flachs, G, Rogers, R, and Yee, H, (1983). A real time video tracking systems, *IEEE, Trans. Pattern Anal. Mech. Intell.* 2(1), pp. 47 – 56

Hashimoto, H, Kimoto, T, and Ebin, T (1991). Manipulator control with image based visual servoing, *In Proc. IEEE, Conf. robotics and automation*, pp. 2267 – 2272.

Haushangi, N, (1990). Control of robotic manipulator to grasp a moving target using vision, *IEEE Conf. robotics and automation,* pp. 604 – 609.

ISO9283, “Manipulating industrial robots performance criteria & related test methods”, 1998

Kelly, R, Shirkey, P and Spong, M, (2001). Fixed camera visual servo control for planar robots.

Korayem, M H, Khoshhal, K, and Aliakbarpour, H, (2005) Vision Based Robot Simulation and Experiment for Performance Tests of Robot“, *International J. of AMT*, Vol.25, No. 11-12, pp. 1218-1231.

Korayem, M H, Jaafari, N, Jamali, Y, Kiomarsi, M, (2005) “Design, Manufacture and Experimental Analysis of 3R Robotic Manipulator”, Paper in TICME.

Korayem, M H, Jamali, Y, Jaafari, N, Sohrabi, A, Kiomarsi, M, Asadi, M and Reazaee, S (2005),Design & Manufacturing a Robot Wrist: Performance Analysis, Paper in TICME.

Korayem, M H, Shiehbeiki, N, and Khanali, T (2006). Design, Manufacturing and Experimental Tests of Prismatic Robot for Assembly Line“, *International J. of AMT*, Vol.29, No. 3-4, pp. 379-388.

Peter I. Corke, (1996), *Visual control of robotics: high-performance visual servoing*, John Wiley.

Webber, T and Hollis, R, (1988).A vision based correlation to activity damp vibrations of a coarse fine manipulator, Watson research center.

Sheet Incremental Forming: Advantages of Robotised Cells vs. CNC Machines

Massimo Callegari[1], Dario Amodio[1], Elisabetta Ceretti[2], Claudio Giardini[3]
[1]*Dipartimento di Meccanica, Università Politecnica delle Marche, Ancona, Italy*
[2]*Dipartimento di Ingegneria Meccanica, Università degli Studi di Brescia, Italy*
[3]*Dipartimento di Progettazione e Tecnologie, Università degli Studi di Bergamo, Italy*

1. Introduction

In recent years the traditional sheet metal forming processes, suitable for high volume batches, do not correctly meet new market requirements characterised by high flexibility, reduced time-to-market, low cost for small batch production, etc. Moreover, they are not suitable for producing low cost prototypes and pre-series components. Thus new sheet metal forming techniques are very often required and pursued by manufacturing industries and have been intensively undertaken by scientific research groups (Siegert *et al.*, 1997; Amino *et al.*, 2000; Shima, 2001; Kochan, 2001; Shim *et al.*, 2001; Filice *et al.*, 2002; Iseki & Naganawa, 2002; Kim *et al.*, 2003; Yoon *et al.*, 2003; Ceretti *et al.*, 2002, 2004; McLoughlin *et al.*, 2003; Allwood *et al.*, 2005; Lamminen, 2005; Meier *et al.*, 2005). Among the new innovative technologies, the sheet Incremental Forming (IF) can be successfully used for small pre-series batches or prototypes. IF is a process where common and simple tools mounted on CNC machines, instead of complex die sets, are used to deform locally a workpiece. In recent years many studies have been done on IF and many are still in progress with the aim of finding both the most affecting process parameters and the suitable machines and working centres to run experiments and production (Park & Kim, 2002, 2003; Jeswiet *et al.*, 2005a, 2005b; Duflou *et al.*, 2005a, 2005b; Hirt *et al.*, 2005; He *et al.*, 2005a, 2005b; Ambrogio *et al.*, 2005; Bambach *et al.*, 2005).
Unlike the standard metal forming process, fast production changes are possible thanks to the very simple IF machine configuration. Even if the time required for making one product is much longer than in the traditional press forming, the IF advantages are gained on tool design and production in prototyping phase. IF could be also successfully applied in completion flexible work cells, for example after hydroforming operations for slots or small parts finishing. Furthermore, instead of using general purpose CNC machines, the modern incremental sheet processes can be directly performed on robotised cells. This will enhance the advantages in flexibility and production time reduction since a robotised cell equipped with the proper tools can produce the part and, on the same fixture, realise the completion operations such as flanging, trimming and so on.
To form the sheet into the desired shape an *ad hoc* tool, mounted on the machine spindle or on a robot gripper, is moved according to the given tool path. Several IF strategies have been developed which mainly differ for equipment and forming procedure. In particular,

the process can be divided into:

- Single Point Incremental Forming (SPIF), where the sheet is deformed by a single tool (no support is present).
- Two Points Incremental Forming (TPIF), sheet deformation is ensured by a tool and a local support (a kind of partial die).
- Full Die Incremental Forming, where the tool deforms the sheet against a die; this die can be realized with cheap materials such as wood, resin or low cost steel; the use of a die ensures a better and more precise shape of the final piece.

The above described strategies can be used on both CNC machines and industrial robots.

This paper presents an overview of the most recent researches carried out by the Universities of Bergamo and Brescia and by the Polytechnic University of Marche about the innovative IF process, describing the used working machines, the equipment devices and the process experimental and simulative optimisation applied to Full Die IF.

More in details, the results of the use of an industrial robot for the IF with Full Die will be analysed in terms of a potential increase of technological opportunities (e.g. the use of 5 axes manufacturing) and of process versatility (e.g. the capability of performing extra flanging or completing works) and compared with traditional 3 axes machining centre results.

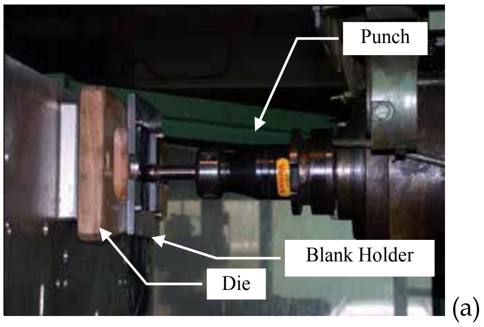
2. Machines and equipment

2.1 CNC milling machine

As reported in the previous paragraph, IF is characterised by different strategies. The one utilised in the present research refers to Full Die Incremental Forming. In fact, since the final goal of this research is to identify working rules for pre-series production, this methodology should allow the achievement of the best results in terms of geometrical and surface tolerances. It must be underlined that it is not necessary to use an expensive material for the die (such as particular resistant steel) since the working loads and deformation areas are reduced, but polymeric materials, thermosetting resins, hardwood and any other material satisfying strength, stiffness and surface finishing requirements can be used so allowing an overall cost reduction.

IF with Full Die is based on the local deformation of a sheet on a die obtained by means of a punch moved by a CNC machine over a die which can have a negative (Fig. 1a) or positive shape (Fig. 1b): in this last case a movable blank holder is required. The main components of the equipment are:

1. a punch mounted on the spindle of the CNC machine;
2. the die, which reproduces the part geometry;
3. the blank holder to keep the sheet in the right position during the process (fixed or movable).



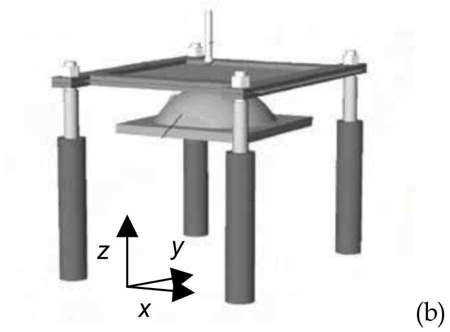


Fig. 1. The set up components.

The die is mounted on the table of the CNC machine and the moving punch deforms the sheet up to the die. The punch path must be optimized by means of an accurate study of the final geometry, in order to obtain a good surface finishing. When the die has a positive geometry the blank holder is moved by four hydraulic actuators to keep the sheet steady in the working position (Fig. 1b) otherwise is fixed (Fig. 1a).
The punch has a cylindrical body with a spherical head. The sphere dimension is an important process variable. In fact, large headed tools assure better material flow and production time reduction, while small sphere dimensions are needed to accomplish the dimensional features of the part (the tool radius must be equal or lower than the minimum radius of curvature of part's surface).
To generate the punch trajectory CAD, CAM and CAE techniques were used. Since the tool movement strategy is very important for obtaining safe components, several tool paths were investigated in previous studies (Ceretti *et al.*, 2003; Giardini *et al.*, 2004a, 2004b, 2004c, 2005a, 2005b) and the best one must be identified by taking into account the positive or negative shape of the die, the shape of the part, the tool dimensions and the sheet material.

2.2 Robotized cell

Conventional industrial robots do not have the stiffness nor the accuracy that are generally required for the incremental forming operations (Lamminen, 2005) but the rather new parallel kinematics machines can be designed so as to show good features from this point of view: for the present research the COMAU Tricept HP1 robot has been used, a six axes machine with hybrid structure (a serial wrist is mounted on top of a parallel shoulder), shown in Fig. 2. It is able to apply a maximum thrust of *15 kN* over a work envelope of *2000 mm x 600 mm*, with a repeatability better than *0.03 mm*. The drawbacks of the machine mainly lie in the great anisotropy of its workspace, that is also characterised by a complex shape and a limited dexterity; therefore a simulation tool has been implemented in order to be able to analyse off-line machine kinematics and assess beforehand the feasibility of a certain task.

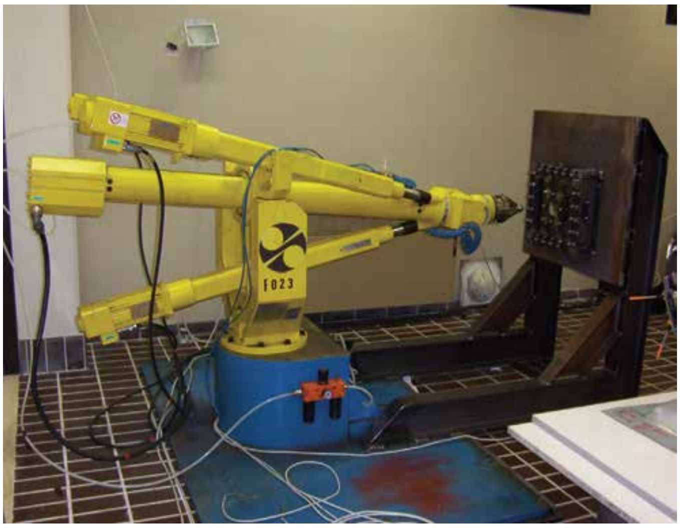


Fig. 2. The hybrid robot Tricept HP1.

A pneumatic gripper holding a punch with a spherical tip is attached to the robot flange. Various punches with different radii are used but in any case the small dimensions of the ball tips allow to obtain the required high local pressures by the application of forces considerably smaller than in conventional stamping. The die and the blank are clamped together by means of several clips, whose design and setting are very important for the quality of the operation: in fact a rigid constraint at frame boundaries can be desired in some cases as well as the possibility to control the material flow towards the die in other situations (Ceretti *et al.*, 2003). The die with the supporting frame and the robot are mounted on the same (thick) metal sheet: an automatic tool changing system was available beside the robot but it has not been used in the present experimentation.

3. Analysis of the whole production process

The design of the production process begins from the definition part's geometry, that is usually made by importing the CAD model of either the part or the die. Then the tool path is generated by a CAM program. The part program used to machine the die is useless and a specific path must be generated instead, to take into account that a plastic deformation is being induced and to satisfy the needs of realizing good quality parts. The CL file is then imported in FEM packages to evaluate the maximum forces that are required to complete the forming process (ANSYS®) and to simulate the operation (PAMSTAMP®) in order to forecast the final shape and the thickness distribution of the part. This is a fundamental task when studying the part feasibility.
A kineto-static model of the robot has been implemented in Matlab by the Authors therefore it is now possible to verify that tool path is entirely inside the workspace of the manipulator and that the forces required at tool tip can be actually developed by the motors at the actuated joints. If the check is not passed, it is necessary to change the punch (e.g. a smaller size one) or the relative position between the die and the robot (e.g. by means of a rotating table) or the kind of tool path (e.g. the tool can be differently oriented in such a way as to load it mainly with axial forces).
At this phase the availability of a commercial multibody simulator can be useful, e.g. to verify that the joints are not running out of their admissible stroke or that no impact with other machinery in the cell occurs. The IGRIP® package, that has been used to this aim, also allowed to generate robot's task in controller's own programming language, i.e. PDL in this case; then the preliminary version of the program has been directly uploaded to Tricept controller for the final tuning. The off-line programming is also enabled by an IGRIP® calibration utility, that has been used for the initial setting of cell's CAD model according to the real data measured in the laboratory.
Some modules of the recalled procedure have been previously outlined in (Callegari *et al.*, 2006) and will be explained in more details in the following sections.

3.1 CAD/CAM module

The solid model of the die is based on the geometry of the part and corresponds either to convex or to concave die shapes. The tool path can be different in the two cases, since for positive die the tool usually deforms the blank starting from the inner part and moving towards the boundary, while the opposite approach is used for negative forming.
A CAD package has been used for modelling the part and the die and for the related CAM

processing: a fixed slicing of the part was used, with a step depth (or pitch) that has been varied from *0.5 mm* to *1.0 mm*. Both 3 axes (for both CNC machine and robot) and 5 axes (for robot) paths have been generated, see Fig. 3 and 4.

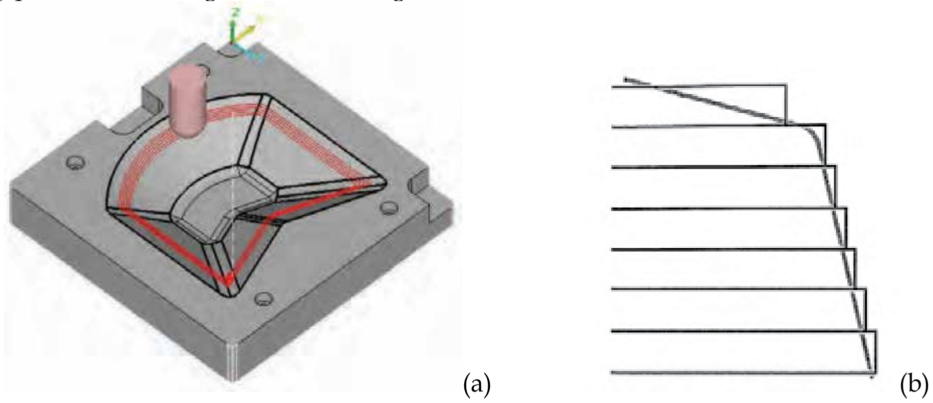


Fig. 3. Negative forming with concave die: 3 axes path (a) and fixed step slicing (b).

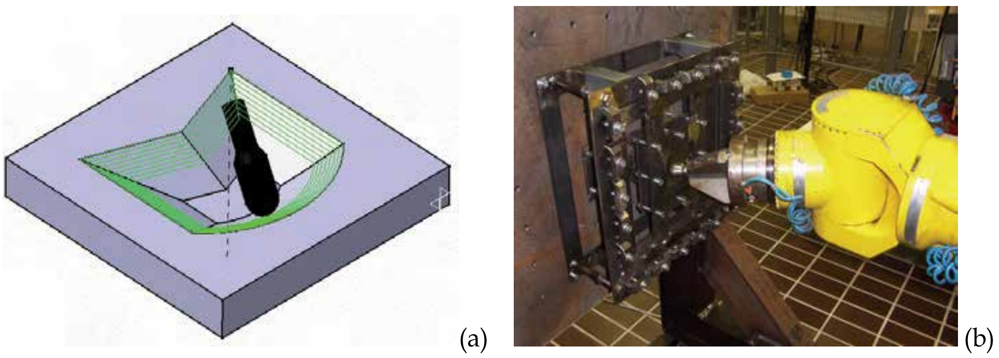


Fig. 4. Negative forming with concave die: 5 axes path (a) and robot wrist (b).

When the die is concave, different punch movement strategies can be used. In particular, one strategy starts the process from the part boundary and progressively deforms the sheet through trajectories drawn at constant *Z* levels (direct negative forming); the other strategy is characterised by a straight tool movement inside the blank till the maximum depth of the die is reached and then a spiral movement towards the part boundaries starts (inverse negative forming), see Fig. 5. This last approach can be used only with limited die depths and assures a better material flow from the boundary so guaranteeing a lower thinning of the part. Unfortunately, this strategy could not be applied with the robotised cell because the robot was not able to develop the high lateral thrusts that were required. To overcome this limit a progressive solution was tested: in this case the punch is moved as in inverse negative forming but it moves downwards up to the bottom of the die following progressive spiral paths with different geometrical dimensions whose effects were investigated.

The velocity of the tool together with the path variables (spiral width, pitch and so on) have a great relevance for the quality of the part and must be chosen according to the other parameters of the process, like sheet material and thickness, depth of deformation, etc.

If the surface finishing of the part is an important aspect of the process, the path conducted as a pocketing with constant step depth needs to be modified. In fact, this step down path from the top to the bottom of the pocket, in which the tool follows a series of consecutive contours with fixed step depth, is the simplest one but presents two disadvantages. In particular the sheet is marked at the transition point between consecutive layers and the quality of flat or near to flat surfaces is poor when high step depth or pitch (more than *0.5 mm*) are used.

In this case to conduct pocketing with constant "scallop height" (Sc) can give better results. This is a step down path from the top to the bottom of the pocket, in which the tool follows a series of consecutive contours with variable step depth (the maximum of which must be furnished) in order to keep constant the value of the scallop height (Fig. 5d). This kind of path reduces the disadvantages of the first type and, in particular, the flat surfaces show a better quality.

Step depth, scallop height and type of tool path influences were studied during the experimental tests, changing their values.

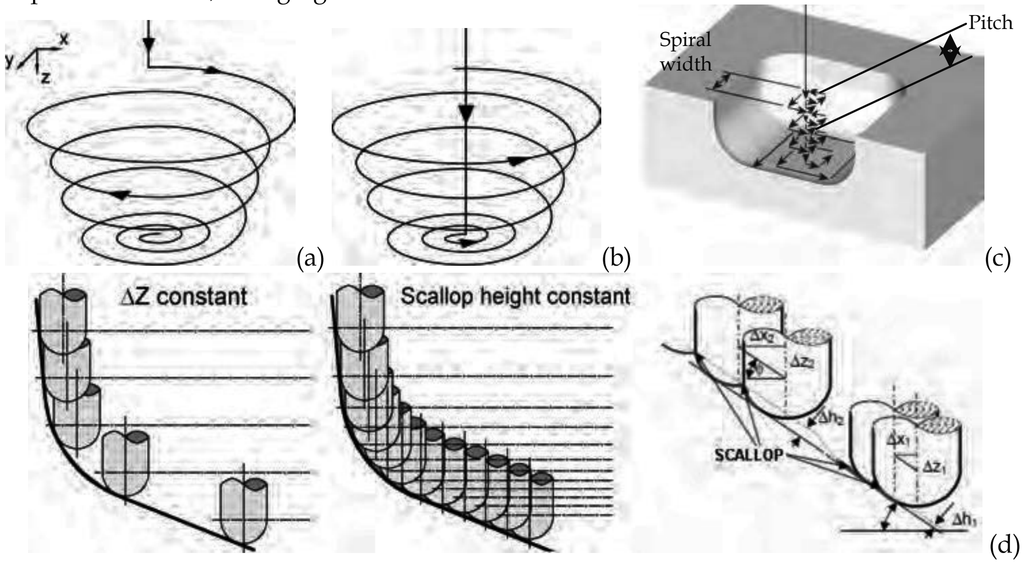


Fig. 5. Direct (a) and inverse (b) negative forming with progressive downward movement (c) at fixed step depth or fixed scallop height (d).

3.2 Analysis of the forming process

Analytical model

A rough estimation of the state of stress of the blank under deformation can be obtained by analytical models, while more detailed FEM simulations must be performed to go inside the matter and assess the influence of the main process parameters.

For sake of simplicity, it has been closely followed the approach of Iseki (2001), which refers to a planar state of deformation. This approximation can be rather well satisfied in case of spherical rollers, with good lubrication and relatively small curvature paths. Referring to the geometrical setting and notation of Fig. 6, representing a negative forming, the z axis is taken along the direction of the incremental depth steps, and the y axis along the current motion direction.

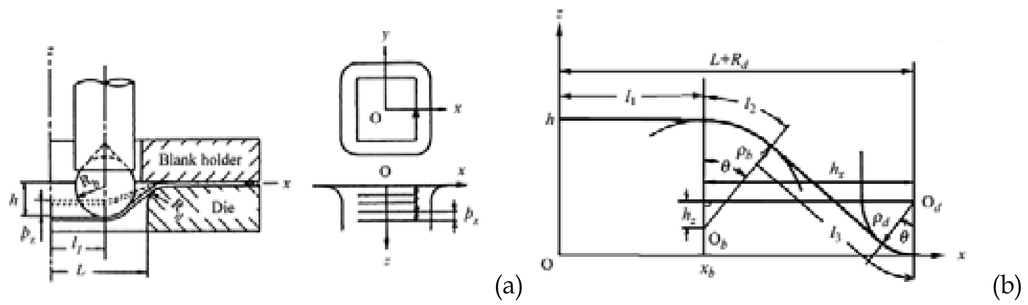


Fig. 6. Sketch of the negative incremental forming setting (a) and planar model of the arising state of deformation (b).

The contact angle θ between tool and blank is:

$$\theta = \arcsin\left[\frac{\rho_b + \rho_d}{\sqrt{h_x^2 + h_z^2}}\right] - \arctan\left(\frac{h_z}{h_x}\right) \tag{1}$$

It is noted that half of the initial thickness of the metal sheet, t_0, must be added to both curvature radiuses of tool and die, ρ_b and ρ_d respectively. The hypothesis of plain strain state leads to:

$$\varepsilon_x = -\varepsilon_z = \ln\left[\frac{\ell_2}{L + R_d - \ell_1 - \ell_3}\right] = \left[\frac{\ell_2}{h_x - \ell_3}\right] \tag{2}$$

where : $\ell_2 = \rho_b\theta$ and $\ell_3 = h_x \cos\theta - h_z \sin\theta + \rho_d\theta$.

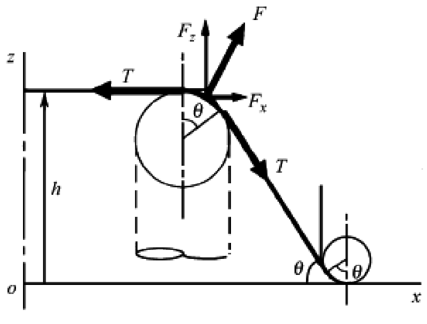


Fig. 7. Equilibrium of metal sheet during negative incremental forming.

With reference to Fig. 7, the total effort causing material deformation, *F*, can be well approximated being equal and opposite to the resultant of the 2 traction forces *T* at the borderline of contact surfaces. The tensile force *T* on the metal sheet is:

$$T = 2R_b t\sigma \tag{3}$$

If actual sheet thickness *t* is expressed as a function of the initial thickness t_0 and of the deformation ε, it can be obtained:

$$T = 2R_b\left(\frac{2}{\sqrt{3}}\right)^{n+1} K t_0 e^{(-\varepsilon_x)} \varepsilon_x^n \tag{4}$$

where the material hardening law $\sigma = k\varepsilon^n$ has been considered. The two Cartesian components of interaction force at the end effector can now be simply expressed as:

$$\begin{cases} F_x = T(1-\cos\theta) \\ F_z = T\sin\theta \end{cases} \tag{5}$$

The simple inspection of (1-5) shows that, as expected, the lateral force to be developed at the tool increases with the contact angles θ (and therefore with the depth h) and decreases with the distance between the local centres of curvature of tool and die.
The results of the current model have been compared with the corresponding FEM simulations and experimental trials, whenever available: a good agreement has been proved in many situations, with results generally over-estimated and therefore conservative for the sake of process safety. On the other hand, the model is not able to emulate closely the actual thinning of the sheet (and therefore forecast the possible occurrence of tears or wrinkles) or the paths with small curvature radii, where clear three-axial states of deformation establish. If these cases must be investigated, a FEM simulation is unavoidable, while the analytical model still preserves its validity for a first assessment of the forces occurring at tool tip, that is an important input information for the kineto-static feasibility check that will be explained in detail in next section.

Numerical process simulation

In order to verify the process ability in realizing the desired shape, that is the process feasibility, it is important to conduct FEM analyses of the sheet deformation. Once the FEM model has been validated by comparing the obtained results with the experimental ones, it is possible to use the model itself to forecast the process parameters influences on the final part shape and on sheet thinning which are relevant aspects in part quality evaluation. Moreover, different punch movement strategies can also be analysed and the best one can be identified prior to the experimental phase.
The simulations were conducted using both implicit (DEFORM®) and explicit (PAMSTAMP®) codes. This latter has the advantage of reducing the computational time with respect to the implicit code, in addition the explicit code results are closer than the implicit to the experimental thickness values. As a consequence, in the following of this paper, only the results of the explicit FE code will be presented. In both cases, the FE programs move the punch using as input the CL files obtained from the CAM module. The model components are the sheet (modelled as elastic-plastic element), the die, the punch and the blank-holder (considered as rigid bodies), see Fig. 8. The friction was modelled according to the Coulomb law.
A first set of simulations was run to identify the most suitable values for friction coefficient and blank-holder force by comparing the experimental thickness distribution along the piece section. Once the model was validated it was possible to use the model itself to forecast the feasibility of a given part shape and to evaluate the part quality, in correspondence of variations of tool path or other process parameters (e.g. tool diameter, feed rate).
To allow the simulation in 3D DEFORM® environment, a suitable program able to move the punch (treated as the upper die) along a defined trajectory as obtained from the CL file has been developed and implemented in a user subroutine. This module allows also to restart the simulation from an intermediate point of the trajectory as required when a remeshing occurs. Moreover, another module has been developed in order to allow the thickness calculation and representation using proper variables specifically defined in the pre-processing phase.

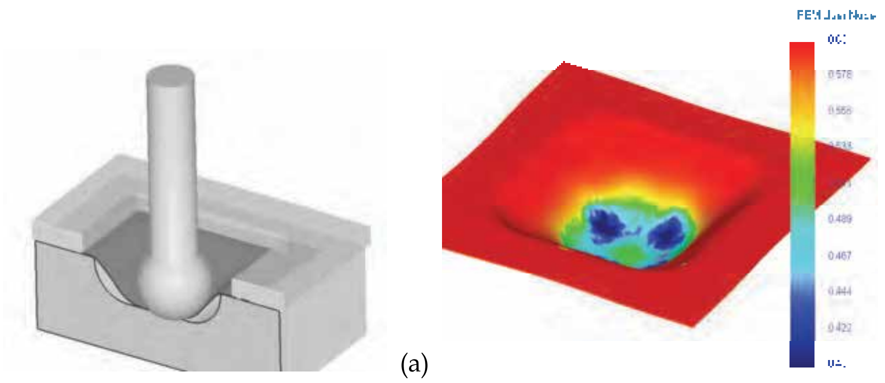


Fig. 8. FEM model objects: sheet, punch, die and blank-holder (a); the thickness representation (b).

3.3 Feasibility kineto-static checks

The only kind of robots presently able to develop the required thrusts at the tool are parallel kinematics machines, that are also characterised by high stiffness and good accuracy: unfortunately, the closed-loop architecture generally yields a workspace of complex shape and characterised by high anisotropy, which complicates the planning of effective operations (Merlet, 2005). On the other hand, the information provided by robot manufacturers are often too poor to enable an off-line assessment of the feasibility of the current incremental forming operation. Therefore more detailed simulations must be performed beforehand to check that all the generated 5-dimensional paths are included inside the 6-dimensional workspace of the manipulator and that, at every position, the motors torques required by the task can be actually developed. Since the velocities of the robot are usually quite low, a kineto-static model of the manipulator is sufficient to assess the kinematic feasibility of the required tool positions and to correlate workspace efforts to joint space torques. The Tricept HP1 used in the tests has a maximum thrust of *15 kN* in the axial direction when the tool is aligned along the same *z* axis but its static performances dramatically downgrade as the structure leaves such favourable configuration (see Fig. 9 for a qualitative representation). Therefore a complete kineto-static model of the robot has been developed and implemented in the Matlab programming language, as explained with details in (Callegari *et al.*, 2005) and summarised in the following sections.

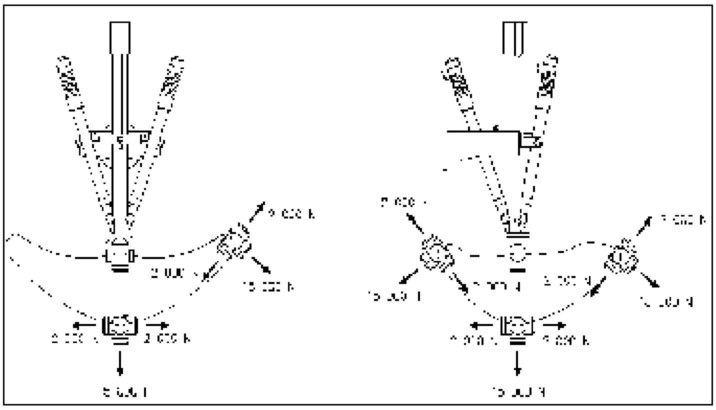


Fig. 9. Static performances of the robot Tricept HP1 (Comau, 1995 and 2001).

Robot kinematic model

In order to develop the kineto-static model of the Tricept HP1, the conventional steps of direct and inverse kinematic analysis had to be performed. It is well known that the resolution of the position problem is always the most challenging phase of the work, usually ending up with a complex non-linear system of algebraic equations. In this case the approach suggested by Siciliano (1999) for the resolution of the (3 dof) parallel shoulder has been followed, by adapting it to the case of a 6 axes hybrid machine: due to the difficulty of the problem, direct kinematics has been solved numerically, while inverse kinematics admits closed-form solutions. Velocity kinematics, on the other hand, can be expressed in the usual linear form:

$$\begin{bmatrix} v_x \\ v_y \\ v_z \\ \hdashline \omega_x \\ \omega_y \\ \omega_z \end{bmatrix} = J(\theta_1,\cdots,\theta_6)\cdot\begin{bmatrix} \dot\theta_1 \\ \dot\theta_2 \\ \dot\theta_3 \\ \hdashline \dot\theta_4 \\ \dot\theta_5 \\ \dot\theta_6 \end{bmatrix} \tag{6}$$

where in the left hand side all the terms of tool linear and angular velocities are collected in a single vector, while $\dot\theta_i$, $i=1,\ldots,6$ are motors speeds and J is a proper Jacobian matrix. It is noted that the Jacobian incorporates all the terms of the velocity mapping, up to ball screws pitches or gear reducers ratios.

Manipulability ellipsoids

Let us consider the set of joint velocities $\dot{\boldsymbol\theta}$ of constant unit norm:

$$\dot{\boldsymbol\theta}^T\cdot\dot{\boldsymbol\theta} = 1 \tag{7}$$

It is desired to describe the Cartesian space velocities of the end-effector that can be generated by such velocities at a given configuration in space. The image of the six-dimensional sphere (7) in the operational space of tool linear and angular velocities is called (*velocity*) *manipulability ellipsoid* and is given by the quadratic form:

$$\begin{bmatrix}\mathbf{v}^T & \boldsymbol\omega^T\end{bmatrix}^T\cdot\Gamma_v\cdot\begin{bmatrix}\mathbf{v}\\ \boldsymbol\omega\end{bmatrix} = 1 \tag{8}$$

where the *6x6* square matrix Γ_v is defined by:

$$\Gamma_v(\boldsymbol\theta) = J^{-T}(\boldsymbol\theta)\cdot J^{-1}(\boldsymbol\theta) \tag{9}$$

Cartesian space velocities are high along the direction of the major axis of (9), while only low velocities are yielded along its minor axis: therefore an isotropic (kinematic) behaviour of the manipulator is manifested when the ellipsoid becomes a sphere. Just to visualise the matter, in case of a planar robot with two equal motors, the form (8-9) would represent an ellipse whose major and minor axes (the eigenvectors of Γ_v) indicate the directions along which the tool can be moved with the maximum and minimum velocities respectively (that are given by the square roots of the eigenvalues of Γ_v).

Due to the well-known duality between kinematics and statics, for every pose of the manipulator inside its workspace (position and attitude of the tool) the 6 joint torques $\boldsymbol\tau$ balancing the external wrench of forces **F** and moments **M** applied at tool tip are given by:

$$\boldsymbol{\tau} = J^T \cdot \begin{bmatrix} \mathbf{F} \\ \mathbf{M} \end{bmatrix} \tag{10}$$

Following the same approach previously outlined for the characterisation of robot velocities, the ellipsoid representing all the Cartesian forces and moments that correspond to unit motors torques, $\boldsymbol{\tau}^T \cdot \boldsymbol{\tau}=1$, can be derived. In the case of the present application the external moments at tool tip are null, therefore only the three-dimensional ellipsoid of end-effector forces is meaningful:

$$\mathbf{F}^T \cdot \Gamma_f \cdot \mathbf{F} = 1 \tag{11}$$

where Γ_f is the $3x3$ upper left sub-matrix of $J\,J^T$.

The knowledge of matrix Γ_f clearly characterises the static performances of the manipulator and also represents an effective means to visualise them; it is noted that Γ_f depends on all the 6 coordinates of robot's pose. As an example, Fig. 10 shows in dark patterns the *force manipulability ellipsoids* at various points of Tricept workspace, when the tool is asked to lay parallel to the Cartesian axes of global frame. Of course to gain more insight into the performance of the actual robot under consideration, proper weights must take into account the different sizes of joints motors; moreover, due to the linearity of static relations, the application of higher forces/torques can be represented by simply re-scaling the resulting graphs.

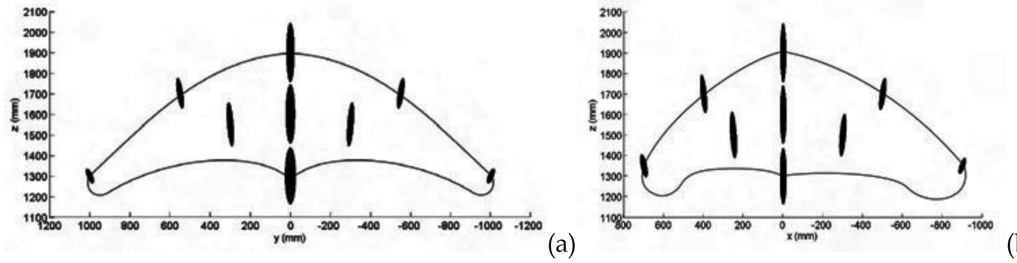


Fig. 10. Force manipulability ellipsoids of the robot Tricept HP1 in two Cartesian planes: upper (a) and side (b) view (end-effector parallel to global frame).

Static performances

For every position of the manipulator inside its workspace (position and attitude of the tool) the 6 joint torques $\boldsymbol{\tau}$ balancing the external wrench of forces $\mathbf{F}$ and moments $\mathbf{M}$ applied at tool tip are given by (10), where the $6x6$ manipulator Jacobian matrix is highly varying with tool position in our case. Once a task has been assigned and the arising forces have been assessed, the feasibility checks can be easily performed: the joint torques that are computed by simulation must be compared with the actual rated torques of the motors driving shoulder and wrist axes, about *13 Nm* and *3 Nm* respectively, but of course the gear ratio of the Harmonic Drives and the efficiency of all the transmission has to be accounted for.

Figure 11 shows a sample analysis performed by means of the developed simulation package: tool tip spans a vertical plane at constant height ($z=1600$ *mm*) with the tool perpendicular to the plane itself and charged by an axial load of *15 kN*. The figure plots the value of the maximum torque requested to any one of shoulder motors: it is noted that the limit value of *13 Nm* is easily overcame when the tool approaches workspace boundaries; wrist motors, on the other hand, are almost idle since the external force passes through the centre of the spherical wrist.

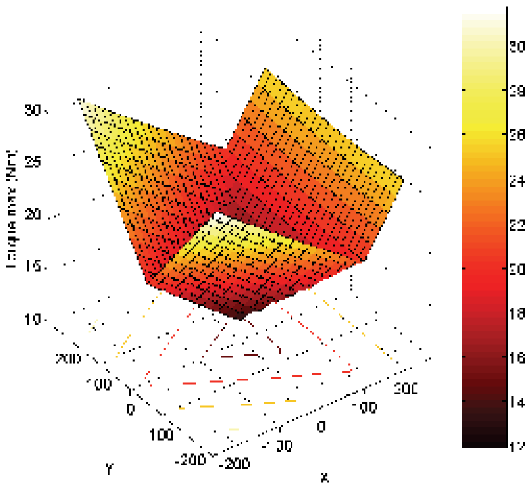


Fig. 11. Maximum torque delivered by shoulder motors (*z=1600 mm*, F_z= *15 kN*).

In the simulation of Fig. 12, instead, the configuration of the robot is still the same but an external force of *500 N* is vertically directed along the *x* axis: in this case shoulder motors are scarcely charged but wrist motors cannot sometimes deliver all the torque that should be required.

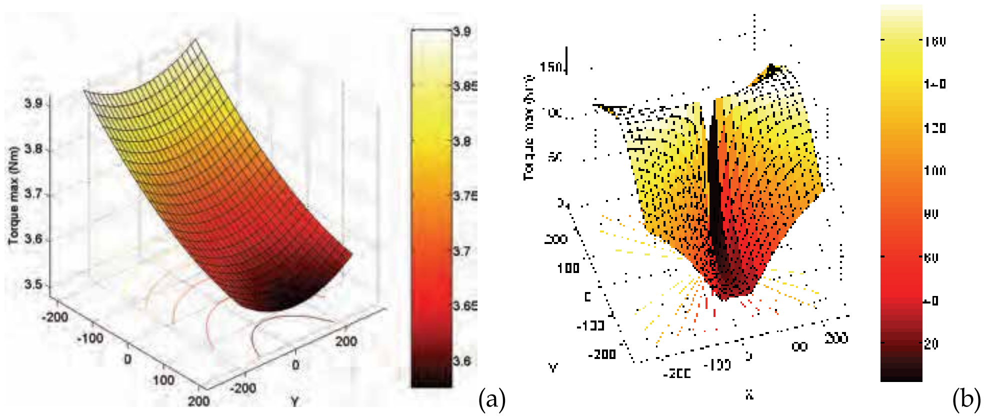


Fig. 12. Maximum torque delivered by shoulder (a) and wrist (b) motors (*z=1600 mm*, F_z= *0.5 kN*).

3.4. Cell simulation

The analysis of all the cell is performed by means of the IGRIP® package by Delmia, Fig. 13: it is a multibody simulation code for applications of industrial automation and its libraries already have all the information needed to simulate and program off-line the main industrial robots presently on the market. Its use is particularly important in case several intelligent pieces of equipment cooperate simultaneously in the cell, therefore needing a strict coordination of the tasks.

In our case, tool trajectory has been previously defined by means of CAD/CAM programs and then statically validated to be sure robot motors can actually push the punch against the blank with the required forces. Then, the path has been imported into the IGRIP®

environment to perform a multibody simulation, providing the kinematic feasibility of the task by taking into account actual joints strokes and possible impacts with obstacles or other equipment. For larger parts the use of a tilting table, synchronised with the robot and governed by the same robot controller, can be necessary, as well as for some operations, e.g. flanging, where the required line of action of tool force would not be close to the preferred thrust direction of the Tricept: in these cases a proper off-line programming reduces the set-up time of the operation and enhances the feasibility chances of the task.
Since IGRIP® data base codes all the kinematic data of the robot and is also aware of control characteristics, robot task can be generated in the proper programming language, i.e. PDL in our case, and directly uploaded to the controller: all production information like cycle-times, occupation of resources, etc. can be estimated off-line at this phase of the work. Of course the PDL program needs to be refined at robot side in any case, but a closer agreement between off-line program and the final code is obtained by exploiting the calibration module, that is purposely developed.

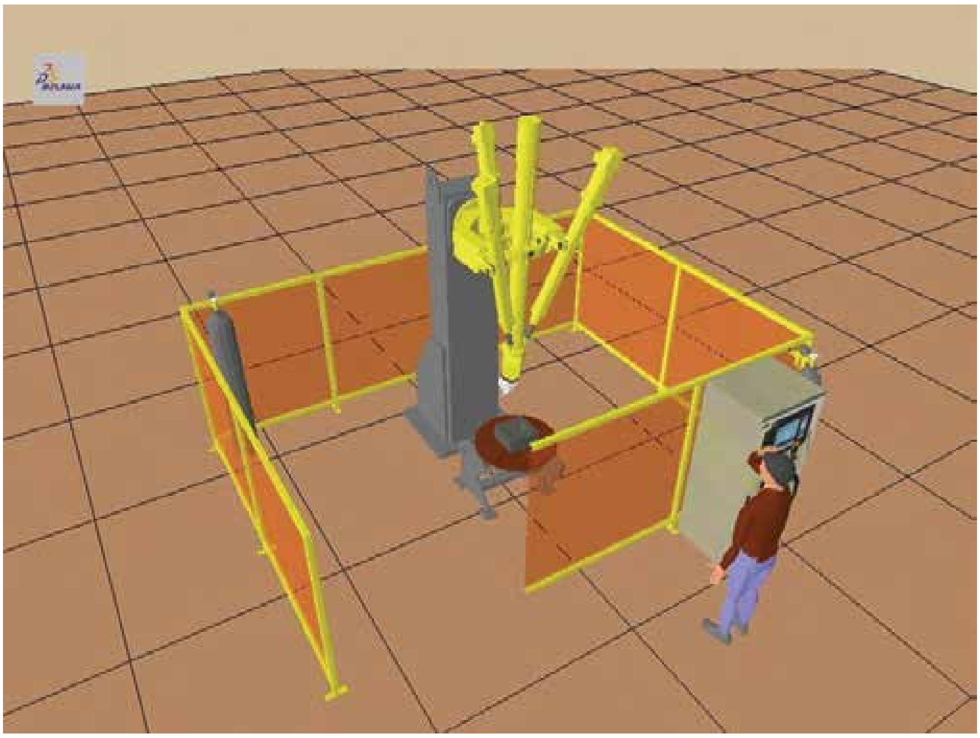


Fig. 13. Multibody simulation of the cell.

4. Experimental and simulative results

The first laboratory tests have been performed on a square socket of small size, *37 mmx37 mm* side and a depth of *21 mm* in Ancona by using the robotised cell and *40 mmx40 mm* side and a depth of *20 mm* in Brescia, where the CNC machine was used. Figure 14 shows the metallic and wooden dies, the blank holder and all the fixtures mounted on a vertical frame.

This set of tests was difficult for the robot due to the particular shape of the die, whose lateral walls were almost vertical; in fact this geometry determines high reaction forces (due to the large contact surface between tool and blank) whose lines of action are almost perpendicular to the (preferential) axial direction, therefore often causing the block of the motors.
During these preliminary experiments it has been shown that, notwithstanding the relatively low velocities of the tool during the process, the velocity profile of the robot influences the quality of the product and even more the performances that can be obtained. In fact during accelerations the inertia forces are added to the elasto-plastic reactions coming from the blank under deformation, while the forming task is easier for the robot during deceleration ramps. Moreover, it resulted that the smaller the radii of the corners are, the higher torques are required to the motors for the impending state of three-axial deformation: better results and lower torques have been registered by using tools with smaller sphere diameters.
It must be emphasized that, when the CNC machine is used to perform the experimental tests, no limits can be identified for this simple geometry. In fact, the machine stiffness and the motor powers allow the deformation of the sheet since the working loads are low. In addition, increasing the feed rate gives the possibility of reducing the working time without affecting the part quality.

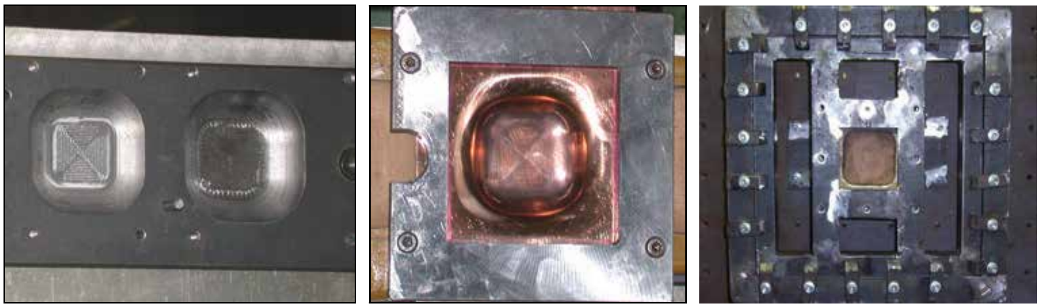


Fig. 14. Metallic and wooden dies with blank holder and clamping fixtures.

The first attempts of forming a steel blank have been performed by using 3 axes paths on a *0.6 mm* thick *AISI 304* stainless steel sheet; in this way it was possible to draw a comparison with the results obtained using both the robotised cell and the CNC machine. In the case of the robotized cell a maximum depth of about *14 mm* was reached in “direct” forming, while the motors blocked at the depth of only *10 mm* in inverse forming (see Fig. 15). In the first case a set of square paths at higher depths has been progressively realised, with a pitch of *0.5 mm* without working the bottom of the pocket; in the second case, after reaching the desired depth, a spiral path was traced with a spiral step of *1 mm*. The quality of surface finishing is quite different in the two cases, since the pocket bottom does not come into contact with the tool in direct forming, while in second case it is deformed by the tool. In this second case the quality is better, even if in the part bottom it is still visible the point corresponding to the straight path of the tool before starting the spiral path which deforms the bottom (Fig. 15). A very tight fixing of the blank holder is required in order to avoid the formation of wrinkles and the rotation of the sheet.
To reduce the working loads, it was necessary to lower the tool pitch to *0.25 mm* instead of *1 mm*. The consequence of this change was an increase of the total cycle time. A reduction of

total cycle time was obtained by using a different language primitive for the path interpolation. Instead of requiring the robot to pass exactly through the programmed points, and therefore to stop when cornering, a spline interpolation allowed the tool to pass "close" to the programmed points, without stopping every time. The quality of the part is still good, while cycle times have been strongly reduced.
In a second set of tests the capabilities of the 6 axes robotized cell were used to orient properly the tool with respect to the sheet local normal and to the feed direction. First of all, it must be made clear that in robot trajectory planning 3 degrees of freedom are constrained by the path to be followed by tool tip, while 2 more degrees of freedom can be used to orient the punch along a "suitable" direction in space; therefore, by having the availability of a 6 axes machine, infinite possible trajectories can still be specified after all.

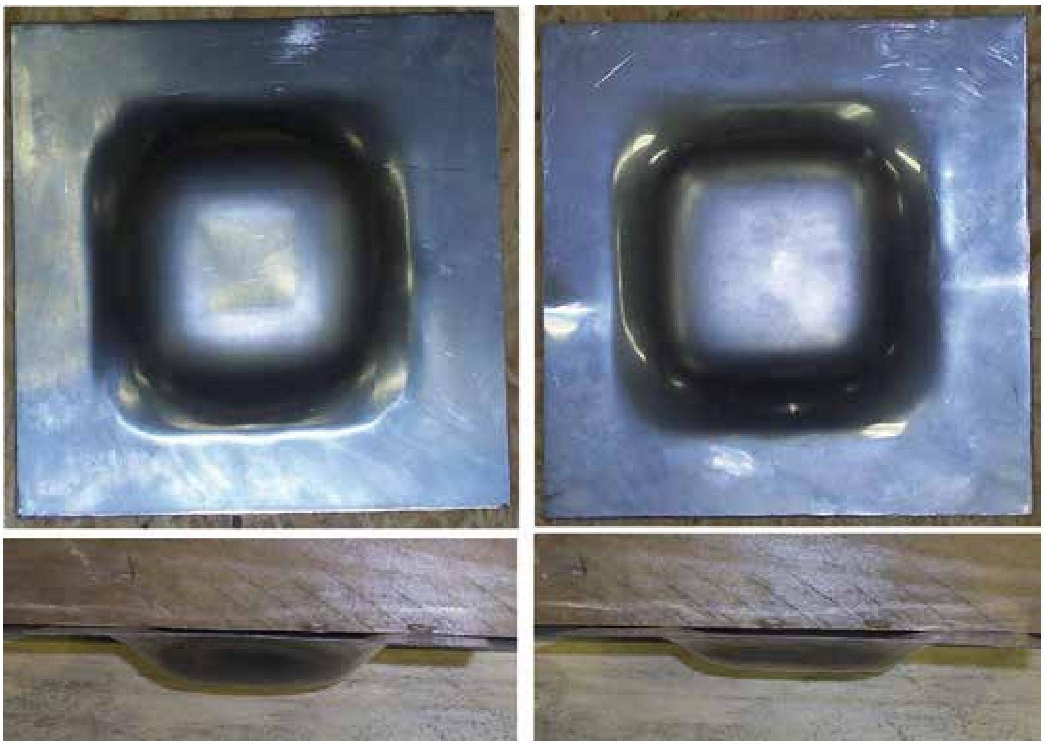


Fig. 15. Deformed steel parts for direct (left) and inverse (right) forming obtained with the robotized cell.

It must be pointed out that the kinematics of the robot is rather complex, therefore the "optimal" alignment of the tool does not necessarily mean that the tool itself is aligned along the direction of robot motion. In our trials, the tilting of the tool with respect to the feeding direction has been set so as to try to "minimise" the moments acting at wrist centre (by the way, the tool is charged by axial loads in this case). Moreover, due to wrist mechanical structure and to the limited winding of its joints, it was necessary to invert the sense of rotation of the motion around the die at every pass. This was also visible on the surface of the final part.

In future work, the redundancy of the robot with respect to the task could be used to:

- take automatically into consideration the constraints coming from the possible mutual collisions between blank/frame and tool/wrist;
- try to avoid the time consuming re-winding of wrist axes, therefore speeding up the process;
- align the whole robot structure along the main direction of the manipulability ellipsoids.

It must be said that the comparison with the 3 axes forming previously performed did not show major differences, apart from a slight higher quality and precision of the finished part mainly due to the smaller bending of the tool.
Considering the experiments conducted with the CNC machine, only the limit of the maximum reachable slot depth was found since it affects the sheet thinning (when thinning is too high the part breaks). Several tool paths of the "direct" forming type were tested by changing the spiral width and the spiral step according to Giardini *et al*. (2005a). The experiments conducted showed that roughness of the produced pieces was influenced only by spiral step and was not influenced by the tool path type. The thickness distribution, analysed along a part transversal section, is affected by the tool path and by the spiral step. In particular, summarising the results obtained in terms of part roughness, sheet thickness and maximum reachable punch depth it is possible to define the optimal Full Die IF process parameters, that is inverse negative forming with spiral step *0.5 mm*, tool pitch *1 mm* and feed rate *400 mm/min*.
The sheet thickness shows a minimum in correspondence of the bottom radius of the pocket, see Fig. 16. In one of the studied cases, the spiral step was so large that the final geometry of the part had a very low quality due to large wrinkles on the pocket bottom.

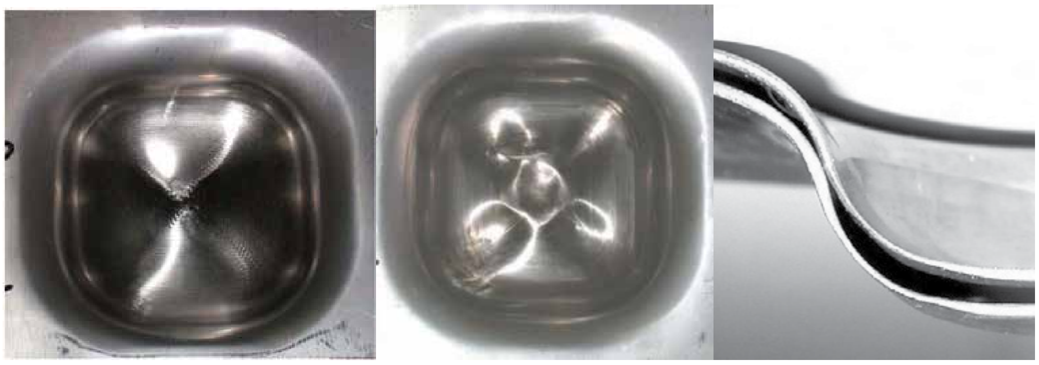


Fig. 16. Deformed steel obtained with the CNC machine.

Figure 17 shows an example of the comparison between simulations and experiments conducted on the CNC machine in terms of sheet thickness measured along a transversal cross-section. In particular the simulative results correspond to the PAMSTAMP® optimized model, that is the model with the most suitable values for friction coefficient (*0.17* between sheet and blank-holder and sheet and die, *0.15* between sheet and punch) and blank-holder force (*50 kN*), referred to *120 mmx120 mmx0.6 mm AISI 304* sheet with spiral step of *0.5 mm*, feed rate of *400 mm/min* and punch diameter of *40 mm.*

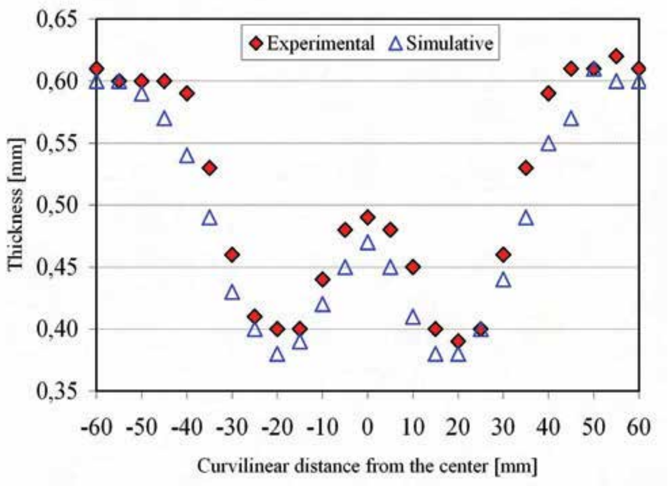


Fig. 17. Simulative and experimental thickness distributions along the transversal cross section of the realised part.

Finally some automotive components have been formed. Starting from CAD models of the parts, the resin dies were produced and then the part programs for the incremental forming have been developed. Figure 18 shows a component used in chair back assembly; two different kinds of steel blanks have been formed with this component: *AISI 304* and *DC04*. The results are satisfactory in both cases but the higher quality of the *AISI 304* part shows that this material can be worked better through the incremental forming technique.
During the working of the central slot, a certain compliance of the wrist became apparent, even if the overall quality of the part was good. Moreover, a little misalignment of the planes of the blank and the die caused the wrinkles that are visible outside the working area: a better realisation and tuning of the blank holder would certainly mitigate this problem.

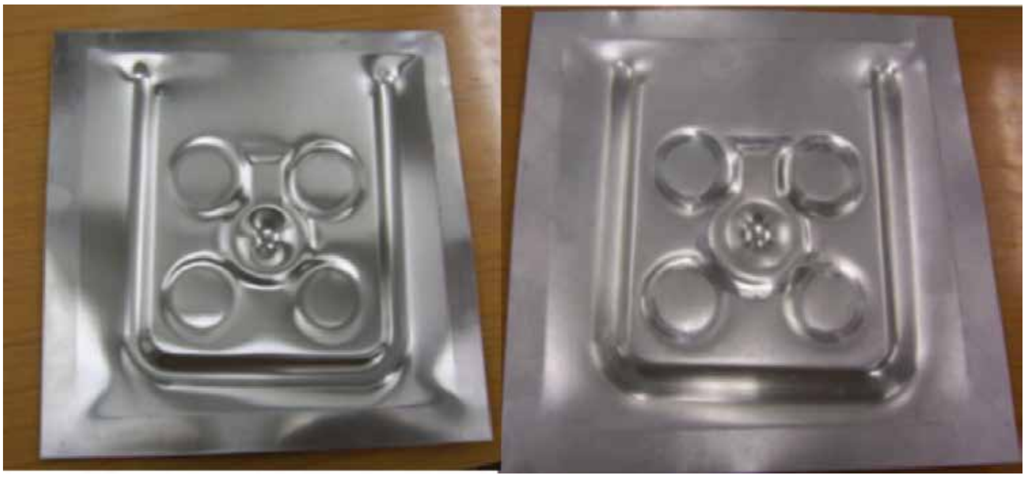


Fig. 18. Formed chair backs: *AISI 304* (left) and *DC04* (right).

The research project analyzed also the feasibility study of some finishing operations just to show the potential versatility of an automated incremental forming cell: to this aim, the hatch of a multipurpose vehicle has been flanged by pressing the outer edge initially bent at 90°. The part was initially clamped between a punch and the die, then a specially designed tool, see Fig. 19a, rolled along all the outside border. The use of this rolling tool to deform locally the outer edge, allowed to flange the part without any problem and to avoid the expensive hydraulic presses generally used in this case or the manual operations that are time consuming and cost expensive but are necessary when the attention is focused on pre-series production. Similar application was studied using a roller mounted on a 3 axes CNC machine. The results are reported in Fig. 19b.

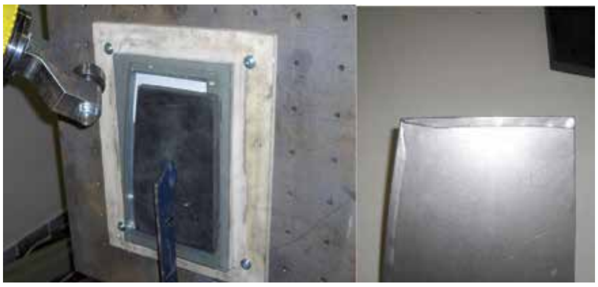
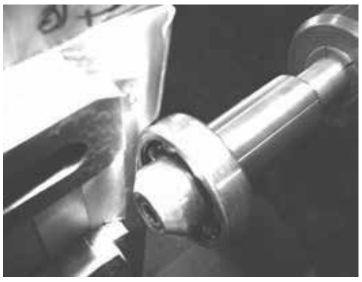

(a)

(b)

Fig. 19. Hemming of sheets in a robotised cell (a) and on a CNC machine (b).

5. Conclusions

The incremental forming of metal sheet parts can be an interesting alternative to manual forging of prototypes and pre-series blanks or to the manufacturing of shells for resin dies used for the production of small batches. Such characteristics of small-volume production would call for an increase in the level of flexibility and automation, possibly leading to the use of CNC machines or robotised cells able to produce or complete the parts. In particular, the use of robotised cells, with automatic tool change, can dramatically reduce the process time since on the same fixture it is possible to deform the part, cut the part, bend or flange the borders, load/unload the part, etc.
The present contribution has described the crossed experiments performed at the Polytechnic University of Marche in Ancona and at the Universities of Brescia and Bergamo to assess the feasibility of the automated processing by using both a traditional CNC machine and an industrial robot. It is noted that the research and industrial processes of incremental forming realised so far have ever used 3-axes CNC milling machines, apart from the hammering process patented by the Fraunhofer Institute for Manufacturing Engineering and Automation of Stuttgart (Shaefer & Schraft, 2005).
Unfortunately the conventional serial robots do not have the required stiffness and are not able to apply the necessary forces to deform incrementally the blank, but the rather new family of parallel robots has characteristics similar to CNC machining centres, while still keeping the versatility of a robot. The complex kinematics of the machine needed the development of a special purpose simulation environment to design beforehand the experiments and assess their feasibility. The necessary force at tool tip has been evaluated both analytically, with a simplified approach based on a plane strain state, and numerically,

by means of a commercial FEM code: different kinds of tool trajectories have been generated by CAM programs, based on actual part geometries. A final assessment of whole cell layout and working has been performed by means of the IGRIP® package, exploiting also the possibilities of the coordination with other external axes (e.g. a revolving table during flanging, a tool changing system, etc.) and its capability to generate the part program for the different commercial controllers.
Several experimental tests have been performed in order to validate the complex methodology for system design and prototyping and to study the several parameters playing a significant role in the process, as for instance: the different types of materials (e.g. *AISI 304* and *DC04* steel, copper), the number of axes of the task (3 or 5), the kind of interpolation between the points, the path imposed to the tool (e.g. depth first or breadth first), the size of punch end, etc. Many tests have been performed on a die specifically developed in-house for the execution system trials, then a few tests have been performed on commercial components, on dies provided by a car manufacturer supplier, with the execution of simple flanging operations too.
At the end of the research a development environment has been set up, able to interface the different software tools in order to support the process designer in making the correct choices. It must be said that, even if the simulation environment proved to be powerful and reliable, the whole design process of an experiment seemed very complicated by the complexity of the used equipment: therefore the use of more powerful robots, with larger workspaces would be desirable. It's Authors' opinion that parallel robots can be a viable alternative to CNC machines for the execution of incremental forming processes, especially if it is possible to exploit the high versatility of the machine for further completing operations. More tests should be needed to completely assess the benefits of robotics, with the possible availability of more powerful machines, that are already available on the market. As for the system design and prototyping tool that has been developed, it proved to be effective and reliable, even if resulted to be quite complex and more integration would be needed between the single software modules.

6. Acknowledgments

This work has been developed within the research programme “Ultraflex” (see Lamier, 2005), partially funded by the Italian Ministry of Industry and by Lamier SpA (La Loggia, Italy).

7. References

Allwood, J.M.; Houghton, N.E. & Jackson, K.P. (2005). The Design of an Incremental Sheet Forming Machine. *Proc. Shemet 2005*, pp. 471-478, Erlangen, Germany, April 5-8

Ambrogio, G.; Filice, L.; Gagliardi, F. & Micari, F. (2005) Sheet Thinning Prediction in Single Point Incremental Forming, *Proc. Shemet 2005*, pp. 479-486, Erlangen, Germany, April 5-8.

Amino, H.; Makita, K. & Maki, T. (2000). Sheet Fluid Forming and Sheet Dieless NC Forming, *Proc. New Developments in Sheet Metal Forming 2000*, pp. 39-66, Stuttgart, Germany, May 23-24.

Bambach, M.; Azaouzi, M.; Campagne, L.; Hirt, G. & Batoz, J.L. (2005). Initial Experimental and Numerical Investigations into a Class of New Strategies for Single Point Incremental Sheet Forming (SPIF), *Proc. Esaform 2005,* pp. 671-674, Cluj-Napoca, Romania, April 27-29.

Callegari, M.; Gabrielli, A.; Palpacelli, M.-C. & Principi, M. (2006). Robotised Cell for the Incremental Forming of Metal Sheets, *Proc. ESDA 2006, 8th Biennial ASME Conference Engineering Systems Design and Analysis*, Turin, July 4-7.

Callegari, M.; Palpacelli, M. & Principi, M. (2005). Analisi della manipolabilità del robot industriale Tricept, *Atti del XVII Congresso AIMeTA di Meccanica Teorica e Applicata*, Firenze, 11-15 Settembre 2005.

Ceretti, E.; Giardini, C.; Attanasio, A. & Maccarini, G. (2002). Some Experimental Edvidences in Sheet Incremental Forming on CNC Machines. *Proc. Numisheet 2002*, pp. 429-434, Jeju Island, Korea, October 21-25.

Ceretti, E.; Giardini, C. & Attanasio, A. (2003). Sheet Incremental Forming on CNC Machines, *Proc. SheMet 2003*, pp. 49-56, University of Ulster, Newtownabbey, UK, April 14-16.

Ceretti, E.; Giardini, C.; Attanasio A. (2004). Experimental and Simulative Results in Sheet Incremental Forming on CNC Machines, *Journal of Materials Processing Technology*, vol. 152, pp. 176-184.

Comau (1995). *Tricept HP1. Maintenance Manual*. rel 3.X/4.X

Comau (2001). *Tricept HP1. User's Manual*. rel 3.X/4.X

Duflou, J.R.; Lauwers, B.; Verbert, J.; Tunckol, Y. & De Baerdemaeker, H. (2005b). Achievable Accuracy in Single Point Incremental Forming: Case Studies, *Proc. Esaform 2005*, pp. 675-678, Cluj-Napoca, Romania, April 27-29.

Duflou, J.R.; Szekeres, A. & Vanherck, P. (2005a). Force Measurements for Single Point Incremental Forming: An Experimental Study, *Proc. Shemet 2005*, pp. 441-448, Erlangen, Germany, April 5-8.

Filice, L.; Fratini, L. & Micari, F. (2002) Analysis of Material Formability in Incremental Forming, *Annals of CIRP*, Vol. 51, No. 1, pp. 199-202.

Giardini, C.; Ceretti, E.; Attanasio, A. & Pasquali, M. (2004a). Analysis of the Influence of Working Parameters on Final Part Quality in Sheet Incremental Forming, *Proc. 3rd International Conference and Exibition on Design and Production of Dies and Molds*, pp. 191-198, Bursa, Turkey, June 17-19.

Giardini, C.; Ceretti, E.; Attanasio, A. & Pasquali, M. (2004b). Feasibility Limits in Sheet Incremental Forming: Experimental and Simulative Analysis, *Proc. Esaform 2004*, pp. 515-518, Trondheim, Norway, April 28-30.

Giardini, C.; Ceretti, E. & Contri, C., (2004c). Analysis of Material Behavior in Sheet Incremental Forming Operations, *Proc. 8th NUMIFORM 2004*, pp. 1016-1021, vol. 712, Columbus-Ohio, USA, June 13-17.

Giardini, C.; Ceretti, E. & Attanasio, A. (2005a). Further Experimental Investigations and FEM Model Development in Sheet Incremental Forming, *Proc. Shemet 2005*, pp. 501-508, Erlangen, Germany, April 5-8.

Giardini, C.; Ceretti, E. & Attanasio, A. (2005b). Optimization of Sheet Incremental Forming Process by Means of FE Simulations, *Proc. Esaform 2005*, pp. 691-694, Cluj-Napoca, Romania, April 27-29.

He, S.; Van Bael, A.; Van Houtte, P.; Szekeres, A.; Duflou, J.R.; Henrard, C. & Habraken, A.M. (2005a). Finite Element Modeling of Incremental Forming of Aluminum Sheets, *Proc. Shemet 2005*, pp. 525-532, Erlangen, Germany, April 5-8.

He, S.; Van Bael, A.; Van Houtte, P.; Tunckol, Y. & Duflou, J.R. (2005b). Effect of FEM Choices in the Modelling of Incremental Forming of Aluminium Sheets, *Proc. Esaform 2005*, pp. 675-678, Cluj-Napoca, Romania, April 27-29.

Hirt, G.; Bambach, M. & Junk, S. (2003). Modelling of the Incremental CNC Sheet Metal Forming Process, *Proc. SheMet 2003*, pp. 495-502, University of Ulster, Newtownabbey, UK, April 14-16.

Iseki, H. (2001). An Approximate Deformation Analysis and FEM Analysis for the Incremental Bulging of Sheet Metal Using a Spherical Roller, *Journal of Material Processing Technology*, Vol. 111, pp. 150-154.

Iseki, H. & Naganawa, T. (2002). Vertical Wall Surface Forming of Rectangular Shell Using Multistage Incremental Forming with Spherical and Cylindrical Rollers, *Journal of Materials Processing Technology*, Vol. 130-131, pp. 675-679.

Jesweit, J.; Duflou, J.R. & Szekeres, A. (2005a). Forces in Single Point and Two Point Incremental Forming, *Proc. Shemet 2005*, pp. 449-456, Erlangen, Germany, April 5-8.

Jesweit, J.; Young, D. & Ham, M. (2005b). Non-Traditional Forming Limit Diagrams for Incremental Forming, *Proc. Shemet 2005*, pp. 409-416, Erlangen, Germany, April 5-8.

Kim, Y. H. & Park, J. J. (2003). Effects of Process Parameters on Formability in Incremental Sheet Metal Forming Technique, *Journal of Material Processing Technology*, Vol. 140, p. 447-453.

Kochan, A. (2001). Dieless forming. *Assembly Automation*. Vol. 21, No. 4, pp. 321-322.

Lamier. (2005). New Advances in Process Flexibility: the Perspective of an OEM Supplier, *Proc. 9th International Conference "FLORENCE ATA 2005 - Vehicle architectures: evolution towards improved safety, low weight, ergonomics and flexibility"*, Florence, May 11-13.

Lamminen, L. (2005). Incremental Sheet Forming with an Industrial Robot – Forming Limits and Their Effect on Component Design, *Proc. Shemet 2005*, pp. 457-464, Erlangen, Germany, April 5-8.

McLoughlin, K.; Cognot, A. & Quigley, E. (2003). Dieless Manufacturing of Sheet metal Components with non Rigid Support, *Proc. SheMet 2003*, pp. 123-130, University of Ulster, Newtownabbey, UK, April 14-16.

Meier, H.; Dewald, O. & Zhang, J. (2005). A New Robot-Based Sheet Metal Forming Process, *Proc. Shemet 2005*, pp. 465-470, Erlangen, Germany, April 5-8.

Merlet, JP. (2005). *Parallel Robots*, 2nd Ed., Springer, Dordrecht.

Park, J.-J. & Kim, Y.-H. (2002). Effect of Process Parameters on Formability in Incremental Forming of Sheet Metal. *Journal of Materials Processing Technology*. Vol. 130-131, pp. 42-46.

Park, J.-J. & Kim, Y.-H. (2003). Fundamental Studies on the Incremental Sheet Metal Forming Technique. *Journal of Materials Processing Technology*. Vol. 140, pp. 447-453.

Schafer, T. & Schraft, R.D. (2005). Incremental Sheet Metal Forming by Industrial Robots. *Rapid Prototyping Journal*. Vol. 11, No. 5, pp.278–286.

Shim, M., & Park, J. (2001). The Formability of Aluminium Sheet in Incremental Forming, *Journal of Materials Processing Technology*, Vol. 113, pp. 654-658.

Shima, S. (2001). Incremental Forming: State of the Art, *Proc. IPMM 2001: Intelligent Processing and Manufacturing of Materials*, Vancouver, British Columbia, Canada, July 29 - August 3.

Siciliano, B. (1999). The Tricept Robot: Inverse Kinematics, Manipulability Analysis and Closed-Loop Direct Kinematics Algorithm. *Robotica*. Vol. 27, pp. 437-445.

Siegert, K.; Rennet, A. & Fann, K.J. (1997). Prediction of the Final Part Properties in Sheet Metal Forming by CNC-Controlled Stretch Forming, *Journal of Materials Processing Technology*, vol. 71, pp. 141-146.

Yoon, S. J. & Yang, D.Y. (2003). Development of a Highly Flexible Incremental Roll Forming Process for the Manufacture of a Doubly Curved Sheet Metal, *Annals of CIRP*, Vol. 52, No. 1, pp. 201-204.

Machining with Flexible Manipulators: Critical Issues and Solutions

Jianjun Wang, Hui Zhang, Zengxi Pan
ABB Corporate Research Center
2000 Day Hill Road, Windsor, CT, USA 06095

1. Introduction

The automotive industry represents the fastest-growing market segment of the aluminium industry, due to the increasing usage of aluminium in cars. The drive behind this is not only to reduce the vehicle weight in order to achieve lower fuel consumption and improved vehicle performance, but also the desire for more sustainable transport and the support from new legislation. Cars produced in 1998, for example, contained on average about 85 Kg of aluminium. By 2005, the automotive industry will be using more than 125 Kg of aluminium per vehicle. It is estimated that aluminium for automotive industry alone will be a 50B$/year market.
Most of the automotive aluminium parts start from a casting in a foundry plant. The downstream processes usually include cleaning and pre-machining of the gating system and riser, etc., machining for high tolerance surfaces, painting and assembly. Today, most of the cleaning operations are done manually in an extremely noisy, dusty and unhealthy environment. Therefore, automation for these operations is highly desirable. However, due to the variations and highly irregular shape of the automotive casting parts, solutions based on CNC machining center usually presented a high cost, difficult-to-change capital investment.
To this end, robotics based flexible automation is considered as an ideal solution for its programmability, adaptivity, flexibility and relatively low cost, especially for the fact that industrial robot is already applied to tend foundry machines and transport parts in the process. Nevertheless, the foundry industry has not seen many success stories for such applications and installations. Currently, more than 80% of the application of industrial robots is still limited to the fields of material handling and welding. (Figure 1)
The major hurdle preventing the adoption of robots for material removal processes is the fact that the stiffness of today's industrial robot is much lower than that of a standard CNC machine. The stiffness for a typical articulated robot is usually less than 1 N/μm, while a standard CNC machine center very often has stiffness greater than 50 N/μm.
Most of the existing literature on machining process, such as process force modelling (Kim et al., 2003; Stein & Huh, 2002], accuracy improvement (Yang 1996) and vibration suppression (Budak & Altintas, 1998) are based on the CNC machine. Research in the field of robotic machining is still focused on accurate off-line programming and calibration (Chen & Hu, 1999; Sallinen & Heikkila, 2000; Wang et al., 2003a, 2003b). Akbari et al. (Akbari & Higuchhi, 2000) describe a tool angle adjustment method in a grinding application with a

small robot. In that case the process force is very small. Matsuoka etc al (Matsuoka et al., 1999) study the characters of an articulated robot in a milling process avoiding large process force by using an end mill with small diameter and high spindle speed. Without the capability of real-time force control, the method to eliminate the force effect on the robotic machining process has not been fully addressed in the research community or in industry.

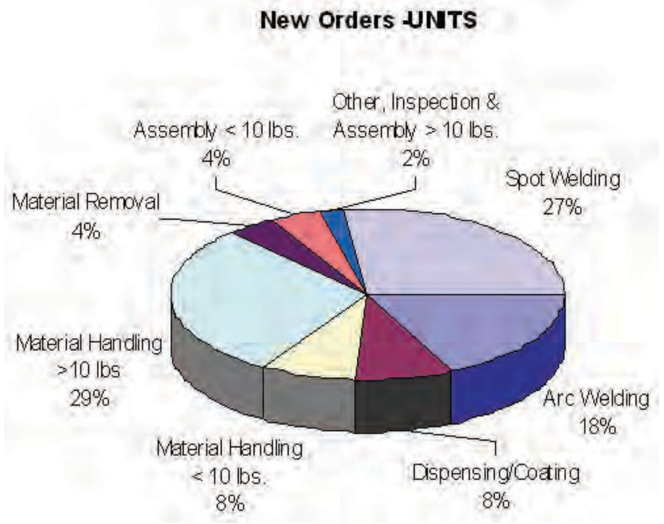


Fig. 1. 2003 Robot Applications in North America . A total of 12,367 robots valued at $876.5 million were ordered. When sales to companies outside North America are added in, the total for North American robotics suppliers is 12,881 robots valued at $913 million. NOTE: These numbers include results from North America and Outside North America. Source: Robotic Industries Association.

Machining processes, such as grinding, deburring, polishing, and milling are essential force tasks whose nature requires the robot end effector to establish physical contact with the environment and exert a process-specific force. The inherent lower stiffness of the robot has presented many challenges to execute material removal applications successfully. First and foremost, the lower stiffness makes chatter much easier to occur in robot than in CNC machine. Severe low frequency chatter has been observed in milling and deburring processes. Although extensive research on chatter has been conducted, none of the existing research has focused on chatter mechanism in robotic machining process. The result is that without a good understanding or even a rule of thumb guideline, robotic engineers and technicians have to spend tremendous time on trial and error for the sheer luck of stumbling a golden setup or has to sacrifice the productivity by settling on very conservative cutting parameters.

The second challenge is the structure deformation and loss of accuracy due to the required machining force. The predominant cutting action in machining involves shear deformation of the work material to form a chip. The contact between the cutting tool and the workpiece generates significant forces. As a result, a perfect robot program without considering contact and deformation will immediately become flawed as the robot starts to execute the machining task. Unlike multi-axis CNC machine centers, such deformation is coupled and varies even subjected to the same force in different workspace locations. Such coupling results in deformation not only in the direction of reaction force and can generate some counter-intuitive results.

Lastly, the machining of casting parts presents a unique challenge with non-uniform cutting depth and width. In conventional robot programming and process planning practice, the cutting feed rate is constant even with significant variation of cutting force from part to part, which dictates a conservative cutting feed rate without violating the operational limits. Therefore, it is desirable to maximize the material removal rate (MRR) and minimize cycle time by optimizing the cutting feed rate based on a programmed spindle load. By optimizing the feed rate in real time, one could compensate for conservative assumptions and process variations to help reduce cycle time. Every part, including the first one, is optimized automatically, eliminating the need for manual part program optimization.

This chapter will address the above critical challenges with detailed experimental observation, in-depth analysis of underlying root causes, and practical solutions to improve stability, accuracy and efficiency (Pan et al., 2006; Zhang et al., 2005). If industrial robots could be made to provide the same performance under contact situations as that they already have, then robotic machining would result in significant cost savings for a lot of potential applications. Moreover, if such applications can be proven to be economically viable and practically reliable, one would expect to witness many success stories in the years to come. This chapter is organized in six sections including the introduction. Section II is devoted to the modelling of robotic machining process and serves as the basis for the following 3 sections, where chatter analysis, deformation compensation and MRR control are presented respectively. The whole chapter is then concluded in Section VI.

2. Modelling of Robotic Machining Process

Machining processes are essential force tasks which require the end effector of the robot to establish physical contact with the environment and exert a process-specific force. Characterization of the interaction bewteen the tool and the workpiece is essential for analyzing and solving the outstanding problems in robotic machining. For this purpose, a simplified interaction model is introduced here as the basis for the following sections.

As shown in figure 2, the dynamic interfaction in the machining processes is modelled as a closed loop connection between a force model of the machining process and a dynamic model of the machine tool-workpiece structure.

2.1 Robot-tool structure model

For a typical robotic machining workcell setup where the robot holds the spindle and the workpiece is mounted on a strong stationary foundation such as a steel table, the model of the machine tool-workpiece structure can be simplied as that of robot-tool structure due to the much larger stiffness of the workpiece-table strucutre. For an articulated 6-axis serial robot, the robot-tool structure is modeled by the transfer function in s domain and differential equation in time domain as:

$$\{\Delta\} = [G(s)]\{F\} \tag{1}$$

$$[M]\{\ddot{\Delta}\} + [C]\{\dot{\Delta}\} + [K]\{\Delta\} = \{F\} \tag{2}$$

where $[G(s)]$ is matrix of system transfer functions, $[M]$, $[C]$ and $[K]$ are 6×6 system mass, damping and stiffness matrix respectively. These matrixes are generally configuration dependent, but for the convenience of analysis, they can be treated as constant when robot only moves in a small range.

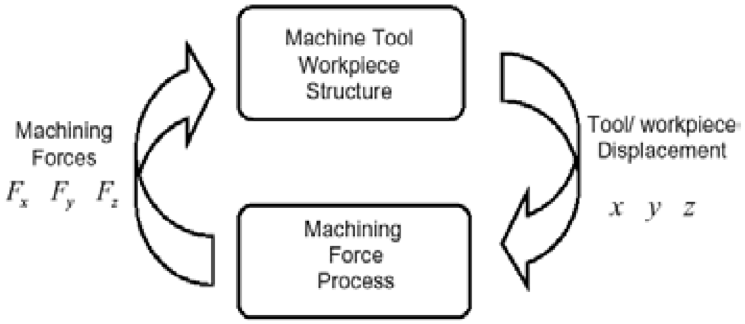


Fig. 2. Closed-loop model of dynamic interaction in the machining processes.

The mass matrix is related to robot rotational inertia in joint space as

$$M = J(Q)^{-T} I_q(Q) J(Q)^{-1} \tag{3}$$

Where Q is the joint angles, $J(Q)$ is the Jacobian matrix of the robot and $I_q(Q)$ is a 6×6 matrix representing the robot rotational inertia in joint space. $I_q(Q)$ is a function of joint angle and is not a diagonal matrix. It could be derived from robot kinematic model by Newton-Euler method or Lagrangian method, if the rigid body inertia parameters are available.

Similar to the mass matrix, stiffness matrix in Cartesian space K and joint space K_q are related by Jacobian matrix of robot as:

$$K = J(Q)^{-T} K_q J(Q)^{-1} \tag{4}$$

When the compliance of robot structure mostly comes from the deformation of gear box, the robot joint stiffness K_q can be modeled as a constant 6×6 diagonal matrix:

$$\tau = K_q \cdot \Delta Q \tag{5}$$

Where τ is the torque load on the six joints, ΔQ is the 6×1 deformation vector of all joints. This simplification is justifiable for industrial robots as they are designed to achieve high positioning accuracy and high strength. Elastic properties of the arms are insignificant. As result, the dominant contribution factor for a large deflection of the manipulator tip position is the joint compliance, e.g., due to gear transmission elasticity. Robot stiffness model could therefore be reduced to six rotational stiffness coefficients in the joint space. From the control point of view, this model is also easy to implement, since all industrial robot controllers are decoupled to SISO joint control at the servo level. Joint deformation could be directly compensated on the joint angle references passed to the servo controller.

Compared to CNC machines, articulated robots have totally different structural characteristics (table 1). First all, the serial structure of articulated robot has a much lower stiffness than that of a standard CNC machine. The stiffness for an articulated robot is usually less than 1 N/μm, while a standard CNC machine very often has stiffness greater than 50 N/μm. Secondly, with a large mass, the base natural frequency of robot structure is very low. Typical for a large robot, it is around 10Hz compared with several hundred Hz or even more than one thousand Hz for the moving component of a CNC machine. Lastly, the stiffness matrix of the robot is configuration dependent and nondiagonal. This means that, first, the force and deformation in Cartesian space is coupled, the force applied in one direction will cause the deformation in all directions/orientations; second, at different locations, the Cartesian stiffness matrix will take different values. As it will be shown in the following sections, a lot of unique problems present in robotic machining have to do with the low and coupled stiffness matrix.

2.86E+02	-8.78E+02	8.12E+02	1.49E+06	3.01E+05	3.97E+05
-8.78E+02	2.39E+02	5.48E+03	-1.85E+05	-5.46E+05	-4.09E+05
8.12E+02	5.48E+03	4.91E+02	-4.56E+06	4.31E+05	8.62E+05
1.49E+06	-1.85E+05	-4.56E+06	7.53E+07	5.85E+08	-7.62E+08
3.01E+05	-5.46E+05	4.31E+05	5.85E+08	1.68E+08	3.20E+08
3.97E+05	-4.09E+05	8.62E+05	-7.62E+08	3.20E+08	1.30E+08

Table 1. One example of the Cartesian Stiffness Matrix (Units are N/mm, N/rad, N·mm/mm, and N·mm/rad respectively.)

2.2 Machining force model

Machining force results from the interaction between the tool and workpiece. It directly affects all facets of the machining operation. Accurate models are necessary to estimate force levels, spindle power requirements, etc. to aid the designer in planning the machining operation. A tremendous amount of effort has occurred in the area of cutting-force modeling over the past several decades. Models used for simulation purposes are often quite complex and incorporate effects such as tool and spindle run out, structural vibrations and their impact on the instantaneous feed, the effect of the cutting tool leaving the workpiece due to vibrations, intermittent cutting, tool geometry, etc. But for controller analysis and design, force model is typically very simple.

Throughout this chapter, the resultant machining force is represented as a static or a linear first-order model as:

$$F = K \cdot w \cdot d \cdot f \tag{6}$$

$$F = K \cdot w \cdot d \cdot f \frac{1}{\tau_m s + 1} \tag{7}$$

where w is width of cut, d is the depth of cut, f is the feed, K is machining coefficent dependent on workpiece material, tool shape, tool matrial and many other factors, τ_m is the machining process time constant. Since one spindle revolution is required to develop a full chip load, τ_m is 63% of the time required for a spindle revolution (Daneshmend & Pak, 1986). For industrial robot typical equipped with 10Hz bandwidth of servo loop, the force process gain may be seen as $\theta = K \cdot w \cdot d$, which is sensitive to the process inputs.

The force process in machining is actually a nonlinear system since the cutting parameters (feed, depth-of-cut, and width-of-cut) are related to machining force in a nonlinear manner. In cases where the linear force model fails to predict the machining force accurately, a nonlinear model should be used. A good model for the setup used in this chapter is found to be:

$$F(s) = K \cdot w^{\gamma} \cdot d^{\alpha} \cdot f^{\beta} \tag{8}$$

and

$$F(s) = K \cdot w^{\gamma} \cdot d^{\alpha} \cdot f^{\beta} \frac{1}{\tau_m s + 1} \tag{9}$$

where the constants α, β and γ depend on such factors as tool material, tool geometry, workpiece material, etc. and do not vary significantly during the machining operation. The cause of the nonlinearities is not well understood but is believed to be caused by the strength and temperature of the cutting zone, as well as the resultant force direction, as the process parameters vary. These nonlinear effects have long been recognized and have been mentioned in the force control literature.

3. Chatter Analysis

One of the major hurdles preventing the adoption of robot for machining process is chatter. Tobias (Tobias & Fishwick, 1958) and Tlusty (Tlusty & Polacek, 1963) recognized that the most powerful sources of chatter and self-excitation were the regenerative and mode coupling effects. Regenerative chatter is based on the fact that the tool cuts a surface already cut during the previous revolution, so the cutting force and the depth of cut vary. Mode coupling chatter is due to the fact that the system mass vibrates simultaneously in the directions of the degrees of freedom (DOF) of the system, with different amplitudes and with a difference in phases. Regenerative chatter happens earlier than the mode coupling chatter in most machining processes, as explained by Altintas (Altintas, 2000).
Although extensive research on chatter has been carried out, none of the existing research has focused on chatter mechanism in robotic machining process. The result is that robotic engineers and technicians are frustrated to deal with elusive and detrimental chatter issues without a good understanding or even a rule of thumb guideline. Very often, to get their process working correctly, one has to spend tremendous time on trial and error for the sheer luck of stumbling a golden setup or has to sacrifice the productivity by settling on conservative cutting parameters much lower than the possible machining capability. This section is trying to bridge the gap by pointing out the underline chatter generation mechanism (Pan et al., 2006). First, the characteristics of chatter in robotic machining process are presented, followed by the detailed analysis of chatter mechanism applying both regenerative and mode coupling theory. Further experimental results are then provided to verify the theoretical analysis. Finally stability criteria and insightful guidelines for avoiding chatter in robotic machining process are presented.

3.1 Characteristics of chatter in robotic machining process

Severe low frequency chatter has been observed ever since when robot was first applied in machining process, nevertheless, no theoretical explanation and analysis are available in the existing literature to date. The conventional wisdom is that this is due to the obvious fact that the robot is much less stiffer than CNC machine, but no answer is provided for the further explanation. The reason for this blank may be the lack of enough sensory information, in particular, the process force information. In addition to the surface damage on the workpiece due to chatter marks, the occurrence of severe chatter results in many adverse effects, which may include a poor dimensional accuracy of the workpiece, a reduction of tool life, and a damage to the machine, etc. Certain conservative cutting parameters, which were proposed by other researchers, intended to avoid chatter at the expense of the loss of productivity.

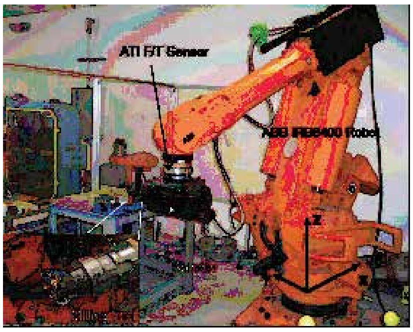


Fig. 3. Setup of robotic end milling.

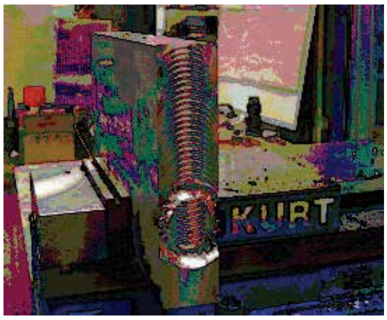


Fig. 4. Chatter marks on the workpiece.

In the present work, a robotic milling work cell is setup with ABB IRB6400 industrial manipulator. The spindle is mounted on robot wrist while the workpiece is fixed on the steel table. An ATI 6DOF Force/Torque sensor is set up between the robot wrist and spindle as shown in Figure 3. After compensating the gravity of spindle and tool, 3 DOF machining force could be measured accurately. When chatter occurs, the amplitude of cutting force increases dramatically and the chatter frequency is observed from the Fast Fourier Transform of force data. The experimental conditions for robotic end milling are summarized in Table 2.

Test	End milling	Deburing
Robot	ABB IRB6400	ABB IRB6400
Workpiece	A2024, L300mm×W38mm×H150mm	A2024, L300mm×W51mm×H150mm
Spindle type	SETCO,5HP, 8000RPM	GCOLOMBO,11HP, 24,000RPM
Tool type	SECO Φ52mm, Round insert, Φ1.89mm×5	SANDVIK, Φ20mm, Helical 2-flute
Cutting fluid	- (Dry cutting)	- (Dry cutting)
Feed rate	30 mm/s	60 mm/s
Spindle speed	3600 RPM	18,000 RPM
DOC	1-4 mm	5mm
WOC	38mm	15mm

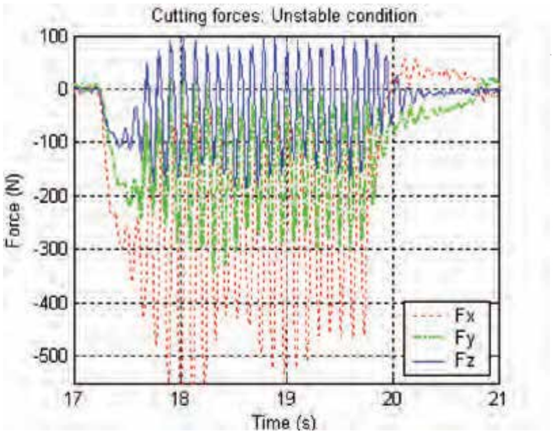
Table 2. Experimental conditions for robotic machining

In most situations, the cutting process is stable; the work cell could conduct 4-5mm depth-of-cut (DOC) until reaching the spindle power limit. Nevertheless, while feed in -Z direction, severe low frequency (10Hz) chatter occurs when the DOC is only 2 mm. The characteristics of this low frequency chatter are:

1. The frequency of chatter is the robot base natural frequency at 10Hz. It does not change with the variation of cutting parameters such as spindle speed, width-of-cut (WOC), feed speed and the location of workpiece.
2. When chatter occurs, the entire robot structure start to vibrate. The magnitude of vibration is so large that obvious chatter marks are left on the workpiece surface. (Figure 4)
3. In the cutting setup of Figure 3 and Table 2, using the exact same cutting parameters (DOC, RPM, WOC, Feed speed), chatter starts to occur when feed in –Z

direction, DOC=2mm, while the process is stable when feed in +Z, ±X direction, even with the DOC=4mm. The cutting forces in unstable (feed in –Z direction) and stable (feed in +Z direction) conditions are plotted in Figure 5.

4. The cutting process has different chatter limit at different locations in the robot workspace. Machining experiments are carried out at three different locations along the table, defined as L1, L2 and L3. They have the same Y, Z coordinates and different X coordinate. L1 is the location shown in Figure 3, L3 is on the other corner, and L2 is in the middle. Without changing other cutting parameters, chatter limit for L1 is DOC=2mm, for L2 is DOC=1.5mm, for L3 is DOC=1.1mm.



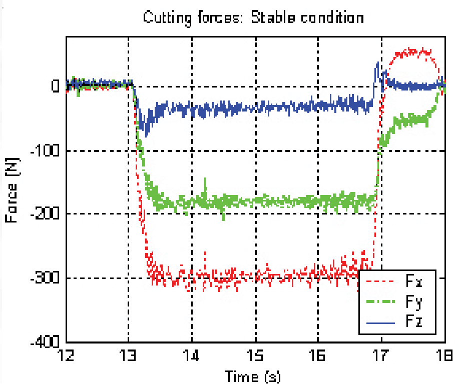


Fig. 5. Cutting force profile. (Left) force plot while low frequency chatter happens, cutting condition listed in Table 2. (Right) force plot while system stable, cutting condition is the same except that feed in opposite direction.

3.2 Chatter analysis based on regenerative mechanism

Regenerative chatter is a self-excited vibration in a machining operation resulting from the interference between the current machining pass and the wavy surface generated during previous machining passes. The energy for the chatter comes from the forward motion of the tool/workpiece. The frequency is typically slightly larger than the natural frequency of the most flexible vibration mode of the machine-tool system. The corresponding mathematical models are delay-differential equations (DDEs) or periodic differential equations (PDFs). Merritt (Merritt, 1965) combined the regenerative theory and a feedback loop to derive a systematic and graphical stability criterion. His approach introduced an elegant new method of stability analysis that was later adopted by many investigators and led to major advances towards the understanding and prediction of the chatter.

A simplified one-DOF analysis could easily prove how regenerative mechanism will not introduce chatter at as low as 10 Hz, while the spindle speed is 3600RPM. The block diagram of one-DOF close loop system is shown in Figure 6. The transfer function of this model is:

$$G(s) = \frac{1}{ms^2 + cs + k + K_p(1 - e^{-s\tau})} \tag{10}$$

where τ is the time delay between the current cut and the previous cut. It is related to the spindle speed and number of teeth on the cutting tool. The stability margin is when the characteristic equation of this transfer function has pure imaginary solutions, which are:

$$d = \frac{m(\omega_c^{\,2} + 4\omega_n^{\,2}\zeta^2)}{2K_p} \tag{11}$$

$$\Omega = 60\left(\frac{\omega_c}{2\phi - \pi(1-2n)}\right) \qquad n = 1,2,3\ldots \tag{12}$$

where d is DOC, Ω is RPM, $\omega_n = \sqrt{k/m}$, $\zeta = c/2m$, and $\omega_c = \omega_n\sqrt{1-\zeta^2}$. The corresponding stability lobe is plotted in Figure 7. Obviously, with spindle speed $\Omega = 3600$, robot works on the very right side of first stability lobe, where DOC is almost unlimited. Based on regenerative chatter theory, one would not expect to observe the low frequency chatter. However, from the experimental test, such chatter behavior indeed exists, which forced us to look into other theories to explain the underline mechanism for this kind of chatter. In the following subsection, we would explore the mode coupling chatter theory and find that is the most reasonable explanation to the aforementioned problem.

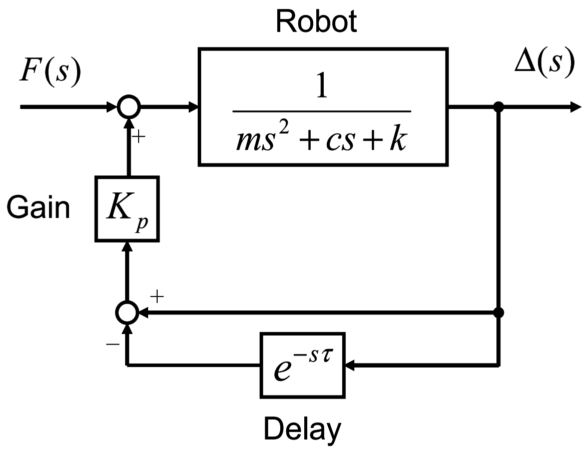


Fig. 6. Block diagram of one-DOF regenerative chatter model.

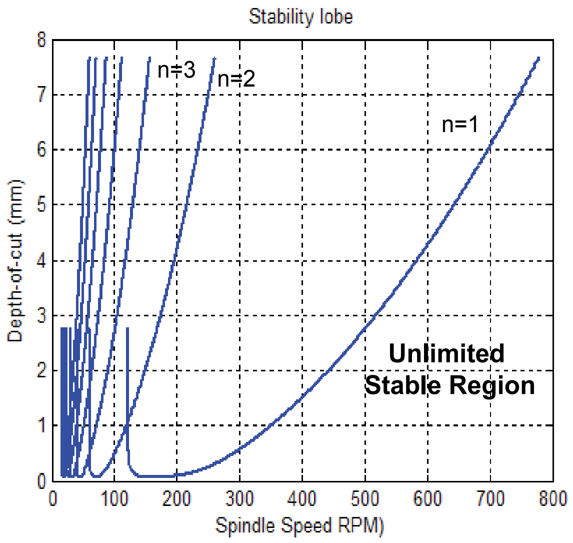


Fig. 7. Stability lobe of robotic end milling.

3.3 Chatter analysis based on mode coupling mechanism

Generally, from Eq. (1) and Eq. (2), the dynamic model of the system is formulated as:

$$[M]\{\ddot{\Delta}\}+[C]\{\dot{\Delta}\}+[K]\{\Delta\}=[K_p]\{\Delta\} \tag{13}$$

The stability of the system depends on the eigenvalues of the equation above. Since the focus here is to analyze the chatter due to the mode coupling effect, the following simplifications are made:

1. Since damping effect will always increase the stability of the system and it is difficult to be identified accurately, we only analyze the undamped system.
2. While round inserts are applied in robotic end milling operation, the machining force in feed direction is much smaller than the forces in cutting and normal direction (Figure 5). Thus while feed in Z direction, we could simplify the analysis into a 2-DOF problem in X-Y frame.

Thus, machining force model become $F=K_pY$, with F and X form a angle of $\alpha+\gamma$, as shown in Figure 8. The 2-DOF dynamic equation of system without considering the damping effect is:

$$\begin{bmatrix} m_{11} & m_{12} \\ m_{21} & m_{22} \end{bmatrix}\begin{Bmatrix} \ddot{X} \\ \ddot{Y} \end{Bmatrix}+\begin{bmatrix} k_{11} & k_{12} \\ k_{21} & k_{22} \end{bmatrix}\begin{Bmatrix} X \\ Y \end{Bmatrix}=\begin{bmatrix} 0 & K_p\sin(\alpha+\gamma) \\ 0 & K_p\cos(\alpha+\gamma) \end{bmatrix}\begin{Bmatrix} X \\ Y \end{Bmatrix} \tag{14}$$

A similar model was analyzed by Gasparetto (Gasparetto, 1998) for wood cutting application. General solutions for Eq. (13) are available, if the coefficient matrixes are identified accurately. Since mass matrix $[M]$ is symmetric and positive definite, stiffness matrix $[K]$ is symmetric and semi-positive definite, there exists a matrix $[V]$ which achieves

$$[V]^T[M][V]=[I] \tag{15}$$

$$[V]^T[K][V]=[K_\Lambda]=diag(k_1,\ldots,k_n) \tag{16}$$

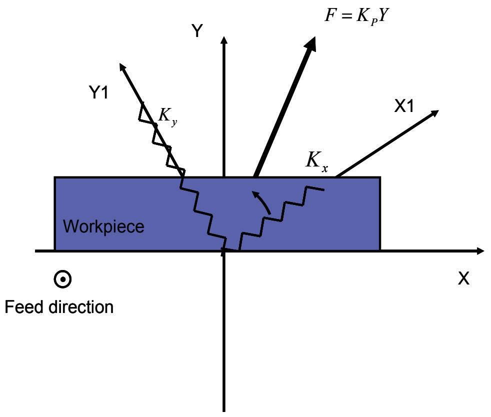


Fig. 8. 2D model of mode coupling chatter system.

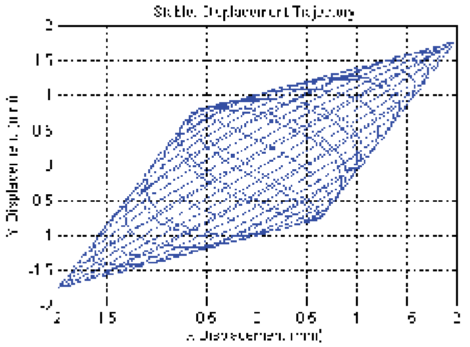


Fig. 9. TCP locus in stable conditionfor undamped system.

By perform this similarity transformation, Eq. (13) becomes

$$\{\ddot{q}\}+[K_\Lambda]\{q\}=[V]^T[K_p][V]\{q\} \tag{17}$$

where

$$\{q\} = [V]^{-1}\{\Delta\} \tag{18}$$

Generally $[V]$ is not an orthogonal matrix; which means the axes of generalized coordinate $\{q\}$ are not perpendicular to each other. The stability of the system depends on the eigenvalues of matrix $[V]^T[K_p][V]-[K_\Lambda]$. If all the eigenvalues of this matrix are negative real number, the system is stable; otherwise, if this matrix has complex eigenvalues, the system will be unstable.

For better explanation of the problem, without loss of generality, we assume that $[M]$ is diagonal without transformation and model mass for each direction is the same

$$[M] = \begin{bmatrix} m & 0 \\ 0 & m \end{bmatrix} \tag{19}$$

In this case, $[V]$ is an orthogonal matrix, that is $[V]^T = [V]^{-1}$, similarity transformation is equal to rotation of the original frame. Thus, the uncoupled principle stiffness directions are perpendicular to each other. In figure 8, X and Y represent frame $\{\Delta\}$; X_1 and Y_1 represent frame $\{q\}$. Cutting process is operated in frame $\{\Delta\}$ with cutting force in X direction and normal force in Y direction (direction of DOC). In frame $\{q\}$, both $[M]$ and $[K]$ are diagonal. The transformation from $\{\Delta\}$ to $\{q\}$ is defined as $\{q\} = [V]^{-1}\{\Delta\}$, where

$$[V]^{-1} = \begin{bmatrix} \cos(\alpha) & \sin(\alpha) \\ -\sin(\alpha) & \cos(\alpha) \end{bmatrix} \tag{20}$$

Then the system equation in the frame $\{q\}$ is:

$$\begin{bmatrix} M & 0 \\ 0 & M \end{bmatrix}\begin{Bmatrix} \ddot{X}_1 \\ \ddot{Y}_1 \end{Bmatrix} + \begin{bmatrix} K_x & 0 \\ 0 & K_y \end{bmatrix}\begin{Bmatrix} X_1 \\ Y_1 \end{Bmatrix} = \begin{bmatrix} K_p\cos(\gamma)\sin(\alpha) & K_p\cos(\gamma)\cos(\alpha) \\ K_p\sin(\gamma)\sin(\alpha) & K_p\sin(\gamma)\cos(\alpha) \end{bmatrix}\cdot\begin{Bmatrix} X_1 \\ Y_1 \end{Bmatrix} \tag{21}$$

Eq. (21) could be rearranged to give:

$$\begin{Bmatrix} \ddot{X}_1 \\ \ddot{Y}_1 \end{Bmatrix} = \begin{bmatrix} \dfrac{K_p\cos(\gamma)\sin(\alpha) - K_x}{M} & \dfrac{K_p\cos(\gamma)\cos(\alpha)}{M} \\ \dfrac{K_p\sin(\gamma)\sin(\alpha)}{M} & \dfrac{K_p\sin(\gamma)\cos(\alpha) - K_y}{M} \end{bmatrix}\cdot\begin{Bmatrix} X_1 \\ Y_1 \end{Bmatrix} \tag{22}$$

Let

$$K_x^{'} = K_x - K_p\cos(\gamma)\sin(\alpha)$$
$$K_y^{'} = K_y - K_p\sin(\gamma)\cos(\alpha)$$

The characteristic equation of Eq. (22) is:

$$\lambda^4 + \frac{K_x^{'} + K_y^{'}}{M}\cdot\lambda^2 + \frac{K_x^{'}K_y^{'} - \frac{1}{4}K_p^2\sin(2\gamma)\sin(2\alpha)}{M^2} = 0 \tag{23}$$

The solution of Eq. (23) gives:

$$\lambda^2 = \frac{-(K_x^{'} + K_y^{'}) \pm \sqrt{(K_x^{'} - K_y^{'})^2 + K_p^2\sin(2\gamma)\sin(2\alpha)}}{2M} \tag{24}$$

In practice, $K_x, K_y >> K_p$ is always satisfied, otherwise, cutting process could not be executed. Thus $K_x^{'} \approx K_x$ and $K_y^{'} \approx K_y$.

If $(K_x^{'} - K_y^{'})^2 + K_p^2 \sin(2\gamma)\sin(2\alpha) > 0$, then the two λ^2 are real negative numbers. In this case the four eigenvalues of the system located on the imaginary axis, symmetric with respect to real axis. The solution of system is Lissajous curves as shown in Figure 9. The system results stable in a BIBO sense. Moreover, since the system damping always exist, the structure damping of the real system will shift the eigenvalues of the system toward left in the complex plan, therefore giving exponentially decaying modes.

If $(K_x^{'} - K_y^{'})^2 + K_p^2 \sin(2\gamma)\sin(2\alpha) < 0$, then two λ^2 are complex number with negative real part. In this case the four eigenvalues of the system are located symmetrically with respect to the origin of the complex plane, so two of them have positive real part. Therefore, instability occurs in this case. The solution of system is exponential increasing as shown in Figure 10. While the damping effect is considered, the locus of TCP is an ellipse.

Unstable region could be represented as:

$$\sin(2\gamma) < \frac{[K_x - K_y + K_p \sin(\gamma - \alpha)]^2}{-K_p^2 \sin(2\alpha)} \tag{25}$$

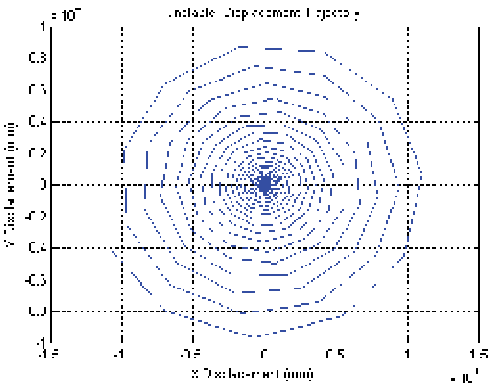


Fig. 10. TCP locus in unstable condition for undamped system.

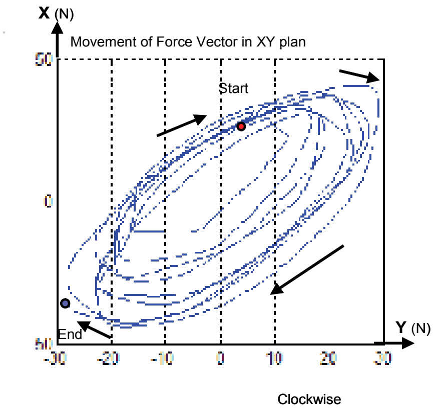


Fig. 11. Locus of force vector while chatter happens.

An important result here is that unstable condition is only possible when $K_p > |K_x^{'} - K_y^{'}|$. That means the mode coupling chatter only occurs when the process stiffness is larger than the difference of two principle stiffness of robot. The physical meaning of the equation gives us the general stability criterion. If a machining system can be modeled by a two degree of freedom mass-spring system, the dynamic motion of the TCP can take an elliptical path. If the axis with the smaller stiffness lies within the angle formed by the total cutting force and the normal to the workpiece surface, energy can be transferred into the machine-tool structure, thus producing mode coupling chatter. The depth of cut for the threshold of stable operation is directly dependent upon the difference between the two principal stiffness values, and chatter tends to occur when the two principal stiffness are close in magnitude.

For CNC machine, the structure stiffness k is on the level of 10^8 N/m, and the process stiffness K_p is usually in the range of $10^5 \sim 10^6$ N/m. As a result, any small difference of k

in each principle directions is much larger than K_p. Before the occurrence of mode coupling chatter, spindle power limit or regenerative chatter limit already reached.
The story is totally different for industrial robot, where stiffness k is on the level of 10^6 N/m and has close magnitude in each direction. Since the process stiffness K_p of machining aluminum workpiece is usually on the level of 10^5 N/m, the mode stiffness of each principle direction could be smaller than process stiffness in certain robot configuration. The mode coupling chatter will then occur even in very light cutting condition. The vibration limit of robotic milling operation depends on the relative angels between machining force vector and the direction of the principle mode stiffness.
The above analysis also coincide with a rule of thumb in the machine tool industry, that the stiffness of two directions should at least has 10% difference to prevent chatter. The characteristics of low frequency chatter summarized in section 3.1 be perfectly explained by mode coupling mechanism:

1. The frequency of mode coupling chatter is the same as the base frequency of the machine. The process parameters such as spindle speed, width-of-cut (WOC), and feed speed won't change the frequency of chatter.
2. In unstable condition, the solution of the system will exponentially increase until it is balanced by damping effects or just crash (Figure 4). The locus of force vector is drawn in Figure 11. The locus of tool tip movement will take the same elliptical path.
3. An unstable cutting process could become stable by only changing the feed direction because the direction of force vector is changed while machine structure keeps the same.
4. Chatter limit of the process is configuration dependent due to the coupled robot structure. The mass matrix and stiffness matrix are not constant; they take different values at different robot configurations. As a result, although the force vector is the same, the chatter limit is different since the machine structure is different.

3.4 Experimental results and analysis

Further experimental verification of the theoretical analysis described in the foregoing subsection was observed in robotic deburing tests. The same type of robot is applied for deburing test of aluminum workpiece as shown in Figure 12; the detailed cutting condition is given in Table 3.

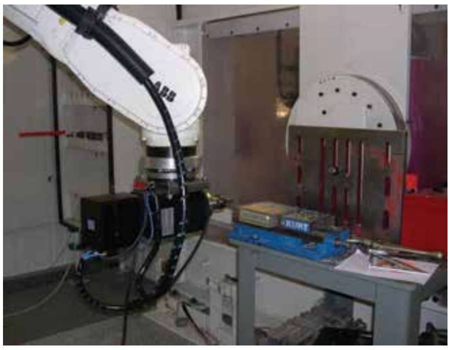


Fig. 12. Setup for robotic deburing.

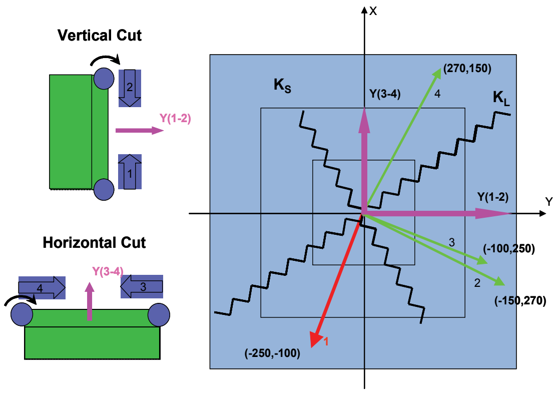


Fig. 13. Stability analysis of deburring.

	Feed direction	Milling type	Force X direction (N)	Force Y direction (N)	Stability
Case 1	+X	Up-milling	-250	-100	Chatter
Case 2	-X	Down-milling	-150	270	Stable
Case 3	-Y	Up-milling	-100	250	Stable
Case 4	+Y	Down-milling	270	150	Stable

Table 3. Summary of deburing test.

In the deburing test, side milling using two-flute helical mill were carried out. Figure 13 presents four machining cases with exactly same cutting parameters except for different feed direction. The cutting conditions and measured machining forces are listed in table 3. Experimental results show that chatter only happens in case 1, while the processes are stable for the rest of the cases. In Figure 13, four force vectors are plotted in the fixed frame X-Y, K_S and K_L represent the smaller and larger principle stiffness of robot calculated from robot structure model, $Y(1-2)$ and $Y(3-4)$ represent the normal direction to the workpiece in vertical and horizontal cutting tests. From the stability criteria established in section four, mode coupling chatter will occur when axis with the smaller stiffness lies within the angle formed by the total cutting force and the normal to the workpiece surface. From Figure 13, stability criteria predict that chatter will only occur in case 1, which perfectly matches the experimental results.
Another important result worth mentioning here is the effect of up-milling and down-milling on mode coupling chatter in robotic machining process. As shown in Figure 14, the direction of cutting force is in a range that is perpendicular to the normal of workpiece in up-milling while the direction of cutting force is almost the same as normal of workpiece in down-milling. Thus, it is more likely for axis with the smaller stiffness to lie between the force vector and the workpiece normal direction during up-milling than down-milling. As a result, from chatter point of view, down-milling is preferred for robotic machining process.

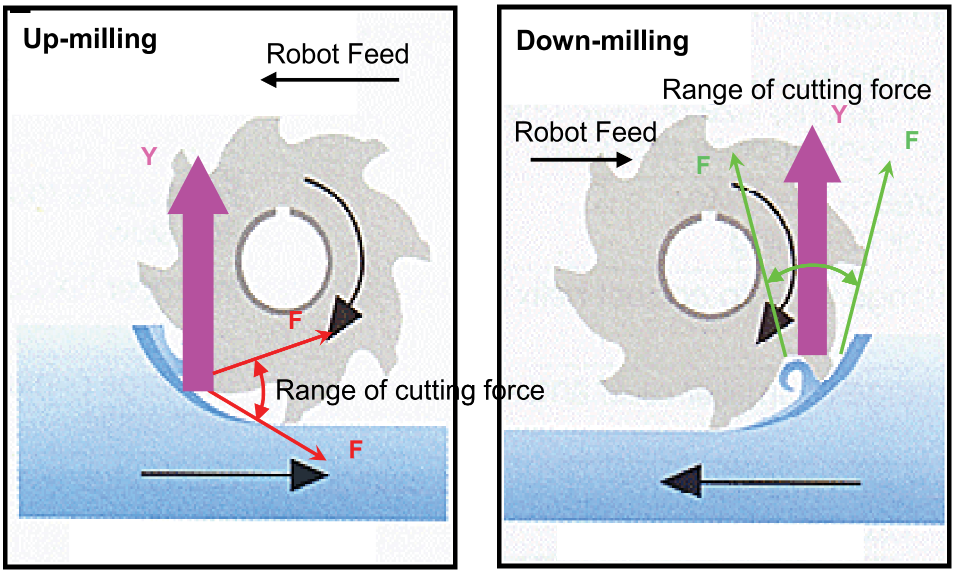


Fig. 14. Up-milling vs. down-milling in mode coupling chatter analysis.

After investing the intrinsic mechanism of low frequency chatter, practical guidelines for chatter-free robotic machining process is summarized here.

1. Select the proper cutting tool. The tool or inserts with different geometry will distribute machining force to different directions. Round insert always generates larger normal force (DOC direction) compared to square insert with zero degree lead angle. Since DOC is the most sensitive parameter related to machining force, chatter may arise more easily if the process has larger normal force. Thus Round insert is not recommended for robotic machining although it has many advantages and is widely using by CNC machine.
2. Plan the proper work space. Since the robot displays different mechanical properties, which are mass, stiffness and damping, at different locations in the workspace, selecting a proper machining location that will have higher chatter limit.
3. Plan the proper path and feed direction. Changing the feed direction is the easiest way to re-direct machining force without affecting the existing setup. Based on the theoretical analysis and experimental results, this is an effective method to avoid mode coupling chatter in many situations.
4. Chatter is more likely to occur in up-milling rather than in down-milling in robotic machining process.

4. Deformation Compensation

Field tests using industrial robots for heavy machining such as milling often found that a perfect robot program without considering contact and deformation fails to produce the desired path once the robot starts to execute the machining task. While thermal induced error is the largest error component for CNC machining, motion error contributes most of the total machining errors in robots. For example, a 500N cutting force during a milling process will cause a 1 mm position error for a robot instead of a less than 0.01mm error for a CNC machine. The major motion error sources in robotic machining process can be classified into two categories, (1) Machining force induced error, and (2) robot inherent motion error (kinematic and dynamic errors, etc.). The inherent motion error, typically in the range of 0.1 mm, is resulted from the robot position controller and would appear even in non-contact cases. While the machining force in the milling process is typically over several hundreds of Newton, the force-induced error, which could easily go up to 1 mm, is the dominant factor of surface error. In order to achieve higher dimensional accuracy in robotic machining, the deformation due to the interactive force must be accurately estimated and compensated in real time.

The existing research of robot deformation compensation is focused on gravity compensation, deflection compensation of large flexible manipulators, etc. Not much attention has been paid to the compensation of process force induced robot deformation due to the lack of understanding and model of robot structure stiffness, the lack of real time force information and limited access to the controller of industrial robot.

4.1 Identification of robot stiffness model

Since force measurement and subsequent compensation is carried out in 3-D Cartesian space, a stiffness model, which relates the force applied at the robot tool tip to the deformation of the tool tip in Cartesian space, is crucial to realize deformation compensation. The model should be accurate enough for the prediction of robot structure deformation under arbitrary load conditions. At the same time, it needs to be simple enough for real time implementation. Detailed modelling of all the mechanical components and connections will render a model too complicated for real-time control, and difficult for accurate parameter identification. The stiffness model in section 2 is an ideal candidate for the deformation compensation. Based on this model, the estimated tool tip deformation in Cartesian space subject to force *F* can be written as:

$$\Delta X = J(Q){K_q}^{-1}J(Q)^T \cdot F \tag{26}$$

Experimental determination of joint stiffness parameters is critical in fulfilling real-time position compensation. Since the joint stiffness is an overall effect contributed by motor, joint link, and gear reduction units, it is not realistic to identify the stiffness parameter of each joint directly by dissembling the robot. The practical method is to measure it in Cartesian space.

To be able to measure small deformations in 3-D space, the end-effector is equipped with a sphere-tip tool shown in figure 15. The tool tip is set to a fixed point in the workspace, and the manipulator joint values are recorded. A given load in the range of 100N~400N is applied to the tool, causing the sphere-tip to move away from the original point. The original and deformed positions are measured with ROMER, a portable CMM 3-D digitizer, and the 3-DOF translational deformations are calculated. From Eq. (26), K_q could be solved by least square method. The same experiment was repeated at several different locations in the robot workspace. As can be seen from figure 16, the deviation of the results at different test locations is small, which means a set of constant parameters could model the robot deformation with small error.

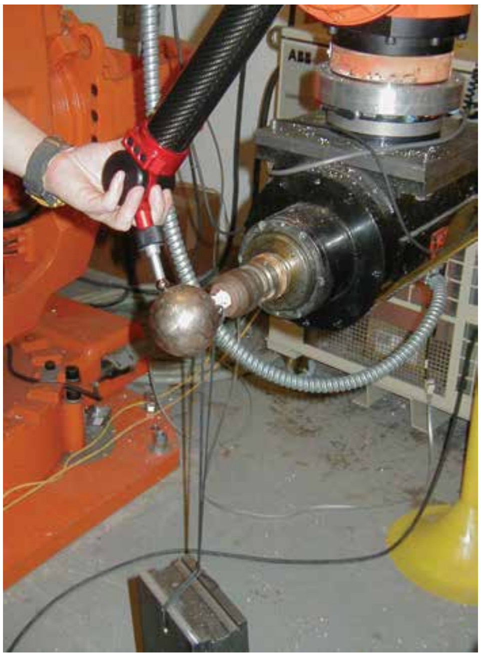


Fig. 15. Experiment Setup for robot stiffness measurement.

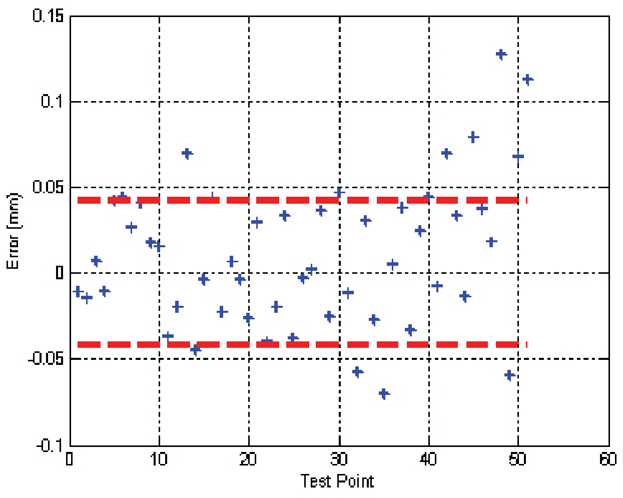


Fig. 16. Error of stiffness modeling.

4.2 Real-time deformation compensation

Figure 17 shows the block diagram of real time deformation compensation. After filtering the force sensor noise and compensating the gravity of the spindle and the tool, the force signal was translated into the robot tool frame. Based on the stiffness model identified before, the deformation due to machining force is calculated in real time and the joint reference for the robot controller is updated.

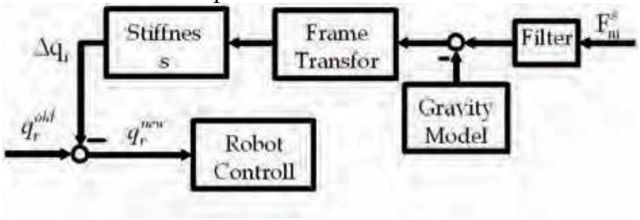


Fig. 17. Principle of real-time deformation compensation.

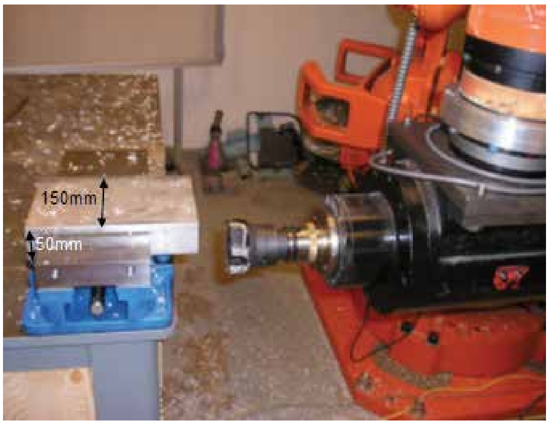


Fig. 18. Experimental setup for robotic milling.

4.3 Experimental results

A robotic milling cell, where an ABB IRB6400 robot holds a spindle as shown in figure 18, is used for deformation compensation test. For illustration purpose, the workpiece is chosen as a 6063 aluminum block. A laser displacement sensor is used to measure the finished surface. The surface error without deformation compensation demonstrates counter-intuitive results; an extra 0.5mm was removed in the middle of the milling path. Conventional wisdom says that a flexible machine would also cut less material due to deformation, since the normal force during cutting will always push the cutter away from the surface and cause negative surface error. However, in the articulated robot structure, the deformation is also determined by the structure Jacobian, in a lot of cases, a less stiff robot could end up cutting more material than programmed. The coupling of the robot stiffness model explains this phenomenon, the force in feed direction and cutting direction will result in positive surface error in that robot configuration. Since the feed force and the cutting force are the major components in this setup, the overall effect will cut the surface 0.5 mm more than the commanded depth. In our definition, negative surface error means less material was removed than the commanded position.

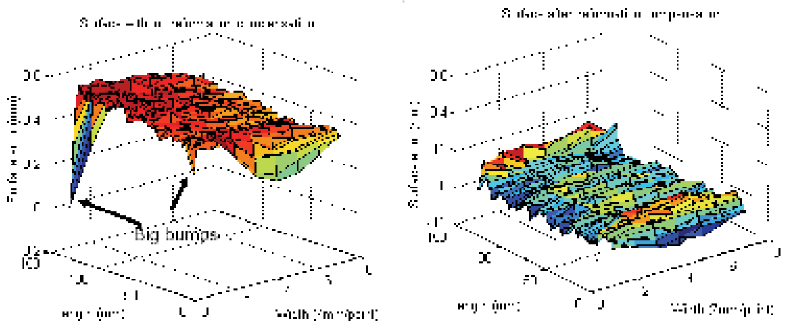


Fig. 19. Deformation compensation results

The result after deformation compensation shows a less than 0.1 mm surface error, which is in the range of robot path accuracy (Figure 19). Further test conducted on the foundry cylinder head workpiece shows that the surface accuracy improved from 0.9mm to 0.3mm, which is below the 0.5mm target accuracy for pre-machining application.

5. Controlled Material Removal Rate

In pre-machining processes, maximum material removal rates are even more important than precision and surface finish for process efficiency. MRR is a measurement of how fast material is removed from a workpiece; it can be calculated by multiplying the cross-sectional area (width of cut times depth of cut) by the linear feed speed of the tool:

$$MRR = w \cdot d \cdot f \tag{27}$$

Where w is width of cut (mm), d is depth of cut (mm), f is feed speed (mm/s).

Conventionally, feed speed is kept constant in spite of the variation of depth of cut and width of cut during foundry part pre-machining process. Since most foundry parts have irregular shapes and uneven depth of cut, this will introduce a dramatic change of MRR, which would result in a very conservative selection of machining parameters to avoid tool breakage and spindle stall. The concept of MRR control is to dynamically adjust the feed speed to keep MRR constant during the whole machining process. As a result, a much faster feed speed, instead of a conservative feed speed based on maximal depth of cut and width of cut position, could be adopted (Figure 20).

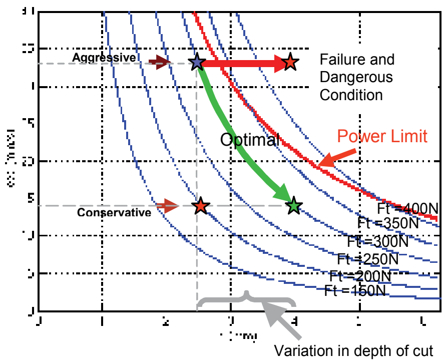


Fig. 20. Controlled material removal rate.

Since the value of MRR is difficult to measure, the MRR is controlled by regulating the cutting force, which is readily available in real-time from a 6-DOF strain gage force sensor fixed on the robot wrist. Placing the analysis of the material removal process on a quantitative basis, the characterization of cutting force is important for research and development into the modeling, optimization monitoring and control of metal cutting.
The challenges for designing a robust controller for MRR is the fact that cutting process model varies to a large degree depending on the cutting conditions. Efforts for designing an adaptive controller will be presented in a separate paper.
As the feed speed *f* is adjusted to regulate the machining force, MRR could be controlled under a specific spindle power limit avoiding tool damage and spindle stall. Also, controlled MRR means predictable tool life, which is very important in manufacturing automation. Figure 21 shows the block scheme of machining force control with controlled material removal rate (CMRR). For the force process model represented in Eq.(7), with the proper selection of reference feed speed f_r and reference force F_r, a PI controller is adopted to regulate the cutting force F_c, while force process gain θ changes.

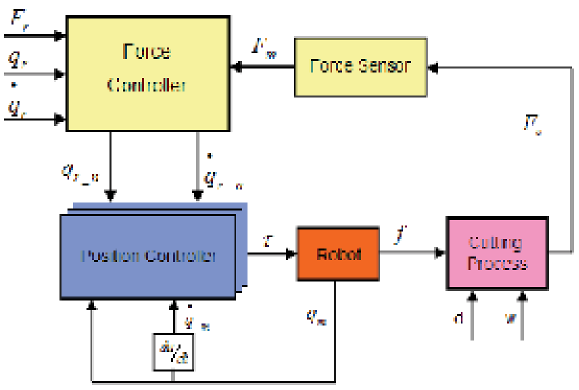


Fig. 21. Force control for robotic machining.

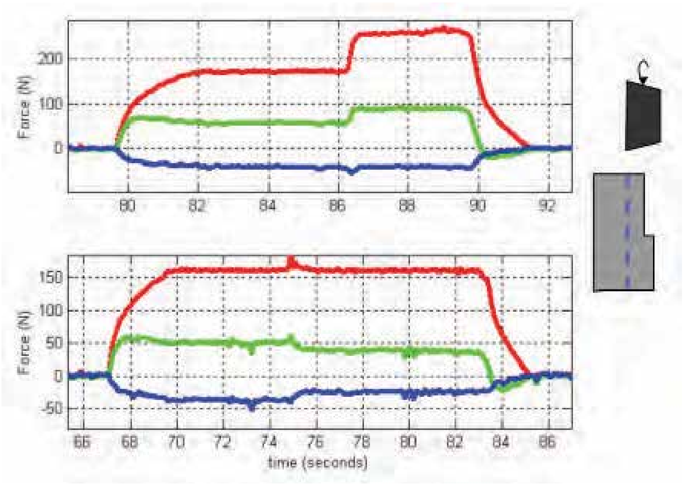


Fig. 22. Force control result of variant depth of cut.

The same workcell shown in figure 18 is used for MRR test. A 6063 aluminum block is intentionally machined to look like in Figure 22 to simulate the variation of cut depth. Tests on an aluminum block with the depth of cut changed from 2 mm to 3 mm shows, when force control is activated, the cutting force is regulated in spite of the variance of depth of cut (Figure 22). The milling test of aluminum with variation of width of cut shows similar results.
As a result, the feed speed could always be setup as fast as the limit of spindle power. In a foundry parts milling or deburring process, the robot won't have to move at a very conservative speed to avoid tool breakage or spindle stall. The cycle time decreased by CMMR is typically around 30% to 50% for different workpieces.

6. Summary and Conclusion

This chapter presents critical issues and methodologies to improve robotic machining performance with flexile industrial robots. The problems under the investigation are low frequency chatter, machining force induced robot deformation and conservative feed rate for non-uniform cutting depth and width, which are often the causes of unacceptable quality, lower productivity, and process instability. For low frequency chatter, it was found that mode-coupling chatter is the dominant source, largely due to the inherent low structure stiffness of industrial robot. The occurrence of mode coupling chatter depends on the relative orientation of the machining force vector and the principle stiffness axes of the robot. Methods such as changing the feed direction, using different robot configuration or changing another type of tool are recommended to avoid chatter occurrence. To address deformation problem, stiffness modeling and real time deformation compensation were adopted for robot control. A constant joint stiffness model, which accounts for the dominant contribution of joint compliance to the tool center point (TCP) deflection, was chosen to minimize the computation time. Test results showed 0.5mm surface error improvement after the compensation. To improve the efficiency in machining parts with uneven cut depth and width, a PI control strategy was proposed to adjust the cutting feed rate based on the measured cutting force. The feed rate was optimized in real time to compensate process variations to help maximize material removal rate (MRR) and minimize cycle time. Practical machining experiments conducted in the lab showed great reduction of cycle time, as well as better surface accuracy. These results outline a promising and practical use of industrial robots for machining applications that is not possible at present.

7. References

Akbari, A. & Higuchi, S. (2000). Autonomous tool adjustment in robotic grinding, *The International conference on Precision Engineering*(ICoPE) ,pp.121-126

Altintas, Y. (2000). *Manufacturing Automation: Metal Cutting Mechanics, Machine Tool Vibrations, and CNC Design*, 1st ed., Cambridge University Press, NY, USA

Budak, E. & Altintas, Y. (1998). Analytical prediction of chatter stability conditions for multi-degree of systems in milling. Part I: modeling, Part II: applications. *Transactions of ASME, Journal of Dynamic Systems, Measurement and Control*, Vol.120, pp.22-36

Chen, Y. & Hu, Y. (1999). Implementation of a robot system for sculptured surface cutting. Part 1. rough machining. *International Journal of Advanced Manufacturing Technology*, Vol. 15, pp. 624-629

Daneshmend, L. & Pak, H. (1986). Model reference adaptive control of feed force in turning, *ASME Journal of Dynamic Systems, Measurement, and Control*, Vol. 108, No. 3, pp. 215-222.

Gasparetto, A. (1998), A system theory approach to mode coupling chatter in machining, *ASME Journal of Dynamic Systems, Measurement, and Control,* Vol. 120, pp.545-547.

Kim, S.; Landers, R. & Ulsoy, G. (2003). Robust machining force control with process compensation. *Journal of Manufacturing science and engineering*, Vol. 125, pp. 423-430

Matsuoka, S., Shimizu, K., Yamazaki, N. & Oki, Y. (1999). High-speed end milling of an articulated robot and its characteristics, *Journal of Materials Processing Technology*, Vol. 95, No. 1-3, pp. 83-89

Merritt, H. (1965). Theory of self-excited machine tool chatter: contribution to machine-tool chatter research-1, *ASME Journal of Engineering for Industry*, Vol. 87, No. 4, pp. 447-454

Pan, Z. ; Zhang, H. ; Zhu, Z. ; Wang, J. (2006). Chatter analysis of robotic machining process, *Journal of Materials Processing Technology*, Vol.173, pp.301-309

Sallinen, M. & Heikkilä, T. (2000). Flexible workobject localisation for CAD-based robotics, *Proceedings of SPIE Intelligent Robots and Computer Vision XIX: Algorithms, Techniques, and Active Vision*, Vol.4197, pp. 130 – 139, Boston, USA, Nov. 2000

Stein, J. & Huh, K. (2002). Monitoring cutting forces in turning: a model-based approach. *Journal of Manufacturing science and engineering*, Vol. 124, pp. 26-31

Tlusty, J. & Polacek, M. (1963). The stability of machine tools against self excited vibrations in machining, *International Research in Production Engineering, ASME*, pp. 465-474.

Tobias, S., & Fishwick, W. (1958). The chatter of lath tools under orthogonal cutting conditions, *The Transaction of the ASME*, No. 80, pp. 1079–1088

Wang, J. ; Sun, Y. & et. al. (2003a). Process modeling of flexible robotic grinding, *International Conference on Control, Automation and Systems,* Gyeongju, Korea, Oct. 2003

Wang, J. ; Sun, Y. & et. al. (2003b). In-process relative robot workcell calibration, *International Conference on Control, Automation and Systems,* Gyeongju, Korea, Oct. 2003

Whitney, D. (1977). Force feedback control of manipulator fine motions, *ASME Journal of Dynamic Systems, Measurement, and Control*, Vol. 99, No. 2, pp. 91–97

Yang, S. (1996). Real-time compensation for geometric, thermal, and cutting force induced errors in machine tools. *Ph.D. dissertation*, The University of Michigan

Zhang, H. & et. al., (2005). Machining with flexible manipulator: toward improving robotic machining performance, *IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM)*, California, USA, July 2005.

Welding Robot Applications in Shipbuilding Industry: Off-Line Programming, Virtual Reality Simulation, and Open Architecture

Chang-Sei Kim[2], Keum-Shik Hong[2], Yong-Sub Han[1]
[1]*Daewoo Shipbuilding and Marine Engineering Ltd, South Korea*
[2]*Pusan National University, South Korea*

1. Introduction

The shipbuilding industry is steadily advancing by introducing robots to its work fields for increases in productivity and improvements in working conditions (Nagao et al, 2000). However, the shipbuilding company still faces with the worker's health problem, an increase of aging workers, a shortage of skilled workers, and environmental protection related issues. Therefore, advanced robotic manipulator is still required to overcome these problems. And, how to apply commercial robotic system properly to meet the production purpose in the shipyard is a key research topic for shipbuilding engineers.
The shipbuilding process is mainly divided into design, cutting, welding, assembling, grinding, blinding, and painting process. Among these manufacturing processes, welding is the most crucial, expensive, and time-consuming process. For that reason, welding robot applications have yielded a big productivity improvement in hull assembly welding and have reduced work-related musculoskeletal disorders of workers.
In this chapter, the results of our work in the development of a welding robot system and a PC-based off-line programming for welding robot application on assembly lines in shipbuilding are explained. Also, a methodology of implementing PC-based off-line programming on users PC is presented. The off-line programming is a system that comprises robot's simulation, robot programming, and other functions such as monitoring, scheduling, etc, that makes users operate robot system easily. The off-line programming is essential for welding robot system in shipyard to prepare robot program and then to shorten production time.
Currently, the operation of industrial robots is through either on-line teaching or off-line programming (Choi & Lee, 2003; Carvalho et al., 1998; Craig, 1986;). On-line teaching is, by definition, a technique of generating robot programs using a real robot system, whereas off-line programming is a method using simulations that are set up in advance. On-line teaching may be suitable for jobs for which a robot only needs to repeat a monotonous motion using one pre-written program that applies to identical sizes or objects. But, in such work places as shipbuilding, where the shape and size of workpiece are various (i.e., there are more than 1200 different shapes of workpieces for grand-assembly, if we account the size of these different shaped workpiece, we may not count the different kind of workpieces. Moreover, the new shape of workpiece is still increasing according to the ship specification

advancing), on-line teaching will cause problems due not only to the decrease of productivity caused by halting the robots while reprogramming, but, more importantly, to not being able to revise work errors that on-line programming itself can cause. Hence, the more profitable method of building a work program is using off-line programming, while only applying programs that were already verified to be effective for the job. The advantages of using off-line programming are: (1) the effective programming of robot-command logic with debugging facilities; (2) the easy verification of the validity of the robot programs through simulation and visualization; (3) organized documentation through simulation models with appropriate programs; (4) the reuse of existing robot programs and easy application to other objects; and (5) the cost independence of production owing to the fact that production can be continued while programming. Fig. 1 shows difference between two robot programming.
Research on the robotic system simulation is prevalent (Ferretti, 1999; Kreuzer & Miojevic, 1998; Richard, 1997). And the industrial robot production enterprises provide commercialized software for robot simulations such as ROBCAD, IGRIP, and etc., which include developed simulation tools for the robots. However, applying this commercialized software to ship construction requires extra preparation, including users becoming fluent with CAD systems, the complete modeling of the work object, and the development of language translators that will work with robot manufacturing

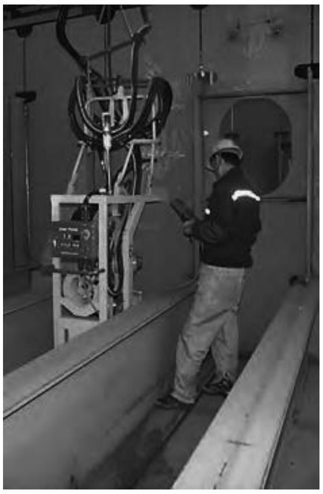
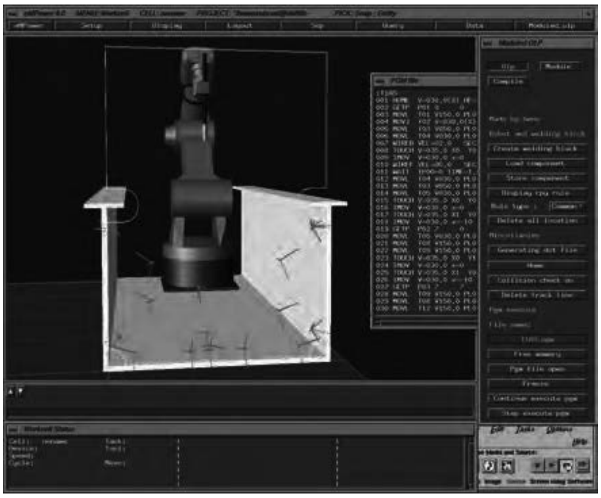



Fig. 1. On-line teaching (Left) and Off-line programming (Right)

companies. In short, because it takes too much time and effort, the utilization of commercial software for robot systems is not suitable. Instead, because of high expectations for computer systems and the rapid development in graphic interfaces, nowadays, establishing a PC-based simulation environment has become easier and has come to be preferred. Therefore, using off-line programming for robot systems is suitable for work in a shipbuilding yard because it is more economical than commercial software that is provided by robot companies.
This chapter is structured as follows: In Section 2, a welding workpiece and a robot system in hull assembly line is explained. In Section 3, the off-line programming and the methodology of robot simulation based on PC is discussed. Section 4 presents the

application of the CAD interface and the robot program generation to the actual welding plant. In Section 5, the process of path planning for scattering the blocks based on a block arrangement simulation is explained. Section 6 provides conclusions.

2. Welding Robot System Configuration

Hull assembly is separated to sub-assembly, middle-assembly and grand-assembly by its size and production sequence. One of the critical problems for applying robots to shipbuilding is transferring method to locate robots in a proper position. There are some researches about the shipbuilding process, where the positioning system is a key technology for effective introduction of robots because the workpiece is diverse and complex (Ogasawara et al., 1998; Nagao at al., 2000; Mori at al., 2001).
Currently, there are three methods of transferring a welding robot to a work place: a manual transfer, a semi-automatic transfer and a fully automatic transferring approach. The manual transfer means that the robot is placed on a workpiece by human. The semi-automatic approach, that is good for high assembly blocks, is composed of x-y plane gantry crane, chain-rope, robot cart, and a robot body. Sometimes, robot origin transfer unit is attached to move robot body to its origin position in a workpiece. The semi-automatic type of robot transferring system hangs the robot body at the end of chain-rope. When working, the robot body is lifted up and down by the chain-rope and moves plenary on a workplace by x-y plane gantry crane. Fig. 2 shows a picture of the real grand-assembly welding robot system and a typical example of a workpiece. A 3-axis Cartesian robotic manipulator operates the fully automatic transfer system. The fully automatic transfer system is applied to the sub-assembly lines.
The manual and fully automatic approaches are not preferred in a grand- and middle-assembly process. The reasons are that a manual move consumes too much time and excessive human effort for individual transferences, and that a fully automatic transference causes an unwanted vibration of the gantry, in moving the welding robot from one place to another, which subsequently deteriorates the welding quality. Accordingly, a semi-automatic approach is widely used. However, due to the small size of sub-assembly, the fully automatic transfer approach is applied to the sub-assembly welding process. Recently, new types of robot transferring systems such as mobile type, rail type, and parallel mechanism, are researched for shipbuilding.

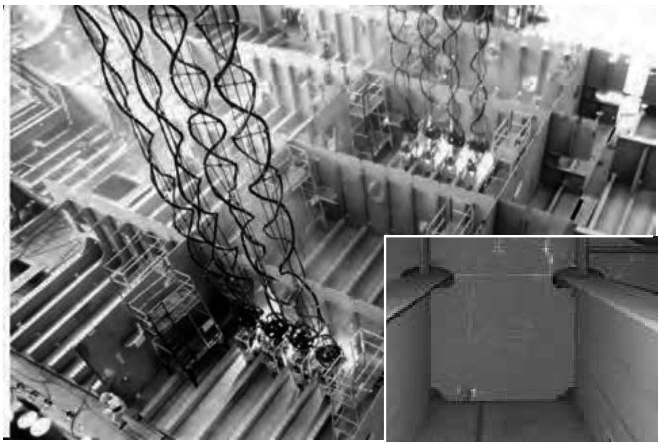


Fig. 2. A grand-assembly and a typical example of a workpiece(Right corner).

Fig. 3 shows the welding robot system that is being used at Daewoo Shipbuilding and Marine Engineering Ltd., South Korea. The system is composed of a welding robot, a controller, a gantry crane, welding equipment, and an on-site industrial computer.

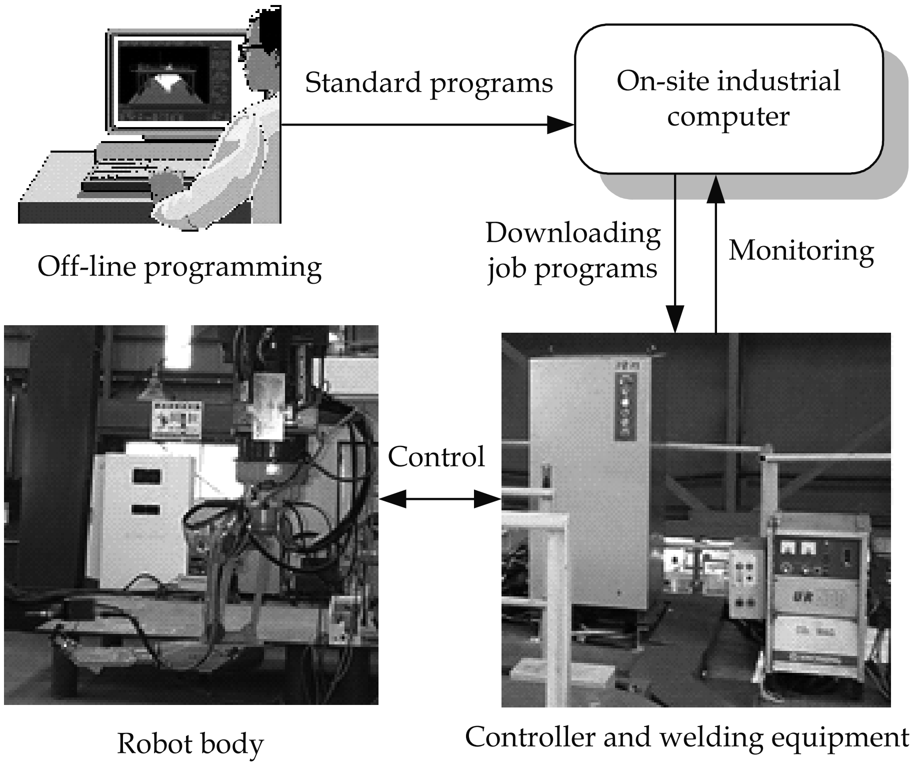


Fig. 3. Configuration of the welding robot system: First, The standard-program is made using the off-line programming. Second, using the CAD interface, the job program which contains real size of a workpiece is made. Third, the job-program is transfered to robot controller. Finally, the robot controller execute the job-program line by line and controls welding robots.

Because the size and shape of workpieces in the shipbuilding industry vary greatly, a 6-axis articulated robot is generally used for welding. The robot controller consists of a Pentium II processor, three motor interface boards and other digital signal processing boards. As one motor interface board can control 4 motors, room for adding more control boards for auxiliary actuators has been reserved. As the controller are designed by open-architecture, regarding the workpiece, auxiliary robotic manipulator to locate robot system or to transfer workpices can cooperated with the robot's controller. The QNX is used as a real-time operating system.

To increase robot's welding ability against various shape of workpice, we define 2 robot programs as a standard-program and a job-program. And, we design the standard-program as a combination of three separate parts: a program-file, an rpy-file, and a rule-file. Rpy represents the roll, pitch, and yaw values of the orientation of the tool. The program-file describes the sequence of robot motions that is written in robot language. The rule-file contains the tool position values of the teaching points indicated by the program file. And the rpy-file contains the orientation values of the teaching points.

```
;FR65-74sl.pgm
  001 RHOME          V=030.0(%) HF=1
  002 GETP           P01 0    0    0    CRD=BASE
  003 GMOVJ          T01 V=030.0(%) PL=0
  004 RHOME          V=030.0(%) HF=2
  005 RMOVJ          T02 V=030.0(%) PL=0
  006 GMOVJ          T03 V=030.0(%) PL=0
  007 GMOVJ          T04 V=030.0(%) PL=0
  008 GETP           P02 0    0    0    CRD=BASE
  009 RMOVL          T05 V080.0 PL0 D0
  010 RTOUCH         V=035.0 X1  Y0  Z0  L100 P00
  011 RIMOV          V=030.0 x=-10  y=0    z=0    CRD=BASE
  012 RTOUCH         V=035.0 X0  Y0  Z-1 L100 P00
```

```
; FR65-74sl.rpy
  T02 =   180     0     0     1     1
  T03 =  -180   -47   180     1     1
  T04 =   180     0     0     1     1
  T05 =    90     0     0     1     1
  T06 =   180   -47    90     1     1
```

```
; FR65-74sl.rule
  T02 G52 2064.0   -2389.0   1680.0
  T03 A52 2734.0   -2389.0    44.0   2064.0   -2389.0   1680.0    0.0
  T04 A52 2734.0   -1889.0    44.0   2064.0   -1889.0   1680.0    0.0
  T05 A52 2734.0   -2889.0    44.0   2064.0   -2889.0   1680.0    0.0
  T06 G52 2177.6   -2344.2   1680.0
```

Fig. 4. An example of the standard program consisting of three parts: a program-file (top), a rpy-file (middle), and a rule-file (bottom). In the program-file, the second column is robot command and the third column is inner paramters of each robot command.

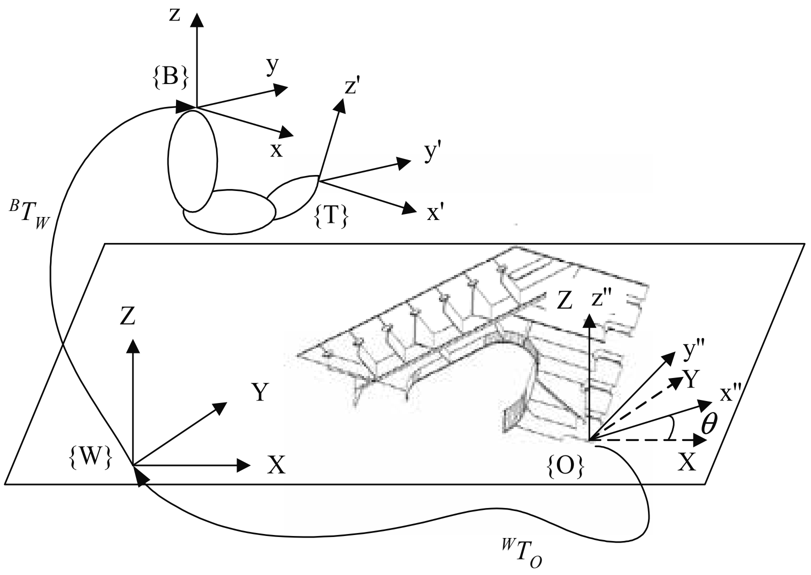


Fig. 5. The coordinate systems defined: world, base, object, and tool.

In order to apply variously sized but identically shaped target objects without modifying the pre-generated robot program, the teaching points in the standard program are written in a variable form. Fig. 4 shows an example of a standard-program. Also, in order to be executed in the robot controller, information regarding the real size of the workpiece and the welding conditions should be contained in the standard-program. The standard-program including this information is referred to a job-program. The size information is obtained by CAD interface and the welding conditions are gathered from a database in which all previous experimental data have been stored.
Fig. 5 depicts the defined coordinate systems: {W} denotes the fixed world (reference) coordinate system attached to the ground; {B} denotes the base coordinate system affixed to the base of the robot, whose position will be determined by the movement of the gantry; {O} refers to the object coordinate system attached to the workpiece; and {T} represents the coordinate system attached to the tool center.

3. Off-Line Programming and Robot Simulation

3.1 Off-line programming for shipbuilding

The developed off-line programming system is composed of CAD interface, robot simulation, automatic robot program generation, block arrangement simulation, and a path planning using Genetic algorithm for fully automated welding robot system of a shipbuilding. Fig. 6 shows a flow chart of the developed off-line programming system. First, if the workpiece is identical to an old one, then an appropriate program is automatically loaded from the D/B. If the workpiece is not the same as old one, the off-line program generates robot's new programs automatically. Second, if the robot programs are new one, then we check it by simulation. Simulations using off-line programming are particularly effective when creating a robot program for movements on a critical surface, whereas an automatic program is used in reference to objects with

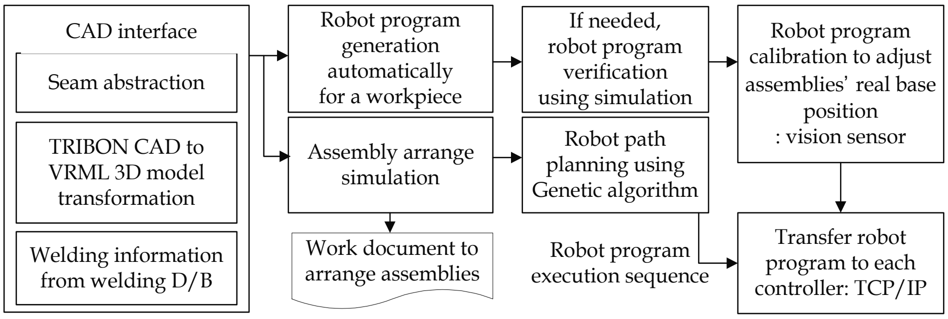

CAD interface
Seam abstraction
TRIBON CAD to VRML 3D model transformation
Welding information from welding D/B
Robot program generation automatically for a workpiece
Assembly arrange simulation
Work document to arrange assemblies
If needed, robot program verification using simulation
Robot path planning using Genetic algorithm
Robot program execution sequence
Robot program calibration to adjust assemblies' real base position : vision sensor
Transfer robot program to each controller: TCP/IP

Fig. 6. Flow chart of the off-line programming.

pre-determined surfaces. Third, depending on the work schedule, off-line programming provides a work-order document of a workplace arrangement for the blocks. Fourth, after arranging the blocks according to the document, a vision sensor verifies and revises the positions of the blocks. Finally, the off-line program sends the generated robot programs to the each robot's controller using TCP/IP communication. For the grand- and middle-assembly line, because the robot body is located on a workpiece by semi-automatic method, and the robot base position is not changing until robot finishes a welding of a workpiece, the vision sensor and block arranging to get information of workipiece base position is not required.

3.2 Robot body and workpiece modelling using VRML

Using the virtual reality modeling language (VRML), various simulations such as robot motion, block arrangement, optimal trajectory planning, and CAD interfaces were performed on a Personal Computer (PC). The VRML utilized is a 3D graphic language, which expresses objects and their motions in a space of proper dimensions, and which is useful in constructing a virtual environment on a PC (Ferretti et al., 1999; Kunii & Shinagawa, 1992). Fig. 7 shows examples of VRML models.

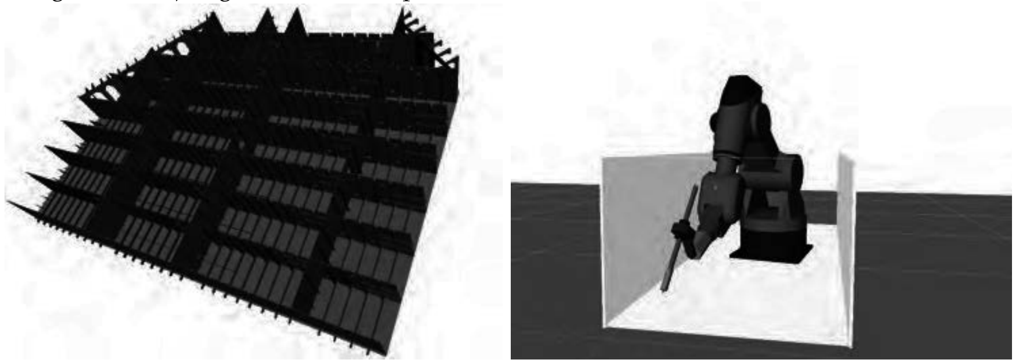


Fig. 7. Examples of VRML 3D models(Left: a grand-assembly model converted from CAD data using developed converting software; Right: a welding robot model drawn by manually using VRML).

The salient features of the VRML are as follows: first, it is easier to interpret because the text is expressed with a grammatical structure of functions; second, one can easily obtain VRML models from other CAD software, because converting a CAD drawing to a VRML model is possible in most 3D CAD software; third, the VRML model contains vertex data which makes it easy to extract useful specific point information; fourth, because the 3D objects are modeled by VRML, which can be shown on the Internet, off-line programming through the Internet is an option.

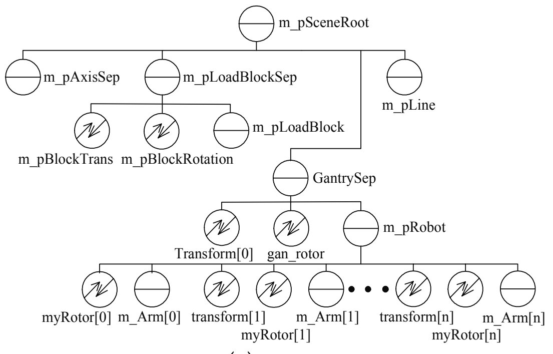
3.3 Kinematics and robot motion structure

To implement robot's 3D solid models and motions on a PC, a structured graphical representation of the nodes, illustrated as in Fig. 8(a), is required (Hoffmann, 1989; TECHNOMATIX, 1998; Wernecke, 1994). Also, Fig. 8(b) shows the 3D robot body and each link model. Each link can be replaced with a newly designed link without influencing other graphic objects. The 3D solid model of each link is defined as *m_Arm*[*n*], whereas each link's motion engine is defined as *myRotor*[*n*], where *n* represents the *n*-th link. The *m_Arm*[*n*] is a variable name that stores the *n*-th link model, and the *myRotor*[*n*] is the variable name of the motion engine. A translation variable *transform*[*n*] is used to place each link on a desired position, i.e. end of the *n*-1th link. The basic composition is a parallel combination of the translation variable, motion engine and the link model of each link. Accordingly, the motion of the $(i+n)$th link, where $n=1,2,3,...$, is affected by the movement of the i-th link. For 6-axis welding robot, as the robot body is composed of 6 rotation links and the value of *myRotor*[*n*] means each link's angle, the robot simulation can be done by changing *myRotor*[*n*] value using robot's forward kinematics and inverse kinematics. The *m_pLoadBlockSep* is added to this node structure to draw workpieces. Usually, There are 2 kinds of simulation option is required for shipbuilding as below:

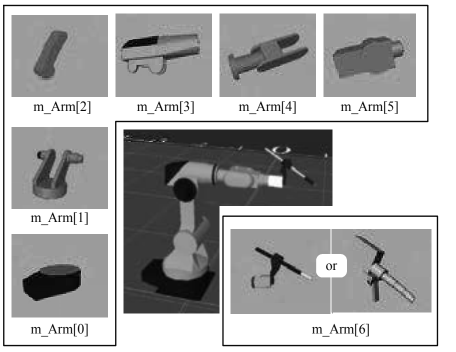
Case 1: Grand- and Middle-assembly
- Because the workpiece base position is fixed,
- Robot base position is also fixed.

Case 2: Sub-assembly
- Because the workpiece base position is not fixed (variable),
- Robot base position is also movable (variables).

Auxiliary graphic objects such as axes, texts, and welding lines are added to the top-node defined as *m_pSceneRoot* directly, in order to be independent of robot's movements. For example, to display the axis of the teaching points and the welding line, *m_pAxisSep* and *m_pLine* are attached to the *m_pSceneRoot* node independently of the motion of the related nodes the *m_pLoadBlockSep* and the *GantrySep*, as shown in Fig. 8. The axis graphic node is added to the *m_pSceneRoot* node, whenever the user generates new teaching points. The added axis graphic node is counted and the entire axis in the simulation window has its own number. By clicking the axis in the simulation window, the simulation window displays the data of the selected teaching point. In the same way, the *m_pLine* containing the line graphic object is added to the *m_pSceneRoot* node. And, whenever the user selects a welding line, the line is displayed in the simulation window. In order to render and change link parameter easily for a similar type robot, a robot initialization file is used. The robot initialization file contains the file name of the robot body VRML model, the link parameters, the limit values of individual joints, the control parameter and the home position data. In addition, the user can arbitrarily specify the robot base position so that the initial position of the robot system can be easily set. Also, through the manipulation of kinematics data, the base coordinate frame can be easily placed at a desired position. Hence, the reachability of the end-effecter and possible collisions with the surrounding parts can be easily examined through the simulations in a virtual environment.



(a)



(b)

Fig. 8. Simulation environment construction: (a) a hierarchical representation of graphic nodes to realize simulation environment; (b) 3D robot body and individual link models.

Two types of simulation modes are provided: a teaching mode and an execution mode. Robot teaching tells the robot what to do. Because the operator can easily move the robot in various motions with via-points using the teaching mode, this mode is very helpful for operators to change teaching point on simulation window. The teaching mode includes two jog functions: a joint jog function that moves the joint actuators in relation to the joint coordinates, and a coordinate jog function that moves the robot according to a given coordinate frame, as shown in Fig. 5. In the program execution mode, robot commands in a standard program are automatically, simultaneously, and continuously executed line by line. Left picture of Fig. 9 shows the simulation of a grand-assembly, while right picture shows the simulation of a sub-assembly for which the robot is the hanging-type.

The simulated motions approach to the real ones, because the algorithms of the simulation program including kinematics, the robot motion planning and the robot language interpreter are identical to those of the real controller's. Because the control input sampling time is 16 msec whereas the interpolation time of a robot motion is 5 msec, the robot motion is updated every 16 msec in simulations. Also, multi-threads called for by a 16 msec timer are used for the multi-robot simulation. In this case, one thread can initiate some of the functions shared with other threads, such as the robot language interpret function, the motion planning function and the starting command function, at the same time, and that results in memory leakage or malfunction. So the multi-threads and *CCriticalSection* of VC++ work together.

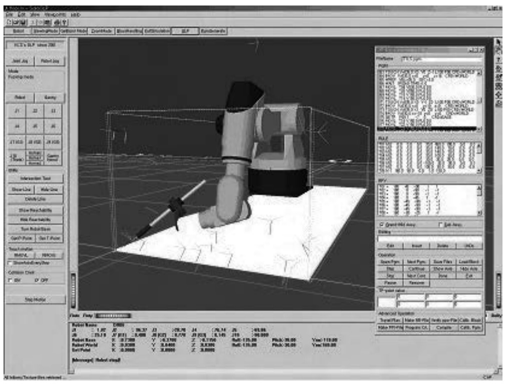
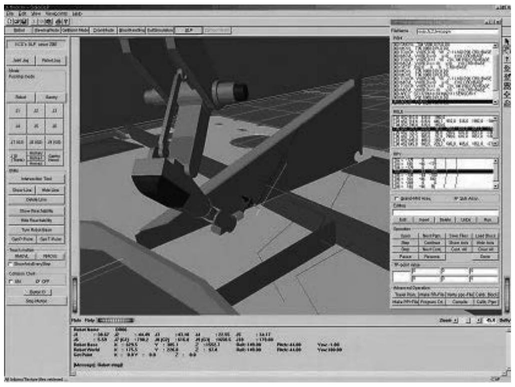



Fig. 9. Simulation example of a welding robot system(Left: for a grand- and middle-assembly, Right: sub-assembly).

4. CAD Interface and Automatic Robot Program Generation

4.1 Tribon CAD interface

The geometric modeling of robotic systems plays an important role in off-line programming. A good geometric model of robots, obstacles, and the objects manipulated by the robots is important to task planning and path planning. As auxiliary 3D modeling for robot simulations is time-consuming and painstaking work, a CAD interface is essential to the robot system.

In developed off-line programming, 3D geometric models of robot simulations are acquired from a TRIBON CAD interface. TRIBON CAD is commercial software used as a design tool of shipbuilding. The output of the CAD interface is a 3D model of the workpiece that is converted into the VRML. Additionally, the welding information, which contains the position value of the tool center point (TCP) and the orientation of the tool on the workpiece, are also obtained by the CAD interface. Fig. 10 shows a simple example of a set of welding information file from CAD interface. A welding information file also contains the welding condition that is specified by class and welding standards. Also it contains general block information such as the number of welding passes, base coordinate definition, block names, and block sizes.

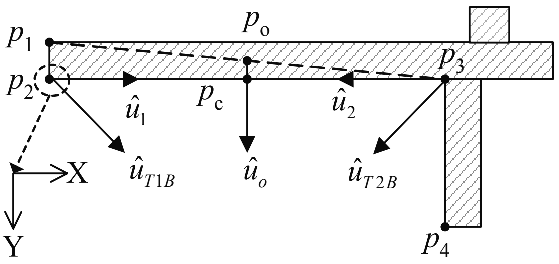
By using the boundary representation (BR) method (Sheu & Xue, 1993), the surface of a solid object is segmented into faces, and each face is modeled by its bounding edges and vertices. Also, the edges and vertices of an object can be extracted from the TRIBON CAD models by the BR method. Therefore, the 3D drawing of TRIBON CAD is decomposed into edges and vertices. The edges and vertices are reconstructed into VRML models. The function *IndexedFaceSet* is a VRML command to render a 3D object from edges and vertices. Using the extracted edges and vertices, all of the elements of the workpiece comprise solid models. Next, these elements are assembled together on the plates according to the assembly's sequence of drawings. This method is similar to the real assembly process in such a way that the operator handles all of the elements in the space and measures the size of each element using simulations, as in a real system.

```
BEGIN_INFO
 'S6G9';
 BEGIN_JOINT
   J-1; 527-FR86A-1;527-FR86A-S1;15.5;18.0;
   BEGIN_WELD
   W-1;H;0.707,-0.036,-0.706;525;
     BEGIN_SEGMENT
     1836,-25,314; 0,0,0 1836,500,314;
     END_SEGMENT
   END_WELD
 END_JOINT
END_INFO
```

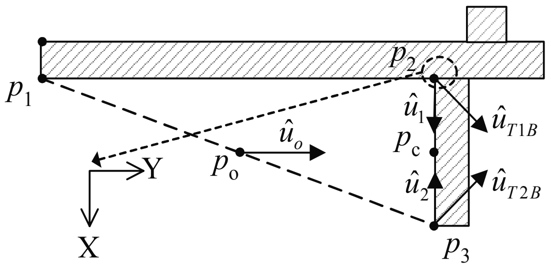
Fig. 10. An example of welding information generated from the CAD interface.

Using the CAD interface, the operator can select the robot weld-able workpieces and extracts the welding information of the selected workpieces on operator's PC. For the case of grand- and middle-assembly, as the robot weld-able workpiece type is pre-defined by operator and designer, only the workpiece size is required to make job-programs. However, for the sub-assembly, robot weld-able workpiece is not pre-defined, the operator have to select robot's workpiece and input not only size but also shape of the workpiece. So, CAD interface works more important role for the sub-assembly line. For example, to plan a robot motion trajectory of a line-type seam in a sub-assembly, the start point and destination point are required. In the case of an arc-shape seam, the start point, mid point, and destination point are required. Accordingly, the required points are extracted from the vertex datum of the TRIBON CAD in according to the {O} coordinate frame in Fig. 5. Because a seam is defined as an adjacent line between two elements, the designated vertexes can be separated from the rest of the vertexes.

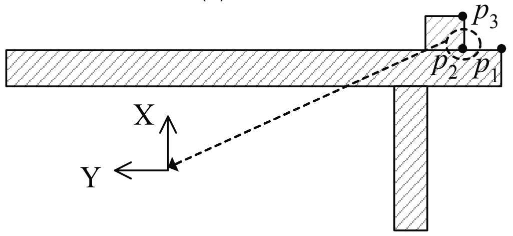
The orientation of the torch defined by roll-pitch-yaw values is extracted as shown in Fig. 11, which depicts an example of a specific shape of a workpiece and of the tool orientation values for the welding start and end points. For other shapes of welding seam such as straight line, curved line, and vertical line, normal robot job is accomplished. For the critical examples in Fig. 11, the geometric constraints of a workpiece are: (1) all the elements are made of plate of a thickness of less than 20 mm; (2) the possible welding length of a robot is longer than 200 mm. After acquiring the welding start and end points of workpieces from the output of CAD interface where the points are listed line by line in accordance to the sequence of composing complete object, 3 adjacent points of all the edge in a workpiece are gathered as $p_1(x_1, y_1, z_1)$, $p_2(x_2, y_2, z_2)$, and $p_3(x_3, y_3, z_3)$. And then, the center positions p_c and p_o are obtained as (1) and (2).



(a) Case 1



(b) Case 2



(c) Case 3

Fig. 11. An example of torch pose calculation (cut view of the workpiece). (a) Case 1: convex and available part; (b) Case 2: concave and available part; (c) Case 3: concave and unavailable part.

$$p_c = (p_2 + p_3)/2 \tag{1}$$

$$p_o = (p_1 + p_3)/2 \tag{2}$$

The values p_c and p_o are depicted in Fig. 11.

Let the distance between two points $p_A(x_A, y_A, z_A)$ and $p_B(x_B, y_B, z_B)$ be

$$l(p_A, p_B) = \sqrt{(x_A - x_B)^2 + (y_A - y_B)^2 + (z_A - z_B)^2} \tag{3}$$

The union vectors are obtained as

$$\hat{u}_o = \frac{\overrightarrow{p_o p_c}}{l(p_o, p_c)} \tag{4}$$

$$\hat{u}_1 = \frac{\overrightarrow{p_1 p_c}}{l(p_2, p_c)} \tag{5}$$

$$\hat{u}_2 = \frac{\overrightarrow{p_2 p_c}}{l(p_3, p_c)} \tag{6}$$

The two vectors are obtained as

$$\begin{cases} u_{T1B} = \overrightarrow{u_1} + \overrightarrow{u_o}, \\ u_{T2B} = \overrightarrow{u_2} + \overrightarrow{u_o}, \end{cases} \tag{7}$$

where u_{T1B} and u_{T2B} are the vectors of the tool's direction projected on a plate.

Considering the shape of the seam line and the constraints, u_{T1B} and u_{T2B} are translated into real tool direction vectors as in the following 4 cases: Case 1 is the convex part of a workpiece; Case 2 is the concave part of a workpiece; Case 3 is the impossible shape of the robot's welding; and Case 4 is considered as a normal welding seam line.

Case 1: $\{l(p_1, p_2) \le 20$ and $l(p_2, p_3) \ge 200\}$ or $\{l(p_1, p_2) \ge 200$ and $l(p_2, p_3) \le 20\}$

$$\begin{cases} u_{T1} = R_{XYZ}(\frac{\pi}{4}, 0, \pi)\, u_{T1B}, \\ u_{T2} = R_{XYZ}(\frac{\pi}{4}, 0, \pi)\, u_{T2B}. \end{cases} \tag{8}$$

Case 2: $l(p_1, p_2) \ge 200$ and $l(p_2, p_3) \ge 200$

$$\begin{cases} u_{T1} = R_{XYZ}\{\frac{\pi}{4}, 0, sign(\overrightarrow{u_1} \times \overrightarrow{u_o})\frac{\pi}{2}\}\, u_{T1B}, \\ u_{T2} = R_{XYZ}\{\frac{\pi}{4}, 0, sign(\overrightarrow{u_2} \times \overrightarrow{u_o})\frac{\pi}{2}\}\, u_{T2B}. \end{cases} \tag{9}$$

Case 3: $l(p_1, p_2) < 20$ and $l(p_2, p_3) < 20$

Indeterminate seam.

Case 4: $20 < l(p_1, p_2) < 200$ and $20 < l(p_2, p_3) < 200$

Normal seam.

where $R_{XYZ}(\gamma, \beta, \alpha) = \begin{bmatrix} c\alpha c\beta & c\alpha s\beta s\gamma - s\alpha c\gamma & c\alpha s\beta c\gamma + s\alpha s\gamma \\ s\alpha c\beta & s\alpha s\beta s\gamma + c\alpha c\gamma & s\alpha s\beta c\gamma - c\alpha s\gamma \\ -s\beta & c\beta s\gamma & c\beta c\gamma \end{bmatrix}$

Finally, the tool orientation value of each seam $R_{T1}(\gamma, \beta, \alpha)$ according to world frame {W} is defined as

$$\begin{cases} R_{T1}(\gamma,\beta,\alpha)={}^{W}T_{O}\,u_{T1}, \\ R_{T2}(\gamma,\beta,\alpha)={}^{W}T_{O}\,u_{T2}\,. \end{cases} \tag{10}$$

The value of ${}^{W}T_{O}$ is shown in Fig. 5.

In this way, the teaching points of newly introduced seams can be calculated using the information of CAD interface. Also, because the once defined rules can be adapted to other similar cases, newly rule-adding works are decreasing.

4.2 Automatic generation of robot program

In operating welding robots in a shipyard, the largest of time consumption is in robot programming. Particularly, for the workpieces of different shapes and sizes, more time is required to render the robot programs operable in real-time. To minimize time consumption, robot programs are often generated automatically from the welding information gathered from the CAD interface. First, by the analysis of the shape of the workpiece, the shape can be represented as a simple geometry such as a scallop, a hole, a horizontal line, a horizontal curve and a vertical line, and others. The programs for these simple geometries are already pre-created and saved in the robot program D/B. When the workpiece is allocated to the robot system, the robot program generation algorithm divides the workpiece into simple geometries already defined. Next, the robot programs required for the respective simple geometries are selected from the D/B and combined together to complete the entire program for the given workpiece. Fig. 12 shows an example of workpiece composition using simple geometry method. The size of the workpiece from the CAD data is reflected in the teaching points in the program.

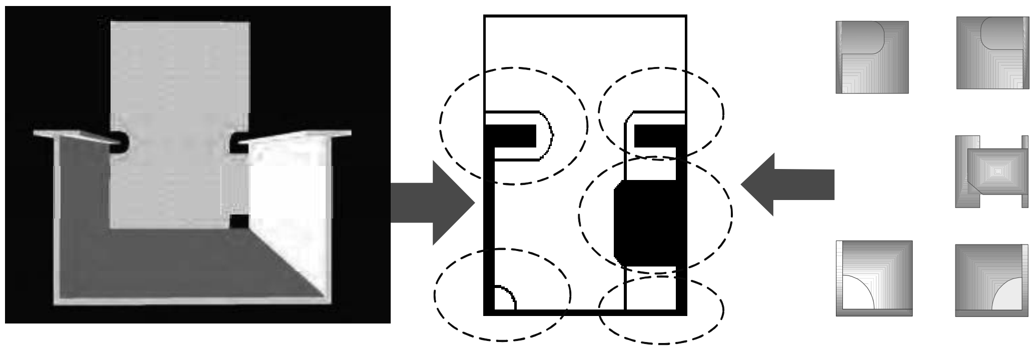


Fig. 12. An example of workpiece composition for a grand-assembly.

Fig. 13 is a flow chart representing the automatic robot program generation. The robot program generation algorithm is composed of 5 sub-routines: input data conversion, via-point creation, robot program selection using simple geometries, compilation of the combined robot programs of simple geometry, and robot program writing. The input data conversion routine creates teaching points for each seam of the work place by the methodology explained above. In the via-point creation routine, all of the via-points are calculated in such a way that no collision and no singular occur in moving from one teaching point to another teaching point. The nonsingular and collision-free path is obtained by pre-simulation results that are obtained for all of the shapes of the workpieces. In the robot program selection routine, the simple geometries of a workpiece are matched to respective robot programs. In the compilation routine, the matched robot programs are combined into a standard program. The writing program routine rewrites the robot program to fit it to the format of the robot language. Moreover, it sorts the teaching points according to the teaching point's number and matches each teaching point in the robot program to the respective values in the rule- and rpy- files.

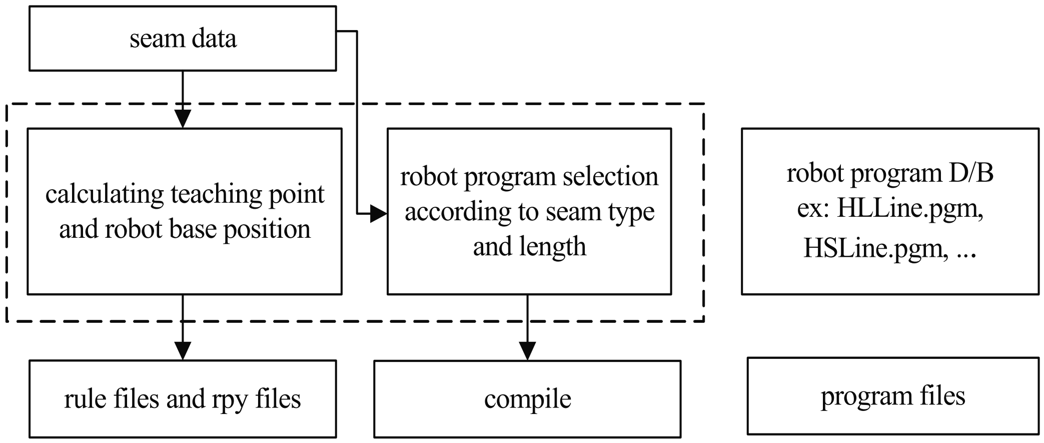

seam data
calculating teaching point and robot base position
robot program selection according to seam type and length
robot program D/B ex: HLLine.pgm, HSLine.pgm, ...
rule files and rpy files
compile
program files

Fig. 13. A flow chart for automatic program generation.

5. Block Arrangement and Path Generation

For the grand- and middle-assembly, the assembly block position is fixed on a workplace, the block arrangement and path generation is not required. But, for the sub-assembly, as the block can be placed on a workplace by operator using over-head crane, we can plan the block arrangement by the simulation to predict the base position of each workpiece. Also, for the fully automated robotic system of a sub-assembly line, robot's path planning to minimize travel length is very important to reduce working hours.

5.1 Block arrangement simulation

In order to increase the efficiency of a block arrangement and reduce the positioning errors between a real arrangement and its simulation, block arrangement simulations are performed on the off-line programming. Fig. 13 illustrates a block arrangement simulation view for the sub-assembly line.

The block arrangement simulation helps not only to improve the efficiency of the workplace but also to correctly position the blocks. In this manner we can arrange as many workpieces as possible in a bounded area. Thereby, the block arrangement simulation with 3D block models is very beneficial to estimate the amount of work per a workplace and job scheduling. Also, the arranged block position and drawing can be printed out to a document that helps the workers to place the blocks on a workplace. The reference coordinates of the robot program generation are based upon the local coordinate frame {O} of each workpiece depicted in Fig. 5. Therefore when the robot program is applied to a workplace where many workpieces are scattered such as Fig. 14, the reference coordinates of each robot program are translated into a world coordinate frame {W}. The translation matrix from {O} to {W} is provided by the results of the block simulation and a vision sensor.

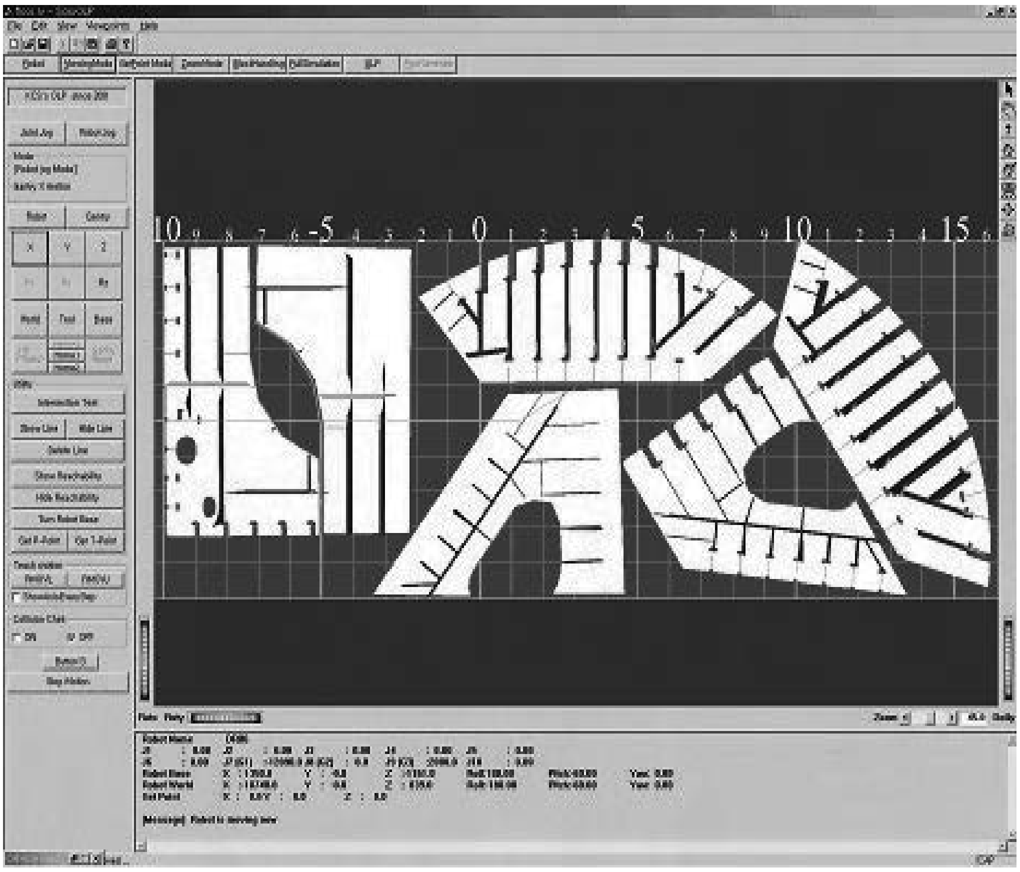


Fig. 14. Block arrangement simulation view.

The vision sensor is used to calibrate the unloading position of workpieces difference between the simulation and the real operation. The operation of the vision sensor is searching marker on a workpiece and marker's position is pre-defined. To reduce the vision sensor's marker-searching time, the initial searching position is obtained by the result of the block arrangement simulation. Once the vision sensor captures the positions of the 3 markers placed on the workpiece, the block's real position is calculated. Because the vision sensor is placed near the robot base frame {B}, translation only on the x-y plane is possible. The values of translation from the vision sensor to the base frame {B} can be obtained as

$$\begin{cases} x_{\{B\}} = x_{vision} + 300, \\ y_{\{B\}} = y_{vision} \,, \end{cases} \tag{11}$$

where x_{vision} and y_{vision} are the captured values of each mark according to the vision sensor base and 300 is the offset of the vision sensor from the base frame {B}. Next, the marker position in reference to the world frame {W} is obtained as

$$M_{\{W\}} = \begin{bmatrix} d_x \\ d_y \end{bmatrix} = {}^{W}_{B}T \begin{bmatrix} x_{\{B\}} \\ y_{\{B\}} \end{bmatrix} \tag{12}$$

where M_W is the marker position. The rotation values relative to the {W} frame are calculated using 3 different M_Ws on the plate. The robot's job program generated for the {O} frame can be transformed into the {W} frame by using the transformation ${}^{W}T_O$ as follows:

$${}^{W}T_O = \begin{bmatrix} \cos\theta & -\sin\theta & 0 & d_x \\ \sin\theta & \cos\theta & 0 & d_y \\ 0 & 0 & 1 & d_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{13}$$

where d_x, d_y, and d_z are the position components of the origin of the {O} frame in reference to the {W} frame. In most cases d_z is zero which means that the workpiece is always placed on the workplace. If d_z is not zero, the operator have to measure the height of the workpiece's base plate and inputs it to the off-line programs. The origin of the {O} frame and the positions of the markers are predefined by the designer.
In the case of a multi-robot system, a calibrated job program is allocated to each robot by a job program distribution algorithm, in which the determining fact is the sum of the lengths of all of the seams in each robot's working area.

5.2 Optimal path generation using genetic algorithm

Given that approximately 100~150 seams are randomly distributed on the workplace, the robot's traveling path should be minimal, resulting in the minimal working time. Among many optimization tools, the genetic algorithm (GA) is used for our robot system (Davidor, 1991; Fonseca & Flemming, 1995; Munasinghe et al, 2003). Travel length minimization is more efficient than working time optimization. There are two reasons for this. First, welding equipment malfunction, a robot's unpredictable motion, and interruptions by operators occur frequently while a robot is working; therefore, the time consumed by these problems cannot be considered in the GA calculation and simulation. Second, as the acceleration and deceleration times for trajectory planning are different according to the length of a seam, so the welding time is not exactly known. Alternatively, by assuming that the velocity of the end-effecter is constant, the welding time can be calculated. But this approach is the same as the travel length minimization by the inverse proportional relationship of time and length. Here, the optimization problem takes the form of a traveling length minimization.
In optimizing a small amount of genetic information or in solving a simple optimization problem, the genetic algorithm that uses binary bit string encoding is prevalent. But, for welding robots in a shipyard, multi-robot cooperative work with regard to several seam positions, teaching points, variant robots and welding information has to be considered. Accordingly, the use of character-type encoding is suggested. Fig. 15 shows the structure of a chromosome. The number of genotypes is the same as the number of seams. The assembly id, path id, and robot id are encoded in a genotype.

< robot ID >	< robot ID >		< robot ID >
(ass_*1* ID, path_*1* ID)	(ass_*2* ID, path_*2* ID)	● ● ●	(ass_*n* ID, path_*n* ID)

Fig. 15. Structure of the chromosome used for robot's path planning.

Frequently, a general crossover method may result in a collision path in the multi-robot cooperative work. The general crossover exchanges a gene of parent 1 with a gene of parent 2. In this case, the exchanged gene of the assigned robot can be different, and so the child generated by the parents has to take the collision path. Therefore, we use the PMX (Partially Matched Crossover) suggested by Fonseca (Fonseca, 1995). The PMX guarantees the validity of the solution of the traveling salesman problem (TSP) presented as a path generation. The children of the PMX inherit some pieces of the genetic information from the first parent by 2-points crossover and the other pieces are inherited according to the gene sequence of the second parent. The random number of each gene determines the possibility of crossover occurrence. For example, assume that the parents are defined as P_1 and P_2, the children are defined as C_1 and C_2, and the chromosomes, P_1 and P_2, are initialized as $P_1 = (1\,2\,3 \mid 4\,5\,6\,7 \mid 8\,9)$ and $P_2 = (2\,4\,5 \mid 3\,7\,6\,9 \mid 1\,8)$, where '$|$' is a cut point, physically, a district separation, for each robot. The number between cut points is an allocated path's number. For the case of P_1, the 1st, 2nd, and 3rd paths are allocated to the first robot, and the 4th, 5th, 6th, and 7th paths are assigned to the second robot, and the 8th and 9th paths are assigned to the third robot. The '$\times$ ' is an undefined gene. If the child inherits the second robot's genes, $C_1 = (\times\times\times \mid 3\,7\,6\,9 \mid \times\times)$ and $C_2 = (\times\times\times \mid 4\,5\,6\,7 \mid \times\times)$, then the related mapping is $3 \leftrightarrow 4,\ 7 \leftrightarrow 5,\ 6 \leftrightarrow 6,\ 9 \leftrightarrow 7$. By the transitivity rule, the mapping is changed into $3 \leftrightarrow 4,\ 9 \leftrightarrow 5,\ 6 \leftrightarrow 6$. If genes independent of the mapping are inherited, the children are $C_1 = (1\,2\,\times \mid 3\,7\,6\,9 \mid 8\,\times)$ and $C_2 = (2\,\times\times \mid 4\,5\,6\,7 \mid 1\,8)$. The final children are obtained as $C_1 = (1\,2\,4 \mid 3\,7\,6\,9 \mid 8\,5)$ and $C_2 = (2\,3\,9 \mid 4\,5\,6\,7 \mid 1\,8)$ by the mapping. Concurrently, the assigned genes for each robot are changed with respect to the cut points.

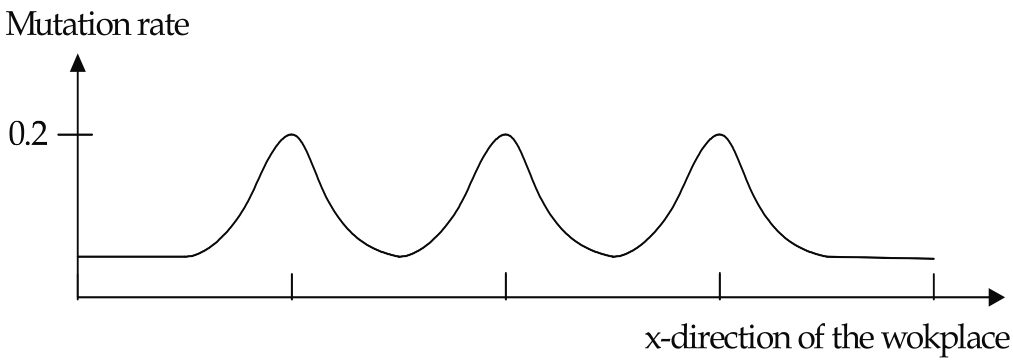


Fig. 16. The mutation rate of the spatially adaptive mutation algorithm.

Although the PMX is suitable to prevent a colliding path, the crossover method is restricted in the sense that a crossover occurs over a limited range. Hence the range of solutions is also restricted, and a set of solutions does not differ from an initial population but is constant. To search variable solutions, an adaptive mutation method that enables the exchange of the allocated work by PMX is used. The adaptive mutation method works better than the constant mutation rate method and also prevents collisions among multi-robots. The mutation rate is an important factor determining the convergence and the rate of the convergence of the solution. For a real robot system, because the robot's access to the neighboring seams near the separation layer is easier than its access to distant seams, the mutation rate of seams near the separation layer is high and the mutation rate of seams distant from the separation layer is low. The separation layer is predefined by the block arrangement. Here, we accommodate a Gauss function in (14) as an adaptive mutation rate. Fig. 16 shows the Gauss function.

$$y_i = k \exp\{-\frac{(x-x_i)^2}{2\sigma^2}\} \tag{14}$$

where $i = 0,\dots,n$, n is the number of robots, $k=1$, $\sigma = 0.25$, x_i is the position of the gantry, and y_i is the mutation rate. The dominant mutation rate in (14) is determined by k, which is the maximum value, and x, which is obtained when y is more than 68% of the whole area.

We use the rank selection method as a selection routine. First, the chromosomes are sorted by the fitness of each chromosome. Second, the value is calculated as follows:

$$\text{value} = \frac{\text{bias - }\sqrt{\text{bias}\times\text{bias - 4.0 (bias - 1.0)}\times\text{drandom()}}}{2.0\times(\text{bias - 1.0})} \tag{15}$$

where the range is the size of a population, and drandom() is a function that returns a random value. The position of the selected chromosome is defined as

$$\text{range - (base + value)} \tag{16}$$

where the base is the position of the last chromosome in a population. The selection constraints are assumed as follows: first, no robot can move beyond its working area restricted by the gantry crane's span; second, no collision is permitted; third, there is no limit of seam length for welding equipment. Examples of GA computing for a simple sub-assembly are shown in Fig. 17.

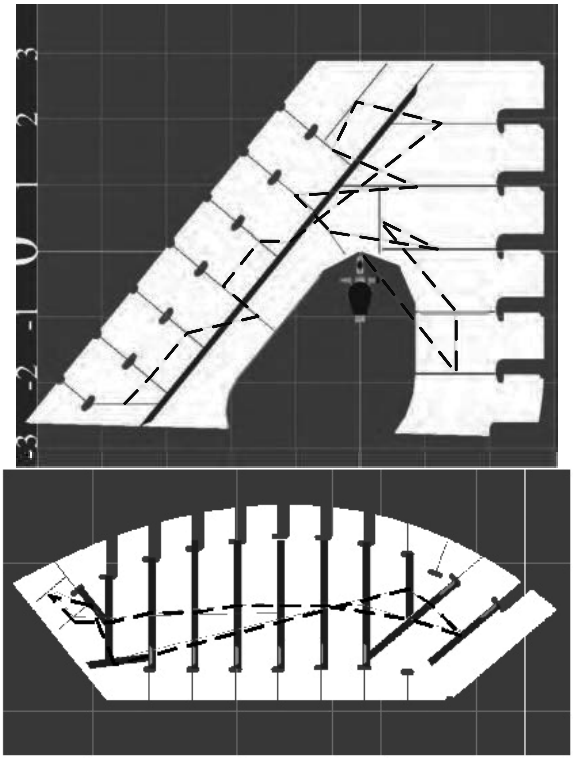
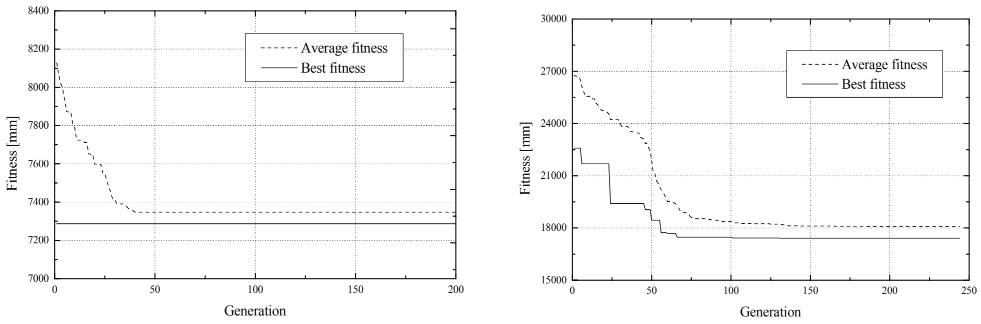


Fig. 17. Two examples of an optimal path generated for a single robot system: The dashed line indicates the optimal path represented by the robot base position.



(a) For the case of 3 robots. (b) For the case of a single robot.

Fig. 18. Fitness of the GA algorithm for an example of Fig. 17. The best fitness value of (a) is 7,250 mm and the best fitness value of (b) is 17,000 mm. As we expected, the 3 robot system is more efficient than the single-robot system. In (a), we can see that due to the best chromosome generated during initial population computation, the best fitness value is constant over the entire generation.

Fig. 18 depicts the best fitness value and the average fitness value of each generation for 3 robots and for a single robot. For the multi robot system, to allocate workpiece to each robot, we engaged simple allotment method handled by operators. 2 main determinant factors of allotments are welding length and collision avoidance. To prevent collision, we restricted robot's working volume by dividing workplace into the same numbered zone of robot. So the workpieces lying on each zone is allotted to each robot system initially. In the GA, initially allotted workpieces are modified partly to justify the welding length of each robot system. As we expected, we can see that 3 robots are more efficient than single robot according to the computation time and the best fitness value.

6. Implementation

The tools utilized in this work are as follows. Welding robots with the semi-automatic transfer system is applied to the grand-assembly and fully automatic robot system is applied to the sub-assembly. To implement the off-line programming both of grand- and sub-assembly: the PC of a Pentium IV 2.4 GHz processor with Windows 2000; Visual C++ for main programming and other functions; Open Inventor for graphic environment; TCP/IP for communication among the off-line programming PC, on-site monitoring PC and controller. Here, because the Open Inventor is a graphic library that provides collision detection algorithms, it is useful in constructing graphic environments. To increase the efficiency of the graphic processing ability, the selection of a video card is very important. The implemented video card has a 64 MB frame buffer memory and a 128 MB texture memory for our system. Also it is optimized to process Open GL. Also, vision system and mark-recognition algorithm is also developed to help welding robot system in shipbuilding.
On account of OOP(Object Oriented Programming), the user can selectively use each function of OLP, which automatically performs all functions in real time. Considering users' convenience, for the grand-, mid- and sub-assembly welding robot systems, robot simulation and robot program automatic generation functions were mainly used. And for the sub-assembly welding robot system, the whole operation of PC-based off-line programming is used. Table 1 shows the effectiveness of welding robot system with PC-based off-line programming.

Approximate time to	With developed OLP	On-line teaching
Generate a standard program	under 2 sec	About 20 min
Generate all standard programs	under 5 min	1 week
Generate job programs	under 5 min	1 hour
Total welding time for a general type workpiece (about 3m length, 3pass)	Under 5 min	About 30 min

Table 1. The efficiency of PC-based OLP compared with on-line teaching.

7. Conclusion

This chapter described the welding robot system in a shipbuilding industry and the PC-based off-line programming system. The welding robot system is very helpful not only to increase productivity but also to solve worker's health problem. The developed off-line programming system provides a robot simulation, a block arrangement simulation, optimal robot traveling path generation, and automatic robot program generation. Because graphic environments are made in the VRML, developed off-line programming is highly compatible with other software, which fact allows the use of off-line programming on the Internet.

Thereby, adjustments to various robot systems are easy. Developed off-line programming is very easy for operators to use and maximizes the operating efficiency of welding robot systems of the shipbuilding industry. In the future, due to intelligent robotic techniques such as PC-based OLP, painstaking human labor will be reduced and manufacturing productivity in shipyards will be increased.
The contributions of this chapter are as follows: (1) a methodology for applying a welding robot system that works for variously shaped objects, especially for assembly lines of shipbuilding, is suggested; (2) the functions required to implement an OLP successfully and the development of a PC-based OLP that helps on-site operators handle a robot system easily are explained; and (3) the practical implementation of recently issued algorithms such as the VRML of the simulation environment, the geometrical computation of the CAD interface, the computing techniques of the automatic generation of robot programs, and the GA of robot path planning, are shown.
Using the techniques of welding robot system, utilizations for other hard process of shipbuilding such as painting and grinding are future works.

8. Acknowledgements

The research described in this chapter has been supported by the Research Center for Logistics Information Technology (LIT) designated by the Ministry of Education and Human Resources Development, Korea.

9. References

Aiyama, Y. & Tanzawa, H. (2003). Management system for distributed and hierarchical robot groups, *Advanced Robotics*, Vol. 17, No. 1, pp. 3-20, ISSN: 1568-5535

Bae, K. Y.; Lee, J. H. & Jung, C. W. (1998). A study on the development of an arc sensor and its interface system for a welding robot, *Journal of the Korean Welding Society*, Vol. 16, No. 3, pp. 129-140, ISSN: 1225-6153

Borm, J. H. & Choi, J. C. (1992). Improvement of local position accuracy of robots for off-line programming, *KSME International Journal*, Vol. 6, No. 1, pp. 31-38, ISSN: 1011-8861

Buchal, R. O.; Cherchas, D. B.; Sasami, F. & Duncan, J. P. (1989). Simulated off-line programming of welding robots, *International Journal of Robotics Research*, Vol. 9, No. 3, pp. 31-43, ISSN: 0278-3649

Carvalho, G. C.; Siqueira, M. L. & Absi-Alfaro, S. C. (1998). Off-line programming of flexible welding manufacturing cells, *Journal of Materials Processing Technology*, Vol. 78, No. 1-3, pp. 24-28, ISSN: 0924-0136

Choi, M. H. & Lee, W. W. (2003). Quantitative Evaluation of an Intuitive Teaching Method for Industrial Robot Using a Force / Moment Direction Sensor, *International Journal of Control, Automation, and Systems*, Vol. 1, No. 3 pp.395-400, ISSN: 1598-6446

Craig, J. J. (1986). *Introduction to Robotics: Mechanics and Control*, Addison-Wesley, ISBN: 0-201-09528-9, Reading, Massachusetts

Davidor, Y. (1991). *Genetic Algorithms and Robotics: A Heuristics for Optimization*, World Scientific, ISBN: 9-810-20217-2, Singapore

Ferretti, G.; Filippi, S.; Maffezzoni, C.; Magnani, G. & Rocco, P. (1999). Modular dynamic virtual-reality modeling of robotic systems, *IEEE Robotics and Automation Magazine*, Vol. 6, No. 4, pp. 13-23, ISSN: 1070-9932

Fonseca, C. M. & Flemming, P. J. (1995). Genetic algorithms for multi objective optimization, *Evolutionary Computation*, Vol. 3, No. 1, pp. 1-16, ISSN: 1063-6560

Goldberg, D. & Lingle, R. (1985). Alleles, Loci, and the TSP, *Proceedings of the First International Conference on Genetic Algorithms*, pp.154-159, ISBN: 0-805-80426-9, London, Oct., 1985, Lawrence Erlbaum Associates, Hillsdale, NJ

Hoffmann, C. M. (1989). *Geometric and Solid Modeling: An Introduction (The Morgan Kaufmann Series in Computer Graphics and Geometric Modeling)*, Morgan Kaufmann, ISBN: 1-558-60067-1

Kobayashi, N.; Ogawa, S. & Koibe, N. (2001). Off-line teaching system of a robot cell for steel pipe processing, *Advanced Robotics*, Vol. 12, No. 3, pp. 327-332, ISSN: 1568-5535

Kreuzer, B. & Milojevic, D. (1998). Simulation tools improve planning and reliability of paint finishing lines, *Industrial Robot*, Vol. 25, No. 2, pp. 117-123, ISSN: 0143-991X

Kunii, T. L. & Shinagawa, Y. (1992). *Modern Geometric Computing for Visualization*, Springer-Verlag, ISBN: 0-387-70105-2, Tokyo

Munasinghe, S. R.; Nakamura, M.; Goto, S. & Kyura, N. (2003). Trajectory Planning for Industrial Robot Manipulators Considering Assigned Velocity and Allowance Under Joint Acceleration Limit, *International Journal of Control, Automation, and Systems*, Vol. 1, No. 1, pp.68-45, ISSN: 1598-6446

Nagao, Y.; Urabe, H.; Honda, F. & Kawabata, J. (2000). Development of a panel welding robot system for subassembly in shipbuilding utilizing a two-dimensional CAD system, *Advanced Robotics*, Vol. 14, No. 5, pp. 333-336, ISSN: 1568-5535

Nielsen, L. F.; Trostmann, S.; Trostmann, E. & Conrad, F. (1992). Robot off-line programming and simulation as a true CIME-subsystem, *Proceedings of 1992 IEEE International Conference on Robotics and Automation*, pp. 1089-1094, ISBN: 0-818-62720-4, Nice, France, May, 1992, IEEE Computer Society Press

Okumoto, Y. (1997). Advanced welding robot system to ship hul assembly, *Journal of Ship Production*, Vol. 13, No. 2, pp. 101-110, ISSN: 1542-0469

Richard, G. (1997). Object-oriented programming for robotic manipulator simulation, *IEEE Robotics and Automation Magazine*, Vol. 4, No. 3, pp. 21-29, ISSN: 1070-9932

Sheu, P. C-Y. & Xue, Q. (1993). *Intelligent Robotic Planning Systems*, World Scientific, ISBN: 9-810-207558-1, Singapore

Wernecke, J. (1994). *The Inventor Mentor: Programming Object-Oriented 3D Graphics with Open Inventor*, Addison-Wesley, ISBN: 0-201-62495-8, Canada

Intelligent Robotic Handling of Fabrics Towards Sewing

Panagiotis Koustoumpardis, Paraskevi Zacharia, Nikos Aspragathos
University of Patras, Dept. of Mechanical Engineering & Aeronautics, Robotics Group
Greece

1. Introduction

Handling of flexible materials is one of the most challenging problems that have arisen in the field of robot manipulators. Besides the difficulties (e.g. geometrical uncertainty, obstacle avoidance, etc.) that emerge when handling rigid materials using robots, flexible materials pose additional problems due to their unpredictable, non-linear and complex mechanical behaviour in conjunction with their high flexibility and high bending deformations. The fact that sewing fabrics is a "sensitive" operation, since fabrics distort and change their shape even under small-imposed forces, poses barriers in the development of automated sewing systems. On the other hand, the need for great flexibility in garment assembly system is really imperative, since cloth manufacturing should cope with the fast fashion changes and new materials for fabrics and responding to the consumer demands.
Our research efforts are focused on the development of intelligent control systems with multi-sensor feedback, enabling robots to perform skillful tasks in realistic environments towards higher flexibility and automation. In this work, the robot control approaches based on artificial intelligence for handling fabrics towards feeding in the sewing machine are described in detail.

2. State-of-the art and Related work

The cloth manufacturing is one of the less automated industries, compared with other industries such as automotive, computer etc. The main reason for this delay is the high bending flexibility of the fabric in changing its shape when it is handled. The high versatility of the fabric size, type, shape and their material characteristics increase the difficulties for the introduction of flexible automation in cloth making industry. The automatic systems that are up-to-date available, are characterized by high specialization, minimal programmability, limited flexibility to changes and require human intervention to accommodate different sizes and fabric types.
The fabric handling operations in the clothing industry can be divided in the following classes: separation, grasping, translation, placing, positioning, feeding and sewing. A lot of research has been done in the fabric acquisition using robotic grippers (Taylor, 1990; Monkman, 1996; Koustoumpardis & Aspragathos, 2004). While few researchers have worked on the automatic feeding of fabrics into the sewing machine.

In the sewing process, two robotic handling tasks (Gilbert et.al., 1995; Koustoumpardis & Aspragathos, 1999) require sophisticated control. The first one deals with the manipulation (translation and orientation) of the cloth and the second one with the control of the cloth tension.

A method for the manipulation of various fabric shapes was introduced (Torgerson & Paul, 1988) using visual information. The determination of robot motion paths is based on visual feedback defining the location of the fabric edges in world coordinates. The developed algorithm was used for handling both polygonal and non-polygonal shapes, whereas the accuracy of the approximation to the desired seam line depends on the camera resolution. However, this algorithm is based on geometrical computations, which has to be done manually for each fabric shape. In addition, the algorithm has been constructed ignoring the buckling or wrinkling that appears during fabric handling.

The first advanced robot sewing system reported by Gershon & Porat (1986; 1988) is the FIGARO system, where an integrated robotic system for sewing has been developed. In the FIGARO system, the following components are integrated: a robot manipulator endowed with a two-fingered end-effector, a sewing machine, two miniature CCD cameras mounted on the sewing machine and a force sensor mounted on one of the two fingers. For the fabric tension control an estimated cloth velocity was used based on the sewing machine shaft encoder and a correction to this estimation was computed by a proportional-integral controller in order to derive the robot velocity. The gains of the controller were adjusted by trial and error, which should be modified for a new type of fabric (Gershon, 1993). With the fabric tension control the buckling of the fabric was prevented and good seam quality was obtained.

A parallel process decomposition of the robot-sewing task was proposed (Gershon, 1990) to address the problem of robot sewing. A robot arm manipulates the fabric towards the desired orientation and control the fabric tension during the sewing process. The sewing task was decomposed into four concurrent processes within a superposition parallel architecture. The controlled system is modelled as: a mass-spring-damper model for representing the robot's equivalent dynamic behaviour, a nonlinear damped spring for the fabric, whereas the table friction acting on the fabric is included in the model. The performance of sewing is decomposed into straight line seams and seams that follow the edge contour. The complete system demonstrated robustness in the experiments.

After FIGARO system, an automated sewing system coordinating two robots handling the fabric on the table has been developed (Kudo et al., 2000). On top of robot hands coordination, fabric tension control and synchronization with the sewing machine speed were considered. Visual information was used to control seam path and its deviation from the desired trajectory through a CCD camera mounted on the sewing machine. Sewing experiments were carried out for a straight-line trajectory using pressing force control, fabric tension control and fabric position manipulation control. The experiments have been extended to seams that follow curved-line trajectory using coordinated position/force control. Extended experimentation under a wide variety of sewing speeds, panel contours, number of plies and fabric type proved the effectiveness and the robustness of the developed system.

As far as artificial vision for the control of the manipulator is concerned, the visual servoing systems are based on two main approaches: position-based and image-based visual servo control (Hutchinson et al., 1996). In the position-based control systems, the error is computed in the 3D Cartesian space. The position error is computed using or not the model of the target depending on the visual features available in the image. The main advantage of this approach is that the camera trajectory is directly controlled in Cartesian space. However, this approach has the limitation of being sensitive to calibration errors that arise either from a coarse calibration of the camera or from errors appeared in the 3D model of the target.

In the image-based control systems, the error is directly computed in the 2D image space. The main advantage of this approach is that there is no need for the 3D model of the target and it is quite robust not only with respect to camera but also to robot calibration errors. However, the Jacobian matrix is a function of the distance between the camera and the target, which is not easily calculated and this is a serious limitation of this model-free control approach.

A combination of the previous two approaches is used in the 2 1/2 D visual servo systems, where the error to be minimized is specified both in the image and in the pose space. This method avoids their respective disadvantages: contrarily to the position-based visual servoing, it does not need any geometric 3D model of the object. In comparison to the image-based visual servoing, it ensures the convergence of the control law in the whole task space.

For the most of the handling tasks and essentially in sewing, the cloth must be held taut and unwrinkled. The seam quality is extremely sensitive to cloth tension variations appeared in the sewing process. These undesirable tension variations affect the product quality. The difficulties are more evident when the seam is performed along the cloth bias, due to the increased fabric extensibility. Gershon (1990) justified the need of force feedback control in order to fulfil the above fabric's tension requirements. Furthermore, he underlined that the conventional control methods are inadequate to handle a fabric tension problem due to the fabric nonlinear behaviour, the noisy cloth tension signal, etc. Therefore, it is vital to develop more sophisticate and intelligent control methods.

Stylios (1996) reported that the intelligent behaviour can be expressed in terms of sensing, processing, actuating, learning and adapting without any previous knowledge about the properties of the object that the human is handling. To apply this approach in robot handling task, control systems based on Neural Networks should be designed. The ability of Neural Networks to work without having any previous knowledge of the controlled system behaviour and their ability to learn from examples and to adapt as they modify themselves during the training phase, incorporate the human like behaviour in handling of non-rigid materials. The Neural Networks and Fuzzy Logic benefits have been used in apparel industry (Stylios & Sotomi, 1996) and especially in gripper's design for fabric handling (Tsourveloudis et.al., 2000).

The objective of the present work is the development of a robotic system for handling of non-rigid materials lying on a working table. The work is focused on the handling of flexible materials lying at a random location and orientation on a table and guide them towards the sewing needle and along the "seam line", as well as on the control of the appropriate tensional force in order to maintain constant high seam quality. The handling strategies are developed for fabrics of various shapes (convex, non-convex, with straight and curved edges). A robot fuzzy controller is presented, for guiding the fabric towards the sewing needle, where the fuzzy rules are derived by studying human sewing. Our goal is to design a robust controller, which autonomously determines the motion of the robot avoiding special geometrical computations and independent of the fabric's shape. The appropriate velocity of the robot end effector in the sewing process is regulated by a Neuro-Controller. Likewise, the appropriate tensional force applied to the fabric is determined by the developed Fuzzy Logic decision mechanism.

3. The robotized sewing problem

A scheme of the concept and the experimental layout of the sewing process are illustrated in Fig. 1. The sewing machine pulls the the fabric with the *machine velocity*, while the robot end-effector has to follow with the *robot velocity*, while manipulating and applying a recommended tensional force to the fabric.

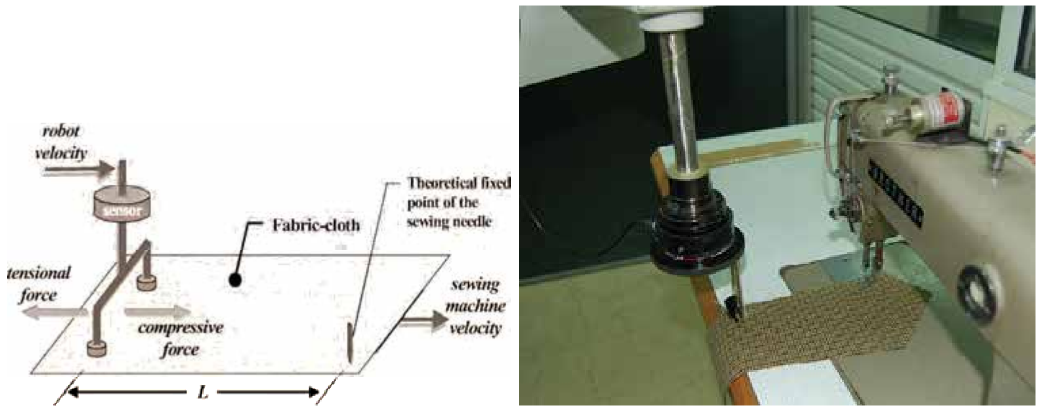


Fig. 1. The concept and the experimental layout of the robotized sewing.

The gripper contact with the piece of fabric is performed through rubberized ends that press slightly the fabric on the sewing table as it is shown in Fig. 1. The force sensor mounted on the robot end-effector measures a tensional force when the distance between the two acting points of the velocities is greater than the free length of the fabric, or measures a compressive force when this distance is less than the free length of the fabric. The first case occurs when the sewing machine velocity is greater than the robot end-effector velocity so the fabric is extended, while in the second case, the robot velocity is greater than the machine velocity and then the fabric is compressed. For obtaining pucker-free seams a constant tensional force should be assured.
The fabric characteristics and properties have to be taken into account in an automated robot sewing system. The appropriate tensional force depends on the fabric properties (Koustoumpardis & Aspragathos, 2003) while its variations during the sewing process affect the seam quality. Thereby, the fabrics have to be recognized into categories (knitted, woven etc.) depending on their physical properties. Another important factor that should be considered is the shape of the fabric, which can be convex or non-convex, with straight- and/or curved lines.
It can be concluded that an automated sewing system demands a classification of the fabrics into various categories as well as a preliminary scheme of the optimum path the robot should follow in order to produce the desired stitches. In the proposed sewing system, these demands have been taken into account for the design of a hierarchical control scheme. The target for this robot controller is the guidance of the robot to apply a

constant tensional force while the robot end-effector manipulates the fabric in the sewing process.

4. Pre-sewing tasks

The fabric handling tasks for cloth sewing are ply separation, placement on the working table, manipulation towards the sewing needle and fabric's tension control during the sewing process.
For the tasks "ply separation" and "placement on the working table" work has been done on the design of a robotic air-jet gripper for destacking fabrics (Zoumponos & Aspragathos, 2004) and the placement of fabrics on a working table (Zoumponos & Aspragathos, 2005) where a fuzzy motion planning was developed to control the robot. After the fabric has been placed at a random location on the working table, a number of sub-tasks should be performed before the sewing process starts. These preliminary sub-tasks are:

1. **The recognition of the fabric's shape.** The camera captures the image of the fabric piece lying on the working table free of wrinkles before the robot end-effector touches the fabric. The shape (convex or non-convex, with or without curvatures) and the location of the fabric are identified and is used as the reference shape, while the piece is handled by the robot.
2. **The edges that will be sewed.** There are two main kinds of stitches: those ones that are performed in order to join two parts of cloth together and others that are performed for decorative and aesthetical purposes (e.g. in the pockets of trousers and shirts). However, there are parts of cloths, where both types of stitches should be conducted. For example, if the fabric piece is a pocket, all edges, except for one, should be sewed and there are decorative stitches should be added. The information of the stitching lines on the fabric and the type of the stitches is taken from a CAD system where the cloth has been engineered.
3. **Planning of the sewing process.** The best sequence of the seam segments is determined before the sewing process in order to reduce the cycle time of the sewing. The optimum sequence can be found using Genetic Algorithms (Zacharia & Aspragathos, 2005) and this is the next step after the stitching lines have been determined. However, one should keep in mind that some stitches have antecedence in relation to others. In the previous example, the stitches that serve aesthetical purposes should be performed before the pocket is sewed onto the trouser part.
4. **The extraction of the "seam line".** The sewing process will be performed on a "seam line" situated inside the fabric edges. Since the fabric edges are extracted from the image taken from the camera, the "seam line" is situated some millimetres inside the fabric's outer line. For the straight lines, the "seam line" is found by transferring the outer lines inside the fabric and the intersection of these lines constitute the vertices of the "seam line". Consequently, the "seam line" is parallel to the outer edge and the distance between the outer edge and the "seam line" is determined, since it is different for different parts of the cloth. For the curved lines, the process of determining the "seam line" is similar to the aforementioned process, since the curve has been approximated by a polygon. The distance between the outer edge line and the "seam line" depends on the cloth part that is going to be sewed and is defined by the clothing industry manufacturer. The

system is capable of automatically extracting the "seam line" after the outer edges have been defined. It is clear that this distance is greater for a part from a trousers' leg than a part from a shit's sleeve.

5. **The initial position of the end-effector.** The fabric lies on the table at a random location with a random orientation and the end-effector has to move towards the fabric and touch it so that it can lead it towards the sewing machine.
   The problem of determining the critical buckling length L_c between the front edge and the end-effector location (see Fig. 2(a)) in order to avoid the buckling of the fabric has been investigated by Gershon & Grosberg (1992). In Fig. 2(b), the fabric buckling is illustrated when the initial length is larger than the critical one. Therefore, the robot gripper should touch the fabric at a distance from the front edge equal or shorter than the critical length in order to prevent the buckling.
   It has been investigated and concluded (Koustoumpardis & Aspragathos, 2003) that the appropriate position of the end-effector on the fabric depends mainly on the fabric type and its flexibility. A fuzzy decision mechanism is used as shown in Fig. 3.
   The term "flexibility" represents an index, which is estimated by an expert specifying a flexibility degree between 0–100%.

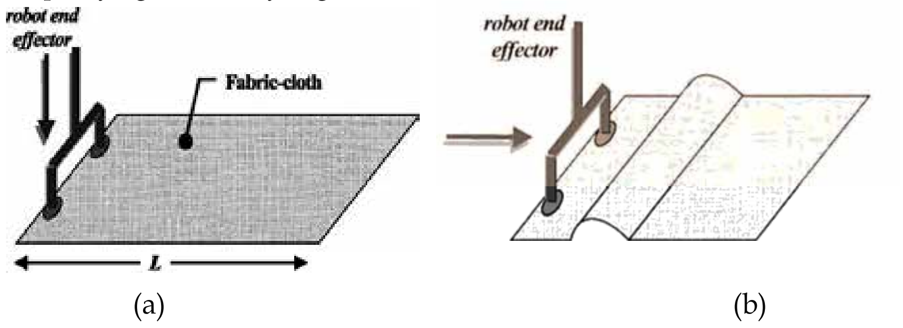


Fig. 2. The buckling of fabric when L is larger than the critical length.

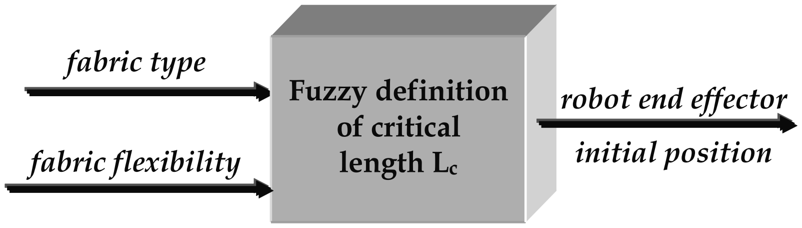


Fig. 3. Fuzzy definition of the initial end-effector position.

The determined values of the critical buckling length L_c are not strictly restrictive. This length can be equal or smaller than that determined using the Fuzzy decision mechanism, since the critical length L_c is the maximum permissible. The cases that smaller length must be used are determined by the geometric limits of the work cell components and their layout. For example, when the robot manipulator used in sewing has a maximum reach smaller than the derived by the Fuzzy mechanism, then the initial length L can be smaller than the critical one in order to meet the physical constrains of the robot. However, in the case of handling small pieces of

fabric, the position of the end-effector should be such that the covered area of the fabric be the smallest to facilitate the identification of the piece by the vision system.
After the execution of the above five preliminary sub-tasks the sewing process can be started. Throughout the sewing two main tasks are investigated and respective robot control algorithms are developed for the manipulation (translation and orientation) of the fabric and the regulation of the fabric's tensional force.

5. Manipulation of fabric

The proposed system is a combination of image-based and position-based control system. The image-based analysis is used for the identification of the fabric's shape. After the image acquisition of the fabric, the features (vertices for the straight edges and dominant points for the curved edges), the needle-point and the sewing line's orientation are derived from the image analysis. The position of the needle is known in the robot coordinate system too. The position of the end-effector on the fabric is random, but is subject to the constraints discussed in the previous section. This position is unknown in the image coordinate system; however, the robot system gives feedback of the current position of the robot in the robot coordinate system and the robot end-effector is now referred to the robot base system. Moreover, the relation between the robot- and the image- coordinate system is known from the calibration of the camera.
For the manipulation of the fabric towards the needle, the image based-approximation is used, since both the distance and the orientation of the movement are known in the image coordinate system. For the rotation of the fabric around the needle, the rotation angle is computed in the image coordinate system, but for the rotation of the robot end-effector around the needle, the needle-position relative to the robot base is used.

5.1 Sewing fabrics with straight edges

Fabrics are limp materials that present a rather unpredictable behaviour during handling. Since the fabric bending rigidity is very small, when the gravitational forces are applied to a piece of fabric its shape changes completely. Therefore, sewing the fabric using a robot is not easy due to the wrinkling, folding and buckling. Since fabric's shape is considerably changed, the handling of fabrics is performed onto a work table to ensure a kind of rigidization. However, it is possible that wrinkles will appear in robotic handling of fabrics lying on the table.
Since the fabric has been laid at a random location on the work table, the end-effector is placed at a proper position on the fabric, so that buckling problems are eliminated. At this point, the robot sewing process is ready to start. The sewing process consists of three sub-tasks: the manipulation of the fabric towards the needle, the sewing of the edge and the rotation around the needle which are described in the following:

1. ***The manipulation towards the sewing needle***. The manipulation of the fabric is performed by properly translating and orientating the end-effector guiding the fabric. The "seam line" are determined on the fabric shape identified by the vision system. In Fig. 4, the robot is going to guide a sleeve towards the sewing machine in order to sew all its edges. The distance (r) between the current position of the seam starting vertex and the needle and the angle (θ) between the sewing line direction and the current direction of the first edge of the "seam line" are computed (Fig. 4)

based on the image features identification. The linear and angular velocity of the fabric are derived from the position and orientation error through the designed fuzzy decision system described in Section 0. The position error (e_r) and the orientation error (e_θ) and their rate are the input data, whereas the linear and angular velocities (u), (ω) of the end-effector respectively are the output data.
The new position and orientation of the end-effector is computed through the linear and angular velocity, the time step dt and the angle φ, which is computed from the image. The fabric is transferred to a new position as a result of the movement of the end-effector, which is stuck on the fabric so the fabric does not slip relative to the gripper. However, the system can overcome the possible slipping under the cost of greater cycle time for the task. This manipulation stops when the edge of the fabric reaches the needle with the desired orientation within an acceptable tolerance.

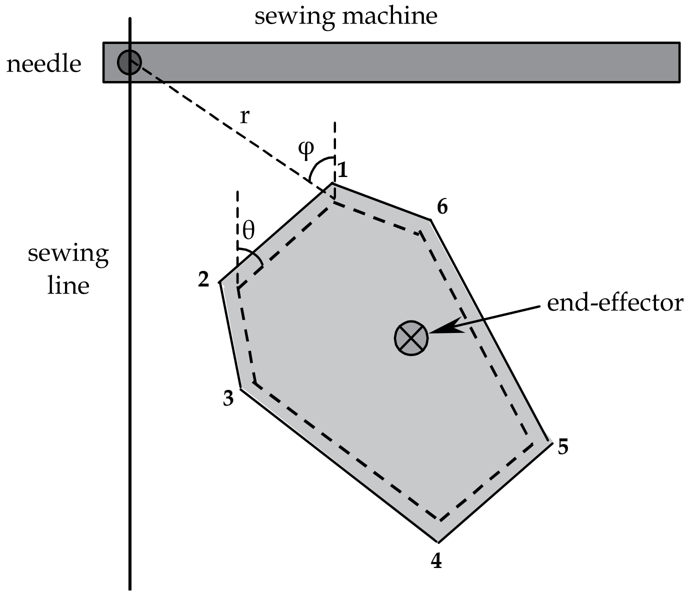


Fig. 4. Scene of the fabric lying on the table.

2. ***The stitching process***. The edge of the "seam line" of the fabric that was aligned with the sewing line is ready to be sewed. In sewing, the fabric is guided along the sewing line with a velocity, which should reconciles with the velocity of the sewing machine, so that good seam quality is ensured. The orientation error is monitored by the vision system and the error is fed to the robot controller in order to correct the orientation of the fabric.
3. ***The rotation around the sewing needle***. After one edge the "seam line" has been sewed, the fabric is rotated around the needle until the next edge of the "seam line" coincides with the sewing line. The orientation error (e_θ) of the next edge in relation to the sewing line and its time rate are computed. These are the inputs to the fuzzy system that controls the rotation of the fabric around the needle, whereas the output is the angular velocity of the end-effector. When this edge of the fabric coincides with

the sewing line under a certain tolerance, it is ready for sewing. The sewing process is iterated until all the edges of the "seam line" planned for sewing, are sewed.

5.2 Sewing fabrics with curved edges

Robot sewing is a complicated task that demands high accuracy, so that good seam quality is produced. The fact that current industrial robots can only be programmed to execute straight or circular motions leads to the need for developing an approach for sewing fabrics with arbitrary shape. To overcome this limitation and simultaneously exploiting the straight motion of the robot, the curved edges are approximated through straight lines.

However, the shapes of cloth parts have edges of arbitrary curvature and circular or free form curved seams are mainly used for aesthetical or decorative reasons. Therefore, more attention should be paid to the problem of sewing fabrics with arbitrary curved edges.

The problem of approximating the arbitrary curved edges of the fabrics with straight-line segment has been addressed using two different methods: the Teh-Chin Dominant Point Detection Algorithm implemented for fabrics (Zacharia[a] et al., 2006) and Genetic Algorithms with variable-length chromosomes (Zacharia[b] et al., 2006). In both approaches, our major goal was to ensure that the deviation from the desired path is lower than a predefined acceptable tolerance, so that the seam can be considered successful and the seam quality is satisfactory. In the approach based on Genetic Algorithms, an additional goal is the minimization of the number of straight line segment of the polygon that approximates the curve.

Initially, the camera captures the image of the fabric with the curved edge. The Teh-Chin Dominant Point Detection Algorithm (Teh & Chin, 1989) has been applied in order to extract the dominant points of the curve, using two criteria. The first deals with the length of the chord and the second with the ratio of the perpendicular distance from a point to the chord to the length of the chord. Next, the successive dominant points are joined with straight lines. However, using this method implies that the maximum deviation between the real curve and the straight lines is not defined through the algorithm, but it is found experimentally.

To overcome the drawback of defining the deviation through experimentation, Genetic Algorithms have been used as an alternative to the polygonal approximation of the curve edges of the fabric. In this work, a Genetic Algorithm version is introduced for determining the minimum total time needed for accomplishing the sewing process. This is achieved by approximating the curve section by a polygon with the minimum number of sides under the condition that the maximum deviation between the real and the desired "seam line" is less than an acceptable tolerance.

After the outer curved edge has been approximated by a polygon section, the problem of robot sewing fabrics with curved edges is reduced to the problem of sewing fabrics with straight edges. The procedure that is followed is similar to the procedure described in Section 4.1 for fabrics with straight edges.

5.3 The fuzzy control system for the manipulation of the fabric

From the standpoint of robot control design, the control systems are based on the plant models and make use of the relevant geometry for robot path analysis. However, in the developed robotic system for handling fabrics, the construction of a model running in real time is rather impossible. To alleviate this difficulty, and simultaneously make the system

more flexible, a fuzzy logic controller is designed for handling the fabric towards the sewing needle. A further advantage of the fuzzy controller is that it is robust and quite fast in deriving the control outputs.
The proposed controller (Zacharia et al., 2005) outputs the linear and angular velocity of the robot's end-effector. The block diagram for the control system is shown in Fig. 5, where the symbols in parenthesis stand for the fuzzy system that controls the orientation. After the camera has captured the image, the vertices of the fabric are extracted. Then, the current position r of the selected vertex and the orientation θ of the fabric edge are computed from the image, whereas the position of the needle r_d and the orientation θ_d of the sewing line in the image coordinate system are a priori known. The errors, defined as: $e_r=r_d-r$, $e_\theta=\theta_d-\theta$, where $r_d=0$ and $\theta_d=0$, and the error rates, defined as: $e_r'= e_r/dt$, $e_\theta'= e_\theta/dt$, are the inputs of the controller. The fuzzy controller outputs the linear and angular velocity and the robot end-effector moves to its new position.

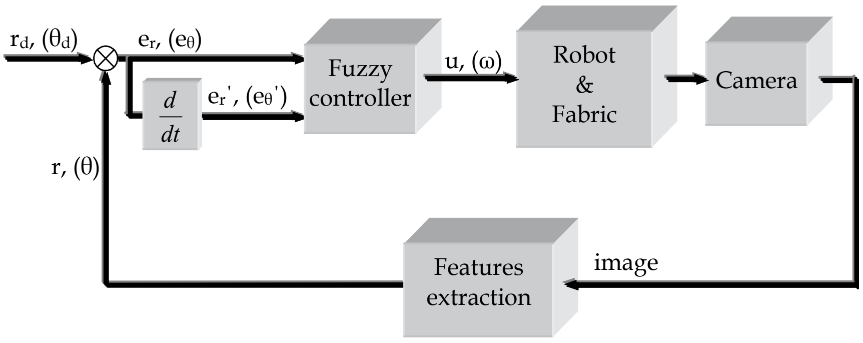
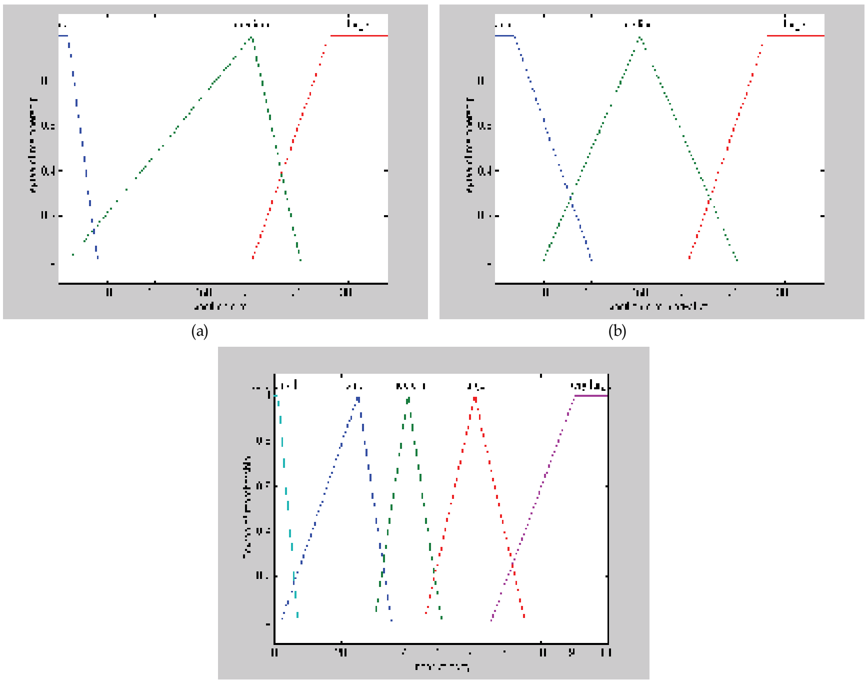


Fig. 5. The block diagram of the system.

The design of the proposed controller has the advantage that an analytical model of the robot and the fabric is not necessary, nor special mathematical computations for each fabric shape. In addition, the knowledge of the behaviour of the system is incorporated into the controller so the system responds to any position and orientation error regardless of the properties of the fabric. Lastly, the controller is able to manipulate a piece of fabric regardless of the shape and the type of the fabric and independent of the position and the orientation of the end-effector onto the fabric taking into account the constraints for avoiding buckling and highly covered fabric area. Therefore, the proposed fuzzy controller is flexible and robust, because it is capable of handling various types and shapes of fabrics.
Two fuzzy control systems are developed to extract the translational and angular velocity of the robot's end-effector. Each one input variable is expressed by three linguistic terms: *small*, *medium* and *large*, whereas each output variable is expressed by five linguistic terms: *very small*, *small*, *medium*, *large* and *very large*. The membership functions for the two inputs and the output of the system that controls the translation of the fabric are presented in Fig. 6(a), (b) and (c). For the fuzzification of the inputs, the linguistic terms *small* and *large* are defined by trapezoidal shapes and the term *medium* is defined by a triangular shape. For the fuzzification of the output, trapezoidal shapes define the terms *very small* and *very large*, whereas triangular shapes define the terms *small*, *medium* and *large*.
It should be mentioned that the universe of discourse for the position error is defined in pixels and not in centimetres, since the distances are computed on the image. For the system

that controls the rotation of the fabric, similar membership functions are provided through extended experimentation.

The rule base of each system is composed of 3×3=9 linguistic rules. The rule base includes the knowledge acquired by the systematic investigation of the fabric handling by human workers in feeding the sewing machine. The acquired knowledge is formulated in rules of the following form:


(a) (b)

(c)

Fig. 6. Definition of the membership functions.

"The larger the distance from the needle and the smaller its error rate is, the faster the fabric should move towards the needle"

"The larger the angle between the fabric's edge and the "seam line" and the smaller its error rate is, the faster the fabric should rotate towards the "seam line""

Table 1 shows the Fuzzy Associative Memory of the system that controls the translation of the fabric, where the rule confidence is equal to one. The Fuzzy Associative Memory for the rotation is similar, but it is not presented due to the space limit.

		position error		
		small	***medium***	***large***
error rate	***small***	*very small*	*medium*	*very large*
	medium	*very small*	*medium*	*very large*
	large	*very small*	*small*	*large*

Table 1. Fuzzy Associative Memory (FAM) of the fuzzy system

It should be mentioned that for the implication process the 'min' operator is used, whereas for the aggregation process the 'max' operator is used and the defuzzification process is performed using the centroid method.

6. Control of the fabric's tensional force

Fig. 7 shows the hierarchical structure of the intelligent control scheme for the robotized tensional force regulation. This system communicates with the controller of a commercial robot by sending commands to adjust the robot end-effector velocity. In the higher level labelled "Decision making level", decisions about the system initialization are taken. This level incorporates two Fuzzy Logic decision mechanisms, where qualitative knowledge concerning fabric properties is processed.

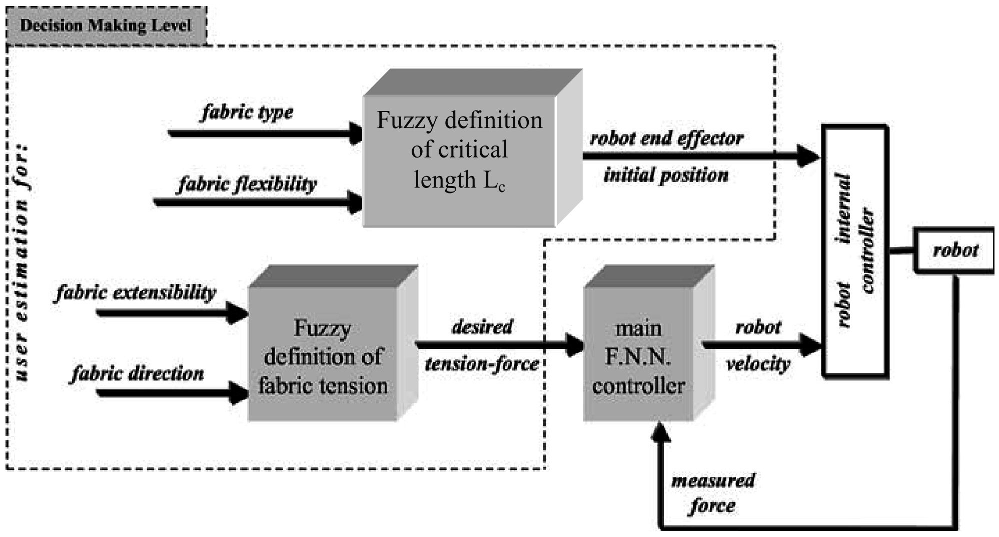


Fig. 7. Fuzzy decision mechanism and Neuro-Controller scheme.

The first part of this level labelled "Fuzzy definition of length L_c" is responsible for the determination of the initial end-effector position on the fabric as it has been described in details in the Pre-sewing tasks (No.5) in Section 0.
The second part labelled "Fuzzy definition of fabric tension" defines the desired tensional-force that should be applied to the fabric; and it is described in Section 0 in details.
In the lower level the "F.N.N. controller" outputs the robot end-effector velocity of the sewing process, by regulating the fabric tension.

6.1 Determination of the appropriate tensional force

The input to F.N.N. controller is the desired tension that should be applied to the fabric in order to meet the quality standards of the seam. Therefore, a specific force value has to be estimated for each type of the sewed fabric. Since the aim is to achieve a flexible system, an intelligent estimator of the desired tension-force is developed.
When the knowledge engineer acquires knowledge from the sewing experts, linguistic variables are used to describe the type of fabric, the necessary tension force applied to the fabric, the location of the hands, etc. Concerning the types of the fabric-cloth linguistic

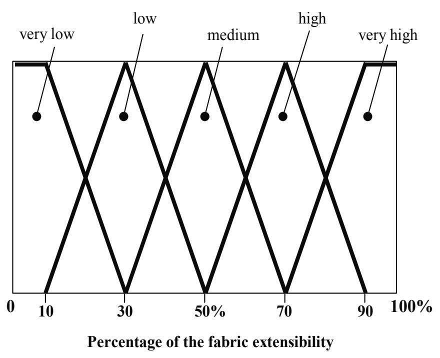
variables such as *soft*, *hard*, *extensible* or *very extensible* etc. are used. For the description of the applied tension-force similar linguistic variables are used: *low force*, *very low force*, etc.
This linguistic form is followed for the determination of the desired tension-force as the input to the F.N.N. controller. A number of expert seamstresses-tailors were interviewed, inquired and their work was investigated. The target of the queries was to acquire the knowledge concerning the way they handle different types of fabrics. In the questioning, the handling actions that were mostly investigated are related to the pushing or pulling each type of fabric and how much.
According to the experience of the seamstresses-tailors (i) the fabric is pulled during the sewing procedure so as to keep it outstretched, which means that only tensional force must be applied on the fabric and (ii) the appropriate tension-force for different fabrics depends on:

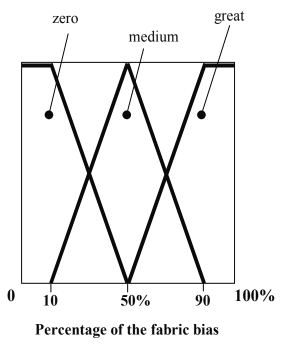
- The **extensibility** of the fabric. The experts delineate the extensibility of the fabric with linguistic variables such as: a cloth has medium extensibility, high extensibility, very high extensibility, low extensibility and very low extensibility.
- The **direction** of the fabric in which the "seam line" is performed (the bias of the fabric).

For "the fabric **extensibility**", the applied tensional force for different fabrics is expressed by the following general but essential rule:

"As much as more extensible the fabric is, so much higher tension-force is applied to it"

According to the experts this tension-force is expressed with linguistic variables such as medium, large, very large, small and very small tensional force.
Considering that the extensibility increases along the bias of the fabric as confirmed by Potluri & Porat (1996), the tension-force has to become higher in that "direction". Therefore, the expert applies "higher" tension-force to the same fabric when it is sewed along its bias, than the tension-force applied when it is sewed along the yarn direction. The maximum tension-force is applied while the sewing direction is 45° to the yarns.
The experts were not able to provide us with the specific values of the tension-force that they apply to each type of the fabric. Nevertheless, they produce high quality seamed products while they use linguistic variables in order to express the tension-force and they use the same linguistic variables when they are training new seamstresses-tailors.
The conclusion was that the decision concerning the determination of the appropriate tension-force that has to be applied to a sewed fabric incorporates fuzziness. Therefore, the fuzzy set theory was assumed as appropriate to be used. According to the acquired knowledge, the two input fuzzy variables "extensibility" and "direction" and one output called "tension-force" are defined. The membership functions and the corresponding linguistic values of these fuzzy variables are illustrated in Fig. 8 & Fig. 9.





(a) (b)

Fig. 8. The membership function of the fuzzy variable (a) "extensibility" and (b) "direction".

As it is illustrated in Fig. 8(a), for the fabric's extensibility five linguistic values are defined, while the universe of discourse is normalized in the percent scale, which indicates the expert's estimation about how much a fabric can be extended without puckering compared with other fabrics.
For the fabric's direction (bias) three linguistic values are defined (Fig. 8(b)), while the universe of discourse is normalized in the percent scale, which indicates the expert's estimation about the sewed direction on the fabric. The percentage of 100% means that the fabric is sewed at 45° to the yarns.
Similarly, the five linguistic values of the output variable "tension-force" are defined as it is shown in Fig. 9.
The rules derived after knowledge acquisition, are of the following form:

If extensibility <u>is</u> *x*
And direction <u>is</u> *y*
Then tension <u>is</u> *z*

and the FAM is presented in Table 2.
where, $x \in$ {*very low, low, medium, high, very high*}, $y \in$ {*zero, medium, great*} and $z \in$ {*very low, low, medium, high, very high*}.

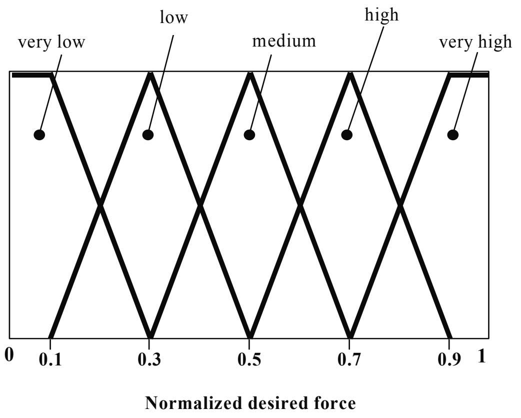


Fig. 9. The membership function of the fuzzy variable "tension-force".

		Extensibility				
		very low	*low*	*medium*	*high*	*very high*
Direction	***zero***	very low	very low	low	medium	high
	medium	low	low	medium	high	very high
	great	medium	medium	high	very high	very high

Table 2. Fuzzy rules for the desired tensional-force

The 7-steps process of a fuzzy system described by Zimmermann (1996) is followed: input, fuzzification, rules, rule evaluation, aggregation, defuzzification and output. For the rule evaluation phase the 'min' operation is used; and for the aggregation mechanism the 'max' operation. The centroid defuzzification method is used since this method is the most widely used and has several advantages (Cox, 1994).

6.2 The Neuro-Controller for the force regulation

Fig. 10 shows the tensional force controller based on Neural Networks. The performance of this scheme is considered successful, when the tensional force applied to the fabric is equal to the desired one, which is specified via the "Fuzzy definition of fabric tension" module.

The measured force representing the tension or compression of the fabric is the input to the F.N.N.; and the velocity command passed to the robot internal controller representing the velocity of the robot end-effector is the output of the F.N.N. The force input and the velocity output are normalized values, since the inputs or the outputs of a F.N.N. must take values in the range of [−1, 1]. From the normalized output of the F.N.N. the robot end-effector velocity is obtained.

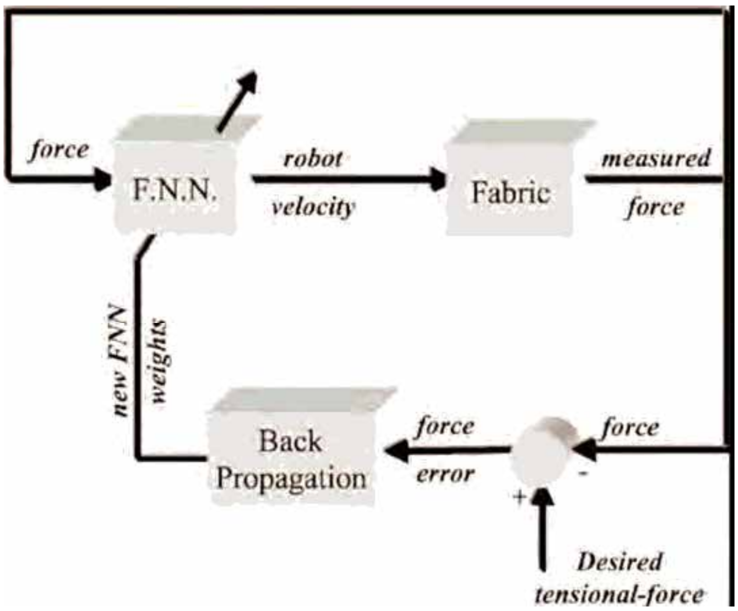


Fig. 10. The F.N.N. controller scheme.

The force error given by the difference between the desired and the feedback measured force is used in the backpropagation method for training the F.N.N.
The Feedforward Neural Network (F.N.N.) is chosen, because its topology is simple and it has been used successfully in the majority of the Neural Nets applications (Narendra, 1996; Sun, 1989). The backpropagation algorithm is used for the training of the formulated F.N.N. (Wasserman, 1989). It is well known that the number of the hidden neurons affects the performance of the F.N.N. In the present work, after a considerable number of experiments it was concluded that five hidden neurons are appropriate considering the quality of the response rate, the response smoothness and the overshooting.

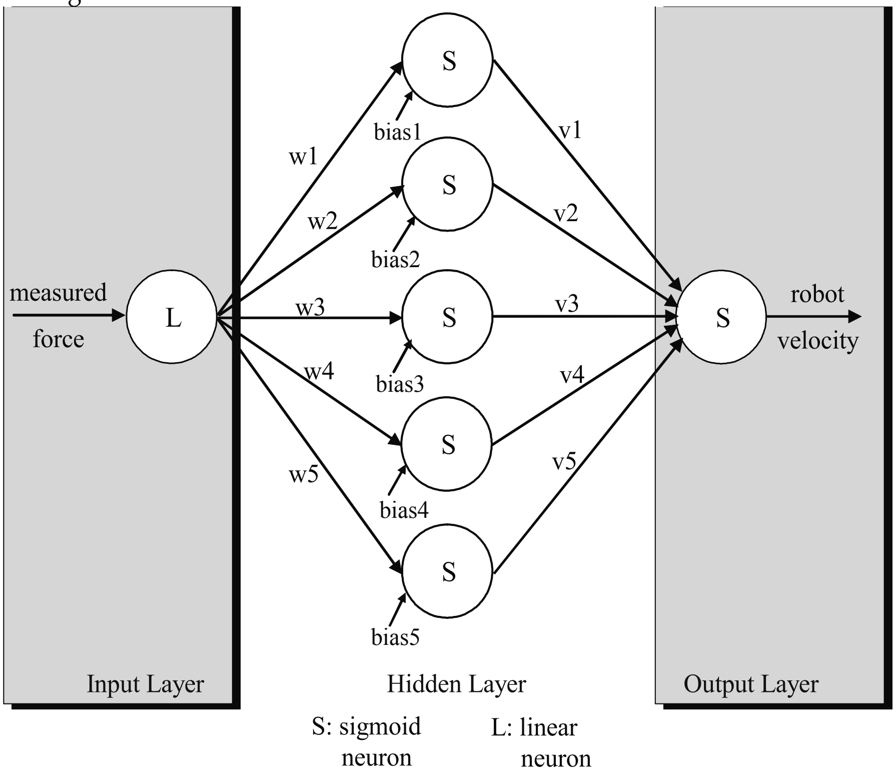
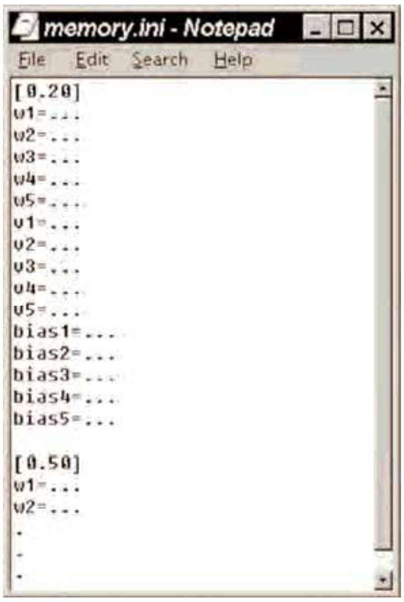


Fig. 11. The Neural Network topology.

The F.N.N. consists of three layers with the configuration (1-5-1), i.e. one neuron in the input layer, five in the hidden and one in the output, as Fig. 11 illustrates. In the hidden layer a threshold parameter is used for the five neurons in order to activate the neuron; and the associated activation function is the sigmoid one.
In the input neuron, the linear activation function is used (Wasserman, 1989), while in the output neuron a threshold parameter is used and the associated activation function is the sigmoid, since the velocity of the robot end-effector was assumed positive, i.e. the robot end-effector moves only forward.
The Neuro-controller scheme memorizes the trained F.N.N. controller characteristics for each of the fabrics that have been successfully handled, so the final adapted weights and bias are stored under a label into a file as shown in Fig. 12. The label of each set of weights-

bias reflects the inputs used to define the specific fabric type. This label corresponds to the desired tension-force that the Fuzzy decision making part of the system concludes as appropriate. If a new handling task with a piece of fabric is going to be executed, then the "memory" is searched to find a label identical or the closer one to the desired tension-force. If there is such a label, then the weights-bias under this label are uploaded to the F.N.N. as the initial values. Even in this case, the system does not remain insensitive, the F.N.N. still adapts its weights-bias to train itself and expand its experience (on-line and endless training). Finally, the weights and bias are stored under the same label by overwriting the old values, or under a new label if the closer to the desired tension-force label had been used to initialize the F.N.N.



```
[0.20]
w1=...
w2=...
w3=...
w4=...
w5=...
v1=...
v2=...
v3=...
v4=...
v5=...
bias1=...
bias2=...
bias3=...
bias4=...
bias5=...

[0.50]
w1=...
w2=...
.
.
.
```

Fig. 12. The "memory" file of the controller.

Instead of training the F.N.N. for a wide variety of fabrics, the technique of storing the F.N.N. characteristics for each type of fabric is followed to overcome some restrictions imposed by the nature of the training set and the structure of the F.N.N. The training set of the F.N.N. including all the appeared fabric types is not a uniform set, which causes difficulties to the generalization of the Network capabilities. Moreover, the F.N.N. has one neuron in the input as well as one neuron in the output; with this topology the network is capable to represent a unique function of the model of the controlled system. Considered that two different fabrics have two quite different functions representing their behaviour, the F.N.N. cannot be trained for both of these fabrics. If another F.N.N. structure would considered in order to train it with a larger fabric's training set, then this topology should be more complicated affecting the Network size and therefore the required computational time, which is very critical for the real time performance of the system.

The above description of the system "memorizing" is an approximation but still quite far from the actual seamstress-tailor train in order to sew a fabric that she/he has not be trained to do. Eventually, the utilization of the memorized characteristics by the controller resembles a continuous accumulation of experience.

7. Experimental results

In the following the simulated and experimental results of the two tasks (manipulation and force regulation) are presented. The specifications of the hardware as well as the limitations of the controller performance are described.

7.1 Manipulation results

The fabric manipulation experiments were carried out using a robotic manipulator with 6 rotational degrees of freedom (RV4A) and controlled by a PC AMD Athlon (tm) 64 Processor 3000 + 1,81 GHz running under Windows XP. The robot is programmed in Melfa-Basic language in Cosirop environment, while the analysis of the visual information is performed with Matlab 7.1., and each step, the resulted positional data are sent to the robot controller. The vision system consists of a Pulnix analogue video camera at 768×576 pixels resolution RGB with focal capabilities ranging from 1m-∞ and a TELE digital camera of the same resolution using Samsung zoom lenses. Both cameras are fixed above the working table in a vertical position, so that the fabric is kept in the field of view of the first camera during the servoing and the area near the needle is in the field of view of the second camera (Fig. 13). The cameras are connected to Data Translation Mach 3153 frame grabber through coaxial cables. A pointer fixed at the contact point-position was used instead of an actual sewing machine, since the effects associated with the actual sewing process is outside the scope of this work.

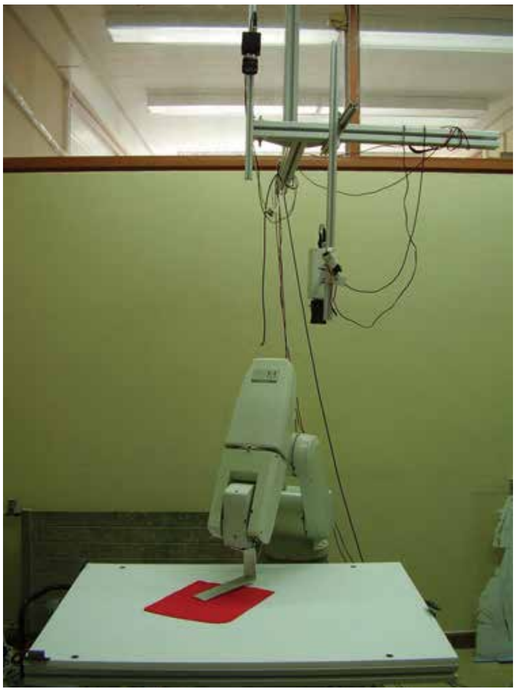


Fig. 13. The experimental stage.

To validate the feasibility of the proposed approach and evaluate the capability and flexibility of the system, a number of experiments were conducted. Firstly, the algorithm for the manipulation is tested for fabrics where the "seam line" is composed by straight-line segments. The handling process is repeated enough times for the same fabric and for various types of fabrics, where each time the fabric starts from a different location on the table and the gripper is attached to the fabric at a different location. The accepted position and orientation error are set equal to 2 pixels (≈ 0.25 mm) and 0.5° respectively. In the tested cases, the system shows robustness and efficiency. When the accepted position and orientation error were set to lower values, the fabric made many oscillations around the needle until the needle and the sewing line are reached.

Next, the system is tested for fabrics with arbitrary curved-line segments using the Genetic Algorithm approach. The maximum number of polygon edges approximating the curved section is set to 6 and the maximum acceptable deviation ε is arbitrarily set to 8 pixels (≈ 1 mm). The optimum polygon section, resulted from the Genetic Algorithm, which approximates the arc curve section of the fabric used for the experiments is shown in Fig. 14. The length of the arc section is about 5 cm and is approximated by a polygon section consisted of three sides and the maximum deviation for each straight-line segment from the corresponding arc section is 6.8722, 5.5480 and 7.0702 pixels, respectively.

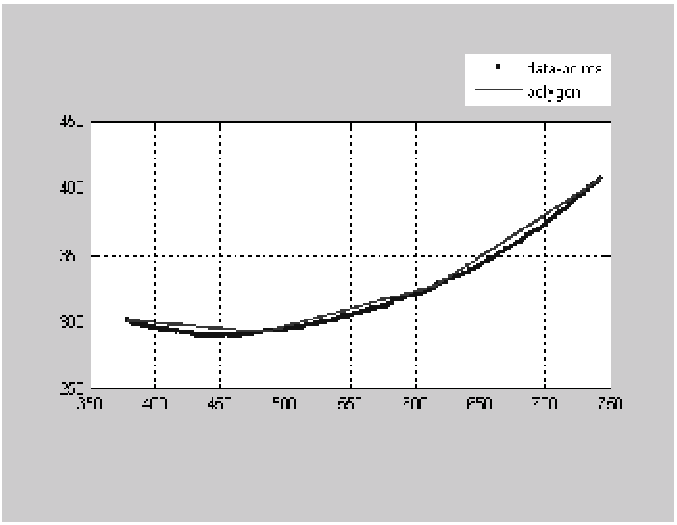


Fig. 14. Polygonal approximation with ε=8 pixels.

The maximum deviation (in pixels) between the needle-point and polygon approximation is computed from the captured image. The sewing process for the curved edge is successively accomplished and the results are shown in Fig. 15. The maximum deviation is 6.6323 pixels (≈0.83 mm) and is lesser than the maximum acceptable limit of 8 pixels. The average value for the deviation is 2.7501 pixels (≈0.34 mm), which is really acceptable. The steps 6-10 correspond to the part of the curve with the maximum curvature.

The experiments demonstrate the efficiency and robustness of the system. Fabrics that present relatively high bending rigidity were used for the experiments, so that the fabric remains almost unchangeable and flat, without considerable wrinkles or folds. However, wrinkles and folds generated during the manipulation of the fabric are supportable, in the sense that the system is capable of coping with them without failing to follow the required motion with acceptable accuracy.

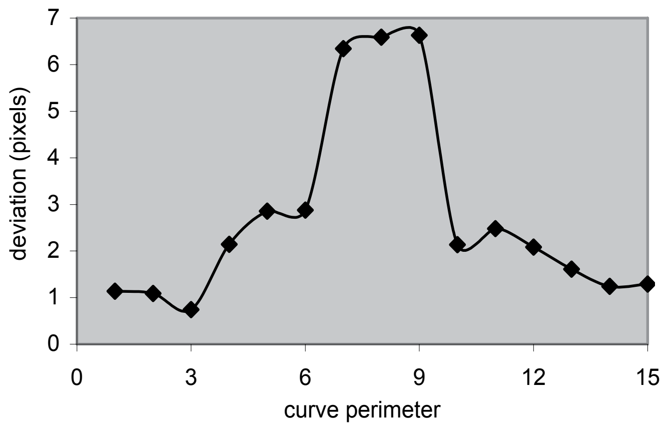


Fig. 15. Deviation between real and desired path.

7.2 Results of the tensional force regulation

The force controller is implemented and tested by simulating the sewing process. The simulation is implemented in a Visual Programming Language (Visual Basic); where the Neural Networks and the Fuzzy Logic algorithms were constructed.

The results presented in the following demonstrate the performance of the force control system. All the numerical data are normalized in order to feed them into the F.N.N. and all the outputs-results data are presented in normalized form.

For the needs of the performance of the controller the inputs "extensibility", "direction" and "flexibility" are specified:

(a) The fabric "extensibility" is assumed to be 20%, which is fuzzified as *very low* and *low* extensibility.

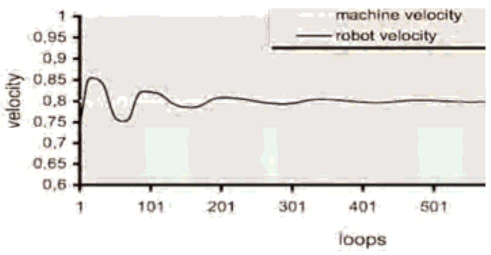
(b) The "direction" of the fabric yarn, in which the seam is performed, is assumed to be 10%. For the fuzzification the bias is labelled as *zero*.

(c) The fabric is assumed to be a knitted fabric with a "flexibility" equal to 70%. After the fuzzification the flexibility of this knitted fabric is labelled as *medium* and *large*.

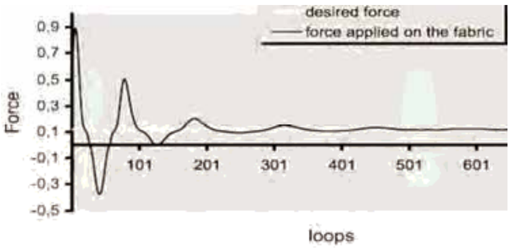
After the defuzzification the fuzzy decision system outputs the desired tension-force equal to 0.12, and the critical length L_c, equal to 20 cm.

For the requirements of the F.N.N. controller the weights and thresholds were initialized randomly in the interval $[-1, 1]$. This indicates that any former experience is not utilized, which means that the controller was initialized with the worst conditions. The updating rate parameter of the F.N.N., which is used in the backpropagation learning algorithm, is assigned equal to 0.7, and the sewing machine velocity for this test is assumed equal to 0.8.

In Fig. 16(a), the velocity of the robot end-effector provided by the F.N.N. controller, is compared with the sewing machine velocity. In this case, the controller reached a good approximation of the machine velocity after 150 training loops.



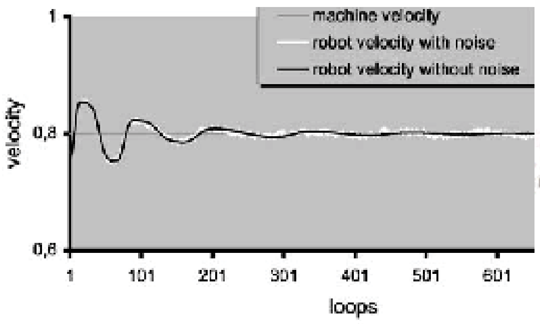
(a)



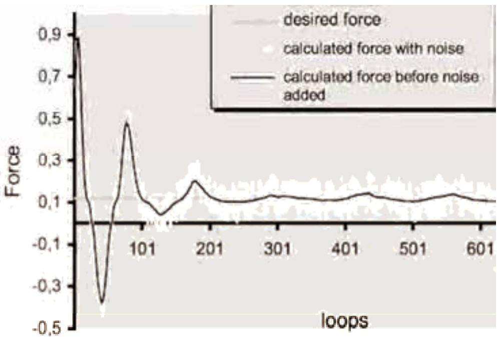
(b)

Fig. 16. Neuro-Controller's response: (a) machine and robot velocities, (b) desired and real force.

From the results shown in Fig. 16(b), it can be concluded that as the robot velocity approaches the machine velocity, the tensional force reaches the desired value (0.12).
The shapes of the "robot velocity" and "force" curves are expected and are qualitatively verified. When the sewing machine starts, the robot end-effector does not move with the appropriate velocity in order to follow the machine; therefore, the measured force is tensional and is increased rapidly as it is illustrated from the first loops in Fig. 16(b). At the same time the controller reacts to this increase of the force by adapting its weights and it derives a robot velocity greater than the machine velocity, in order to reduce the tensional force in the desired level.
Since the backpropagation training method modifies the F.N.N. weights continuously, after the last loop the error varies in the range of the third decimal for the force and of the seventh decimal for the end-effector velocity. These results were expected since a very low increase of the difference between the sewing machine velocity and the end-effector velocity results increases considerably the tensional force.
The controller is also tested for its performance in the presence of disturbances due to the noise interfering in the measured force. A noise of an amplitude ±10% was added to the applied force while the topology of the F.N.N. controller kept the same as presented in the previous test in order to compare the results under the same structure of the experimental stage. For comparison the velocity and force results are shown in Fig. 17(a) and (b) respectively without noise (black line) and with noise (white line) while the desired values are shown by grey lines.



(a)



(b)

Fig. 17. Neuro-Controller's response when noise is added: (a) velocities, (b) forces.

The controller can be characterized as robust, since it is capable to regulate the tensional force applied to the fabric to reach the desired value while approaching the sewing machine velocity. The small variations on the applied force, shown in Fig. 17(b) (black line), are the effects of the noise and have a trend to increase when the added noise increases. These variations are acceptable when the noise ranges in ±10% of the applied force; but the seam quality is deteriorated if the noise is out of the range of ±10%.
The verification of the system "memory" is performed through a simulated example. For a specific fabric, the proposed controller first operates with a "white memory", in terms of the F.N.N. weights-bias. The controller successfully adjusts the force as described in the first example with the results shown in Fig. 16(a) and (b); and the final values of the weights have been stored in the "memory" file under the label [0.12], which is the desired tensional-force. In other words the "experience" was acquired and amassed. In the next test, the controller operates with its embedded "experience" in order to determine the robot velocity for the same fabric. Therefore, the initial values of the F.N.N. weights are set-up using the stored values under the label [0.12]. In Fig. 18(a) and (b) the velocity of the robot end-effector and the force applied to the fabric are presented. By comparing the diagrams before (Fig. 16(a) and (b)) and after (Fig. 18(a) and (b)) the acquired "experience", it is clear that the controller's performance is improved noticeably using the previous "experience". The overshooting in both velocity and force response diagram is reduced about half, as well as the F.N.N. training time is reduced remarkably.

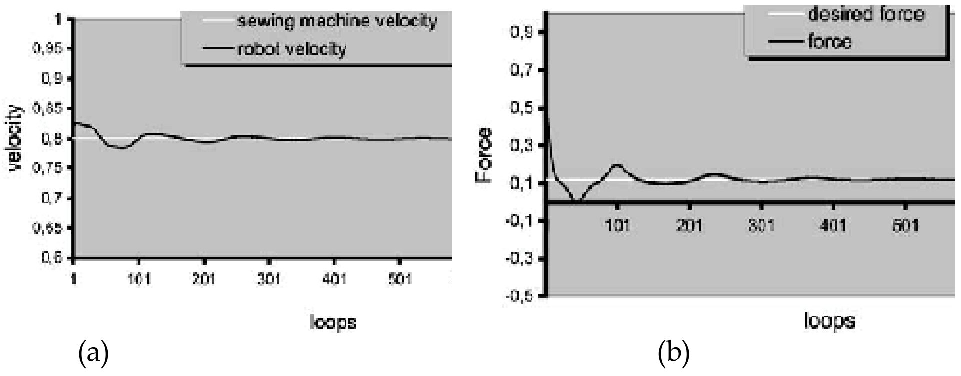


Fig. 18. Neuro-Controller's response when "experience" is utilized: (a) machine and robot velocity, (b) desired and real force.

The simulation is performed in a PC with an AMD Duron 700 Mhz processor. A complete training loop, shown in Fig. 10, is performed in 0.96 milliseconds. For the 150 training loops where the Neuro-Controller has achieved the desired tensional force, the elapsed time is 144 milliseconds. Considering that the usual sewing speeds ranges from 0.1 m/s to 0.3 m/s, the seam that is performed in this training time (144 milliseconds) has a length ranged from 14.4 mm to 43.2 mm. Therefore, it can be concluded that the system is capable to achieve the desired tensional force of the fabric from the firsts stitches of the sewing process. Eventually, the speed efficiency of the proposed system is restricted from the refresh range of the force sensor device and the I/O of the robot controller.

8. Conclusions

In this work, a system for robotic handling of fabrics towards sewing is introduced. A visual servoing manipulator controller based on fuzzy logic is designed in order to guide the fabric towards the sewing machine and produce a good seam quality. The experimental results show that the proposed approach is effective and efficient method in guiding the fabric towards the sewing machine, sewing it and rotating it around the needle. Unlike some of the methods referred in the introduction, the proposed controller does not need mathematical models or calculations, but it is proved to be rather simple and robust. It seems that, using intelligent methods the robotic system could be independent and autonomous in order to perform the sewing process for the majority of fabric types, without or little human assistance.

It should be stressed that the problem of sewing a piece of fabric using a robotic manipulator instead of a human expert poses additional problems, since it is more difficult for a robot to cope with buckling, folding or puckering. Considering the future research work, the proposed algorithms can be extended so that it can take into account the distortions presented during robot handling of the fabric. Finally, the integration of the systems in a single robotic workcell is the ultimate target.

9. Acknowledgments

This research work is carried out under the project "XPOMA-Handling of non-rigid materials with robots: application in robotic sewing" PENED 01 funded by the General Secretariat for Research and Technology of Greek Government. It is also integrated in the I*PROMS Network of Excellence.

10. References

Cox, E. (1994). *The Fuzzy Systems Handbook*, A.P. Professional, pp. 248–249

Gershon, D. (1990). Parallel Process Decomposition of a Dynamic Manipulation Task: Robotic Sewing, *IEEE Trans. on Robotics and Automation,* Vol. 6, No. 3, pp. 357-366

Gershon, D. (1993). Strategies for robotic handling of flexible sheet material, *Mechatronics*, Vol. 3, No. 5, pp. 611–623

Gershon, D. & Grosberg, P. (1992). The buckling of fabrics during feeding into automatic sewing stations, *J. Text. Inst.*, 83(1), pp. 35–44

Gershon, D. & Porat, I. (1986). Robotic sewing using multi-sensory feedback, *Proceedings of the 16th International Symposium on Industrial Robots,* pp. 823-834, Brussels

Gershon, D. & Porat, I. (1988). Vision servo control of a robotic sewing system, *Proceedings of the IEEE Conference on Robotics and Automation,* pp. 1830-1835, Philadelphia, April 1988

Gilbert, J. M., Taylor, P. M.,Monkman, G. J., and Gunner, M. B. (1995). Sensing in garment assembly, in: *Mechatronics Design in Textile Engineering*, NATO ASI Conference, Side, Turkey, pp. 291–308

Hutchinson, S.; Hager, G.D. & Corke P.I. (1996). A tutorial on visual servo control, *IEEE Transactions on Robotics and Automation,* Vol. 12, No. 5, pp. 651-670

Koustoumpardis, P. N. and Aspragathos, N. A. (1999). Control and sensor integration in designing grippers for handling fabrics, in: *8th Internat. Workshop on RAAD '99*, Munich, Germany, , pp. 291–296

Koustoumpardis, P. N. & Aspragathos, N.A. (2003). Fuzzy logic decision mechanism combined with a neuro-controller for fabric tension in robotized sewing process, *Journal of Intelligent and Robotic Systems,* Vol. 36, No. 1, pp. 65-88

Koustoumpardis, P. N. & Aspragathos, N. A. (2004). A Review of Gripping Devices for Fabric Handling, *IMG04*, pp. 229-234, ISBN 88900 426-1-3, Italy, July 2004,Genova

Kudo, M.; Nasu, Y.; Mitobe, K. & Borovac, B. (2000). Multi-arm robot control system for manipulation of flexible materials in sewing operation, *Mechatronics*, Vol. 10, No. 8, pp. 371-402

Monkman, G. J. (1996). Sensory integrated fabric ply separation, *Robotica*, 14, pp. 119–125

Narendra, K. S. (1996). Neural networks for control: Theory and practice, *Proceedings of IEEE*, pp. 1385–1406, 84(10)

Potluri, P. & Porat, I. (1996). Low-stress fabric testing for process control in garment assembly. Application of robotics, *Internat. J. Clothing Science Technology*, Vol. 8, No. 1/2, pp. 12–23

Stylios, G. (1996) Intelligent textile and garment manufacture, *Assembly Automation* 16(3), 5–6

Stylios, G. and Sotomi, J. O. (1996). Thinking sewing machines for intelligent garment manufacture, *Internat. J. Clothing Science Technology* 8(1/2), 44–55

Sun, Y. K. and Hwang, J.-N. (1989). Neural network architectures for robotic applications, *IEEE Trans. Robotics Automat.*, Vol. 5, No. 5, pp. 641–657.

Taylor, P. M. (1990). Sensory Robotics for the Handling of Limp Materials, *NATO ASI Series F64*, Springer, Berlin/Heidelberg,

Teh, C.H. & Chin, R.T. (1989). On the detection of Dominant Points in Digital Curves, *IEEE Trans. on Pattern Analysis and Machine Intelligence,* Vol. 11, No. 8, pp. 859-872

Torgerson, E. & Paul, F.W. (1988). Vision-Guided Robotic Fabric Manipulation for Apparel Manufacturing, *IEEE Control Systems Magazine*, Vol. 8, No. 1, pp. 14-20

Tsourveloudis, N. C., Kolluru, R., Valavanis, K. P., and Gracanin, D. (2000). Suction control of a robotic gripper: A neuro-fuzzy approach, *J. Intelligent Robotic Systems*, Vol. 27, No. 3, pp. 251–235

Wasserman, P. D. (1989). *Neural Computing Theory and Practice*, Van Nostrand Reinhold, NY

Zacharia, P.Th. & Aspragathos, N.A. (2005). Optimal robot task scheduling based on Genetic Algorithms, *Robotics and Computer-Integrated Manufacturing,* Vol. 21, No. 1, pp. 67-79

Zacharia, P.Th.; Mariolis, I.G.; Aspragathos, N.A. & Dermatas, E.S. (2005). Visual servoing of a robotic manipulator based on fuzzy logic control for handling fabric lying on a table, *1st Virtual International Conference on I*PROMS*, pp. 411-416, 4-15 July 2005

Zacharia^a, P.Th.; Mariolis, I.G.; Aspragathos, N.A. & Dermatas, E.S. (*2006*). Visual servoing controller for robot handling fabrics of curved edges, *accepted in Virtual International Conference on I*PROMS NoE,* 3-14 July 2006

Zacharia^b, P.Th.; Mariolis, I.G.; Aspragathos, N.A. & Dermatas, E.S. (*2006*). Polygonal approximation of fabrics with curved edges based on Genetic Algorithms for robot handling towards sewing, *CIRP, pp. 603-608, Ischia, Italy*, 25-28 July 2006

Zimmermann, H.-J. (1996). *Fuzzy Set Theory and Its Applications*, Kluwer Academic, Dordrecht

Zoumponos, G.T. & Aspragathos N.A. (2004). Design of a robotic air-jet gripper for destacking fabrics, *IMG '04*, pp. 241-246, Genova, Italy, 1-2 July 2004

Zoumponos, G.T. & Aspragathos N.A. (2005). A fuzzy robot controller for the placement of fabrics on a work table, *IFAC World Congress,* Prague, 4-8 July 2005

Complex Carton Packaging with Dexterous Robot Hands

Venketesh N. Dubey[1], Jian S. Dai[2]
[1]*Bournemouth University,* [2]*King's College London (United Kingdom)*

1. Introduction

Product packaging is one of the key industrial sectors where automation is of paramount interest. Any product going out to the consumer requires some form of packaging, be it food products, gift packaging or the medical supplies. Hence there is continuous demand of high-speed product packaging. This requirement is dramatically increased with seasonal consumer products and fancy gifts, which invite innovative designs and attractive ideas to lure the potential customers. Typically such products are delivered in cardboard cartons with delicate and complex designs. Packaging of such designs is not only tedious and challenging but also time consuming and monotonous, if manual methods are employed. For simple cardboard packaging fixed automation has been in use by dedicated machines along a conveyor belt system, however, these machines can handle only simple type of cartons; any change in shape and structure cannot be easily incorporated into the system. They require, in most cases, a change of more than 40 points (Dai, 1996a) to fit into the same type of carton of different dimensions, which means one specific type of cartons requires one packaging line. Capital expenditure increases with the change-over (Dai, 1996b) from one type to another type of carton folding assembly lines. Thus the flexibility is lost due to these limitations and the associated cost in the change-over. This research demonstrates the feasibility of designing a versatile packaging machine for folding cartons of complex geometry and shapes. It begins by studying cartons of different geometry and shapes and classifying them into suitable types and operations that a machine can understand. It conceptualizes a machine resorting to modeling and simulation that can handle such cartons, and finally developing the design of the packaging machine. It has been shown that such a versatile machine is a possibility, all it requires is miniaturization and investment on its development when such machines could be a reality. However, for practical implementation considerations need to be given for a compact, portable system incorporating sensors. The presented design is unique in its existence and has been shown to fold cartons of different complexity.

2. Design Requirements and Principles

Historically, manual methods have been used to complement adapting to different types of carton with changing styles. It takes about 10% of the work order and is called upon as assembly lines to take the order of promotional products. However, problem still exists with the manual lines with a large learning curve for both supervisors and operators, and with inconsistency and

labor injuries mostly due to hand twisting motions. Further, the manual line is generally considered to be the seasonal force that a dedicated machine still has to be used on a year-run basis to save cost and time. To make the task more difficult, the carton designers must pursue fantasy and originality in carton packaging to respond to a highly competitive market. The frequent change of style and types of carton and the small batches of production present a challenge to the carton assembly and packaging line, and demand a flexible machine to be designed. The onus is thus placed on the packaging industries to fully speed-up the change-over process for different types of carton with the aid of programmable and reconfigurable systems. Development of such agile and highly reconfigurable systems requires systematic analysis and synthesis of each component, i.e. cartons and their folding patterns, machine operating on them, and the complete assembly operation itself. One such approach (Lu & Akella, 2000) has been reported for folding cartons with fixtures. Whilst this approach generates all folding sequences for a carton, the implemented work just handles a simple rectangular carton for which fixed automation is already in place. For cartons of complex geometry, however, synthesis of both the carton and the folding mechanism needs to be considered together to achieve flexible automation in the assembly line. Extensive study has been undertaken by the authors on the operation of complex cartons folding pattern and its sequence analysis, resorting to graph theory, screw theory, matrix theory and representing cartons as a spatial mechanism; thereof studying the mobility and configuration analysis of the carton (Dai & Rees Jones, 1997a-c, 1999); (Dubey et al. 1999a-c, 2001).
This chapter presents the research undertaken to design a reconfigurable carton folding machine from design inception to the applications that can handle cartons of complex geometry and shapes.
This project was on the persuasive wish-list of many cosmetic and perfumery suppliers like Elizabeth Arden® and Calvin Klein® and has been under active consideration by the Unilever Research UK for several years. They were willing to support any research ideas to find some alternative means for automating the entire packaging process of fancy cartons, if at all it was possible. As a result the project was jointly sponsored by the UK and the Dutch Unilever consortium to explore the feasibility of developing a flexible packaging machine that can handle variety of cartons of different shapes and sizes. The research began with the study of manual packaging process to reveal that a higher degree of dexterity is required as we move from cartons of simple types to a complex one (Dai, 1996a). Such cartons are formed on a pre-broken cardboard sheet of different geometrical shapes. As the sheet is broken, it has a number of movable panels that can be rotated about the lines of crease. These crease lines facilitate the folding thus resulting in a shape transformation. Figure 1 shows an example of a fancy carton, which after folding, takes the final shape of the 'boy-scout-tent'; generally manual processes are resorted due to the complexity and typically small batch run of such cartons.

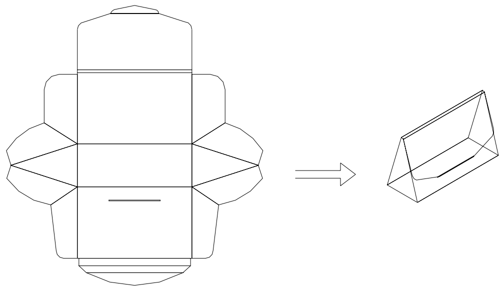


Fig. 1. Shape transformation in a fancy cardboard packaging.

In the manual process of packaging the side panels are bent using fingers since it requires folding along three axes as shown by the arrows in Figure 2, whilst the top and bottom panels are bent using the palms, shown by the flat jaws. The arrows representing the fingers apply a pointed force called 'poking' to the panels while the flat jaws apply 'pushing' forces. At the intermediate level of closing, fine motions are required by the fingers to cross-lock the panels by inserting the projected panels into the slitted areas called the 'tucking' operation.

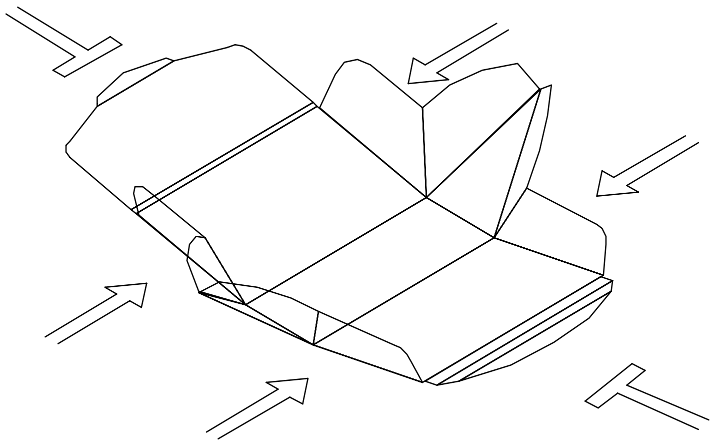


Fig. 2. Manipulative Forces in manual packaging.

In order to generalize the folding process, study of various cartons geometries were conducted that led to the following classification as shown in Figure 3; it also shows various operations involved in cardboard packaging. It is important to note that most cartons have to pass through the three stations of erection, insertion and closure, however, except for the tray-type cartons, the closing stations in other cases involve various manipulative functions (some of which are indicated above), depending on the complexity of the carton geometry.

For designing a mechanical system to erect and fold such cartons the following points need to be considered:

- Multiple functions: to provide various manipulative functions including poking, tucking, squeezing and twisting operations.
- Dexterity: to reach the manipulative positions in different ways.
- Minimum number of axes to control: to reduce the complexity of the system.
- Reconfigurability: to handle variety of foldable cartons with different geometrical configurations.
- Programmability: to control motions on many axes simultaneously and sequentially.

A flexible system to provide fine motions and manipulative functions requires that articulated finger-like links be used. The reconfigurability of the system to handle cartons of different shapes and sizes can be ensured by mounting such fingers on a movable base, such as XY-table or circular tracks. The architecture of the controller should be capable of moving each axis independently. The design should provide all manipulative functions to the fingers without making the system complex, thus resulting in a cost effective solution.

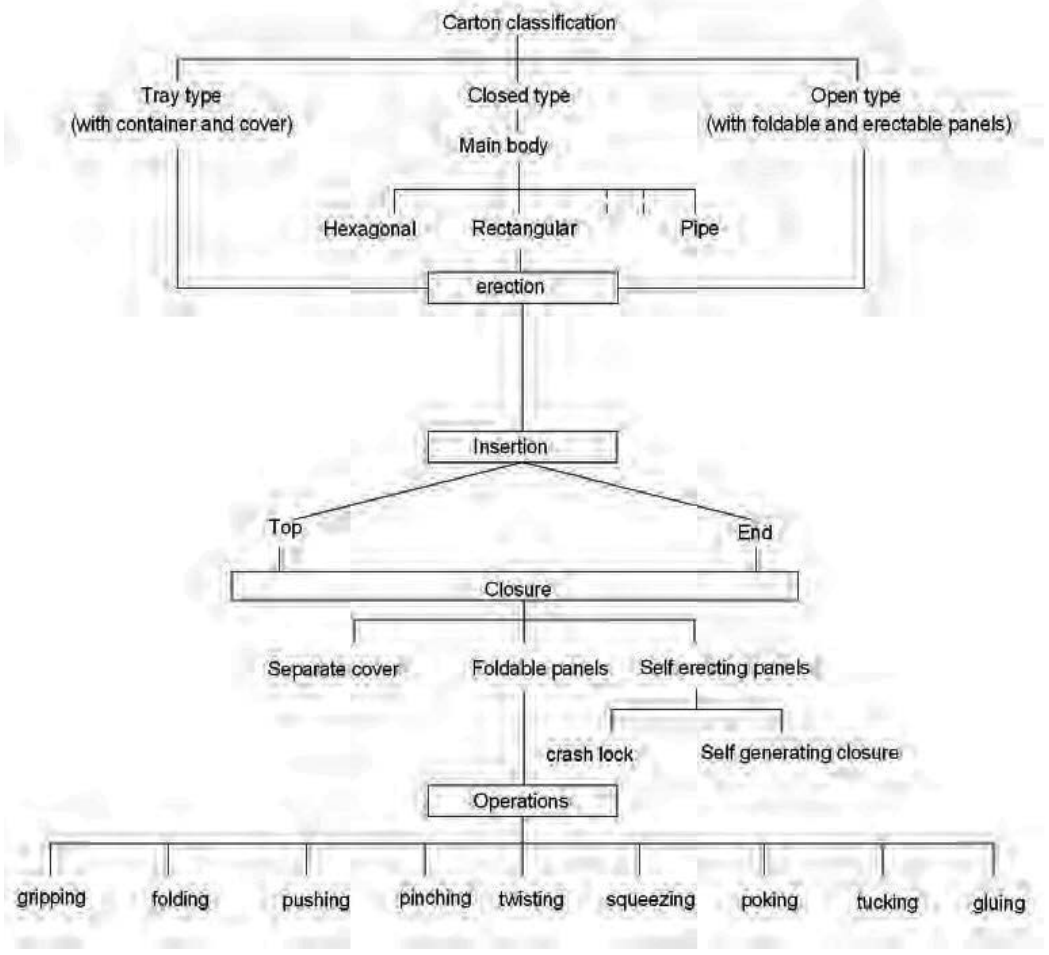


Fig. 3. Operations involved in cardboard packaging.

3. The Conceptual Design

The conceptual design based on the above criterion is shown in the Figure 4 (Dubey, Dai and Stamp 1999a). The design has four fingers, two with three degrees of freedom, and two with two degrees of freedom each. The three-degree-of-freedom fingers provide a yaw (Y) motion at the base and pitch (P) motions on the following two joints forming a Y-P-P configuration. The two-degree-of-freedom fingers have only pitch motions allowing it to move on a planar surface. These fingers are mounted on linear tracks that can move orthogonally and can be oriented at any angle at their base. Two horizontal jaws shown in the figure are arranged with the pushing direction parallel to the fingers' horizontal tracks. The plates attached to the jaws and are to be mounted on passive joints that are under-actuated i.e. during the pushing operation, they can orient along the shape of the cardboard panels. The base of the system, on which the carton is attached, has a circular turntable that can rotate as well as move vertically up and down, thus allowing any orientation and elevation of carton to be achieved that may be required during the packaging operation. These considerations were specifically based on providing high degree of reconfigurability with minimum number of axis to be controlled. The design as conceptualized is based on agility and versatility of the human hand which is capable of performing variety of dexterous functions.

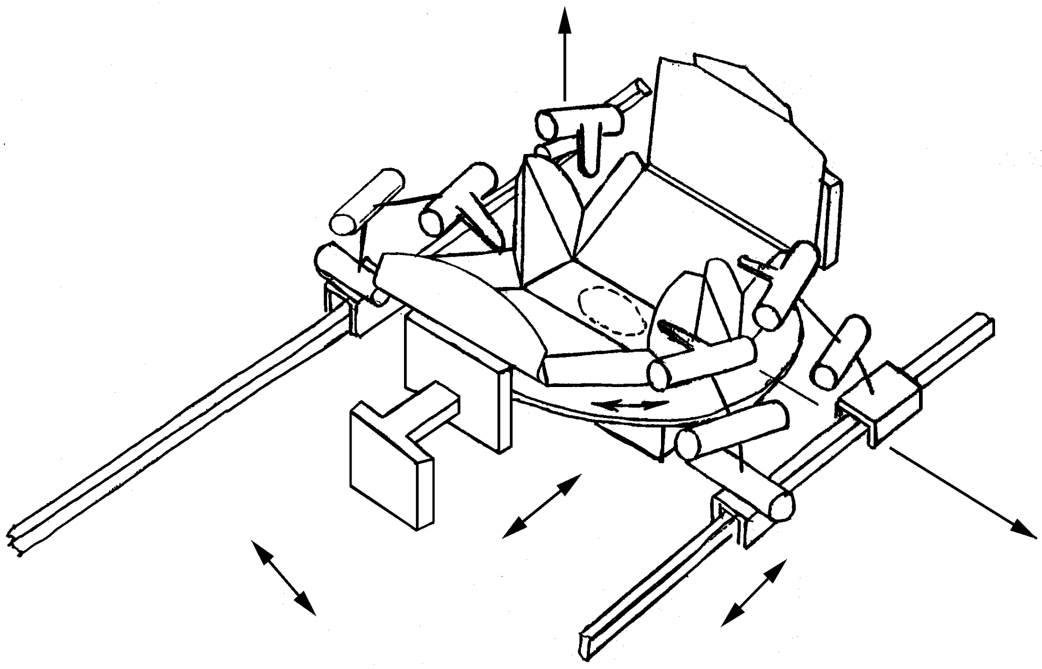


Fig. 4. Conceptual design of the packaging machine.

Based on the conceptual design a model was developed in a robotic simulation environment (Workspace4, 1998) with fingers and horizontal pushers as shown in Figure 5. The base of the small rectangular table at the center, which holds the carton is shown to be actuated by a single motor for vertical movement as well as for the rotation of the table, however, in actual system these two axes will be controlled independently. As far as simulation is concerned, this model will provide all the kinematic information that may be required to run the machine. The fingers are seen to be mounted on tracks, which can slide along the track and the track itself can move laterally. Further the fingers are mounted on swivel-base that allows these axes to be oriented suitably. The joints of the fingers are actuated directly by joint-motors and the whole system has 14 controlled axes. Special considerations have been given to design the fingertips, since they are required to perform various manipulative functions as discussed in the previous section. Inspired by the manual packaging processes, fingertip design incorporates pointed tip with V-groove to apply 'poking' and 'squeezing' effort to the cardboard panels as required in the tucking operation. The pointed tip is used for poking operation while the V-groove holds the common crease line between the panels for squeezing to allow the panels to open up for the tucking operation. The Y-shape of the two-degrees-of-freedom finger is to provide occasional pushing force on flat panels in addition to the poking and squeezing forces. Thus the design can provide the flexibility of offering many manipulative functions with limited degree of freedom to handle different types of cartons in different configurations.

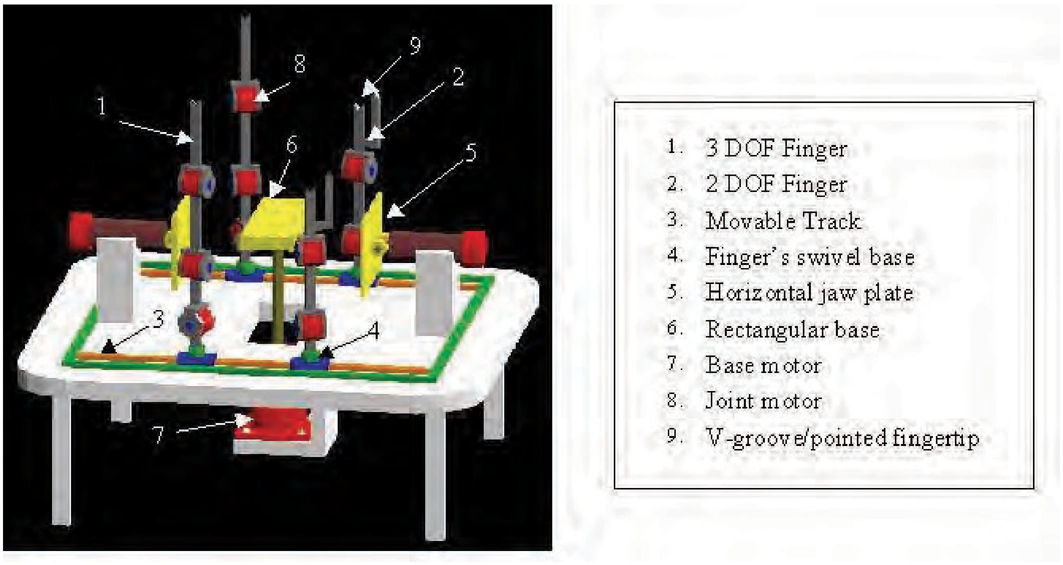


Fig. 5. Model of the packaging machine.

4. Simulation of the Packaging Process

In order to simulate the packaging processes, kinematic model of carton as well as packaging machine components (fingers, pushers and the turn-table) need to be developed and integrated together. The kinematic model of the carton has been studied (Dubey et al., 1999); (Dubey & Dai, 2001), which is governed by two basic angles (β and γ) as shown in the Figure 6, and the following equations (1-2) result.

$$\beta = cos^{-1}\left(\frac{R_1\, C\phi - R_3\, S\phi}{C\alpha}\right) \tag{1}$$

$$\gamma = cos^{-1}\left(\frac{(R_1\, S\phi + R_3\, C\phi)C\alpha\, S\beta + R_2\, S\alpha}{S^2\alpha + (C\alpha\, S\beta)^2}\right) \tag{2}$$

where,

$$R_1 = (C^2\delta\, V\theta + C\theta)C\alpha - C\delta\, S\delta\, V\theta\, S\alpha\,,$$

$$R_2 = (S^2\delta\, V\theta + C\theta)S\alpha - C\delta\, S\delta\, V\theta\, C\alpha\,,$$

$$R_3 = S\delta\, S\theta\, C\alpha + C\delta\, S\theta\, S\alpha\,,$$

$$V\theta = (1 - C\theta)\,,\ S = \sin\,,\ C = \cos\,.$$

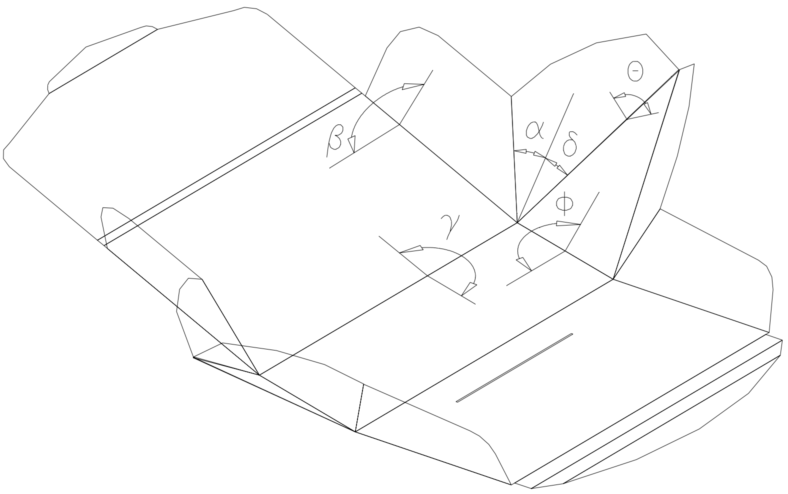


Fig. 6. Defining the kinematic model of the carton.

These equations completely define the folding sequence of the panels for proper shape transformation. If the folding of the carton is to be achieved by a packaging machine, each movable element of the machine needs to be connected kinematically. This requires inverse kinematic solution of the fingers to be developed. The 2-dof planar finger has standard solutions available (Fu et al., 1987), but solution for the 3-dof fingers with Y-P-P configuration is not very common; the closed form solution for which is given by the following equations (3-5), which need to be implemented into the controller for tracking the carton-panel movement.

$$\theta_1 = \tan^{-1}\left(\frac{p_y}{p_x}\right) \tag{3}$$

$$\theta_3 = \cos^{-1}\left(\frac{p_x^2 + p_y^2 + p_z^2 + l_1^2 + 2l_1(p_x c_1 + p_y s_1) - l_2^2 - l_3^2}{2l_2 l_3}\right) \tag{4}$$

$$\theta_2 = \sin^{-1}\left(\frac{(c_1 l_2 + l_3 c_1 c_3)p_z - (p_x - l_1 c_1)s_3 l_3}{(l_2 + l_3 c_3)(c_1 l_2 + l_3 c_1 c_3) + l_3^2 c_1 s_3^2}\right) \tag{5}$$

where px, py, pz are the co-ordinates of the target point, c and s are sine and cosine of the angle, l1, l2, l3 are the link lengths of the finger corresponding to joint angles 1, 2 and 3 moving from base to the tip.

The kinematic connectivity between carton and the fingers can be achieved by locating various contact points on the carton and recording the displacement of these points as the folding of the carton takes place. The contact points on the carton can be identified by geometrical interpretation of the folding sequences (Dubey & Dai, 2001). These contact points are then used to find joint displacement for each finger's joints. The displacement data are further interpolated to generate optimal finger paths minimizing the unwanted fingers motion and thus reducing the packaging cycle-time. The interpolated data from the simulation can be downloaded to drive the actual fingers. This whole packaging process can also be automated based on the study of the geometric features of the panels and their

folding sequences without resorting to the simulation of the carton; this is our current research effort. The model provides all the kinematic information that may be required to run the machine (Dubey & Crowder, 2003). A parametric model of the packaging machine was developed in the software (Workspace4, 1998) that allows geometrical and dimensional alterations in the design to be easily incorporated including the configuration verification. This also allows kinematic parameters of the machine-components to be ascertained before the actual fabrication.
Figure 7 shows the fingers tracking the contact points over the carton while it is folding. The simulation provides many valuable information for design as well as control of the packaging machine. For example, the simulation can be used for geometric as well as configuration checks before deciding on the dimension and structure of the machine. Any new geometrical information of the machine components can be directly obtained from the model by just making parametric changes in the basic dimensions of the model. Motion-data and trajectory of the fingers so obtained during folding of the carton panels can be used for finger control in the actual system. Currently the motion parameter available from the simulation has not been directly integrated to the actual controller thus the data has to be fed off-line in the form of a data-file. Nevertheless, this technique allows full validation of folding sequence and then downloading of these data to the controller.

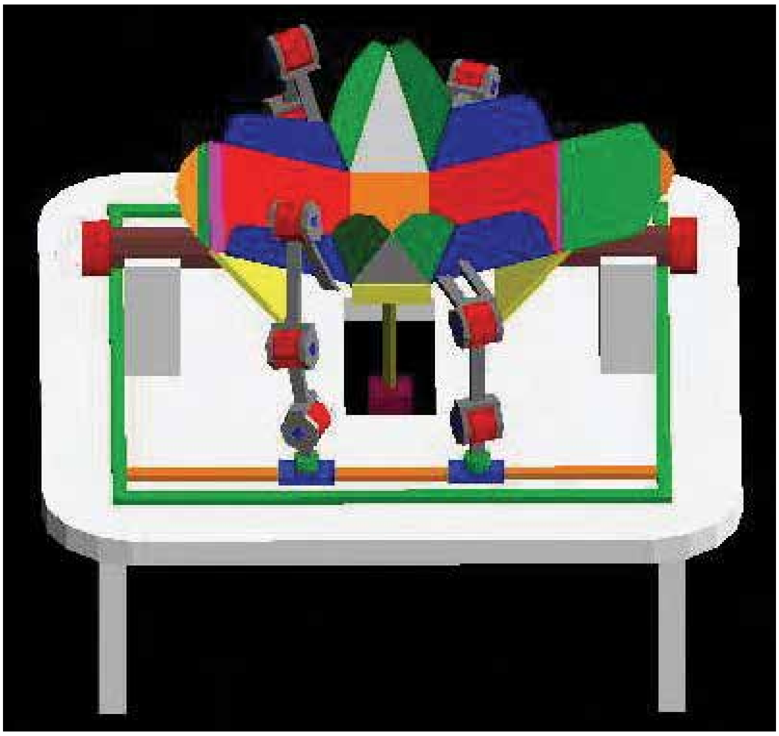


Fig. 7. Carton folding operations with the Fingers.

5. The Packaging Machine

Using the dimensional information from the simulation the packaging machine was developed which uses three linear motors; two for jaw-pushers and one for the vertical motion of the turn-table (Dubey & Dai, 2006). Ten high torque, high performance motors were used on the finger joints supplied by the Yaskawa Motors®, Japan, specifications of these motors were:

Dimension: □30□30 mm
Weight: 70 grams
Torque: 0.7 Nm at 22.5 rpm
Gear ratio: 80:1, harmonic drive

Optical encoder with 96 pulse/rev.
This means a 10 cm long arm connected to the motor can apply a fingertip force of 7 N, which is high enough to fold the cardboard panels along the crease lines. The controller architecture of the system is shown in Figure 8. It has four motion control cards and each can control up to 4 axes (Dubey & Crowder, 2003). These cards are supported by motion control library in C-programming language; it also has a G-code programming interface for quick trial and running motors in teach-mode. The system also incorporates pneumatic connections for attaching suction cups, which can be activated in ON/OFF position by the controller. Currently this is employed to move the turntable from one orientation to the other and to hold the carton in fixed positions, however, at later stage it is planned to use these cups on the fingertips as well, for astrictive mode of grasping. This will have an advantageous effect in handling flat carton panels without any slip (Dubey et al., 1999).

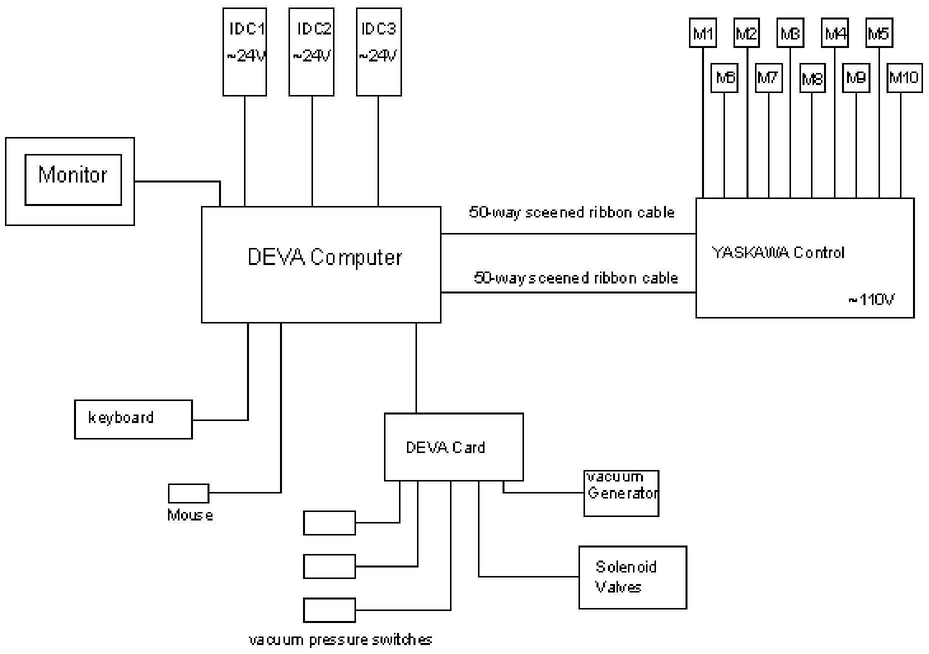


Fig. 8. The controller architecture.

In order to establish the capability of the packaging system for performing erection and folding of cartons, the data-file generated from the simulation was fed to the controller after ensuring that the both model and the machine have geometric and configuration equivalence. The data-file contains the motion data in a single row for simultaneous operation of the motors, whereas the succeeding lines have next stage motion control parameters. Accordingly, the controlling program reads the data file sequentially and generates appropriate interrupts within, to pass the motion commands simultaneously. Thus the fingers can duplicate motion in parallel as well as sequential modes. The programming capability of the controller can be further enhanced by developing sub-routines specific to various manipulative functions, thus allowing modular structure of the controller to be achieved, which can be easily adapted to any new carton folding and packaging sequences.

Reconfigurability of the system was one of the key issues for which this system was developed. The idea was to use the system to fold different type of cartons with little alteration in the system design. But to achieve this, it is important to ensure that the system is exactly configured as per the developed graphical model and the basic structural set up. To run the machine, the data file is included in the main program and the system is first operated in a stepwise trial-mode to verify the fingers' movement whilst the carton is being folded. Once this is achieved the system is ready for automated packaging. The developed machine is shown in Figure 9 where all fingers and horizontal pushers are involved in a complicated sequence of tucking operation. The system was able to erect and fold the carton (Figure 1) successfully in less than 45 seconds. Although it was difficult to give a comparative timing for manual packaging, we conducted some trial run for the same carton. The manual packaging on average (after learning) took around 60s.

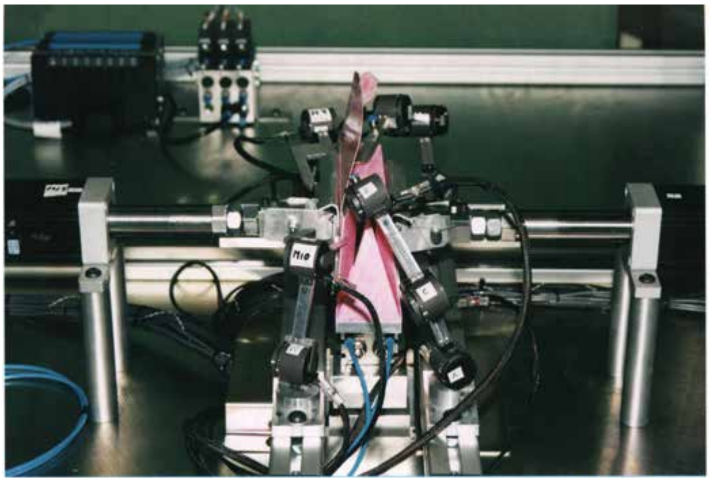


Fig. 9. The packaging machine in action.

Reconfigurability of the machine was demonstrated by folding a second type of carton as shown in Figure 10. This carton is a closed-type of completely different shape and complexity with multiple flaps. This requires various manipulative functions of pocking, twisting and tucking to be performed on the panels. After reconfiguring the structure of the machine, the folding sequences took a cycle time of 45 seconds to complete the carton. The reconfigured machine can be compared with the previous setup in Figure 9. This demonstrates the highly flexible and reconfigurable design of the system.

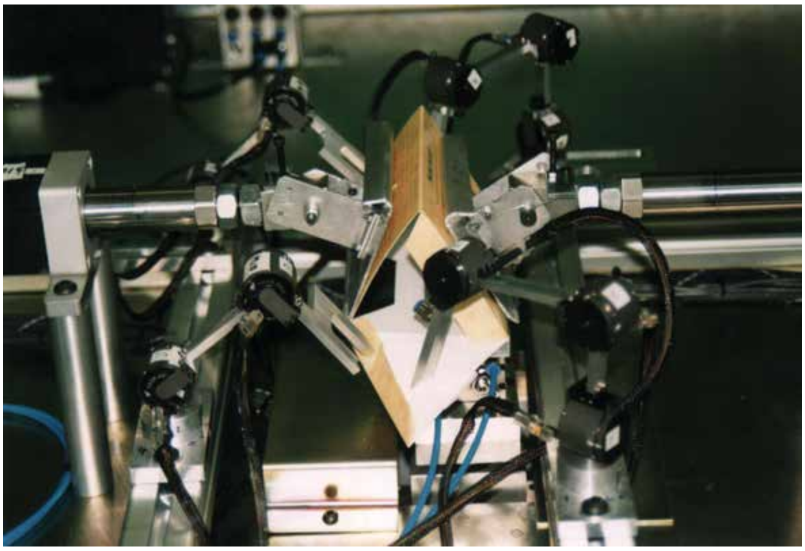


Fig. 10. The packaging machine folding a different carton.

6. Conclusions

The chapter has presented a dexterous reconfigurable assembly and packaging system (D-RAPS) with dexterous robot fingers. The aim of this research was to design a reconfigurable assembly and packaging system that can handle cardboard cartons of different geometry and shapes. The initial idea was to develop such a system that can demonstrate adaptability to cartons of different styles and complexities. It was shown that the packaging machine could fold two cartons of completely different shapes. The cycle time for the folding was approximately 45 seconds in each case. Though this is not an optimized time for folding, it is envisaged to reduce the cycle time to 30 seconds or less with on-line data transfer. Although there are many issues that need to be addressed before a fully flexible machine can be realized on the shop floor, nevertheless, the research was aimed at proving the principle of quick change-over, which faces the packaging industry.

The future enhancement will include optimization of finger trajectory, use of tactile sensors for force feedback to avoid excessive pressure on the panel, and astrictive mode of grasping to involve the vacuum system at the fingertips. It is also proposed to integrate the simulation model directly with the actual machine to download the motion data online. The XY-table can be motorized and controlled for auto-reconfiguration. These advanced techniques will automate the entire packaging process starting from the two dimensional drawing of the cardboard, defining its kinematics then generating the motion sequences leading to the finished product packaging. It is also envisaged to mount such dexterous reconfigurable hands directly onto a robotic arm to offer higher level of flexibility in packaging, if this can be miniaturized. The system will not only perform carton folding but can also be used to insert the product into the carton during the folding sequences. This will reduce the packaging time and will also be able to meet challenges of high adaptability to ever changing demands of high-end personal product packaging.

7. Acknowledgment

This research was supported by a grant from the Unilever Research, Port Sunlight, Wirral (UK), which the authors highly appreciate. The authors also thank Prof John Rees Jones and Dr Kevin Stamp for their discussions and advices.

8. References

Dai, J.S. (1996a). Survey and Business Case Study of the Dextrous Reconfigurable Assembly and Packaging System (D-RAPS), Science and Technology Report, No PS960321, Unilever Research (UK).

Dai, J.S. (1996b). Conceptual Design of the Dextrous Reconfigurable Assembly and Packaging System (D-RAPS), Science and Technology Report, No PS960326, Unilever Research (UK).

Dai, J.S. and Rees Jones, J. (1997a). New Models for identification of distinct configurations of Cartons during Erection and Assembly, Science and Technology Report, PS 97 0066, Unilever Research (UK).

Dai, J.S. and Rees Jones, J. (1997b). Structure and Mobility of Cartons through Screw Theory, Science and Technology Report, PS 97 0067, Unilever Research (UK).

Dai, J.S. and Rees Jones, J. (1997c). Theory on Kinematic Synthesis and Motion Analysis of Cartons, Science and Technology Report, PS 97 0184, Unilever Research (UK).

Dai, J.S. and Rees Jones, J. (1999). Mobility in Metamorphic Mechanisms of Foldable/Erectable Kinds, ASME Journal of Mechanical Design, Vol. 21, No. 3, pp. 375-382.

Dubey, V. N. and Dai, J. S. (2006). A Packaging Robot for Complex Cartons, Industrial Robot, Vol. 33, No. 2, pp. 82-87.

Dubey, V.N. and Crowder, R.M., (2003). Designing a dexterous reconfigurable packaging system for flexible automation, ASME Design Engineering Technical Conference, DETC2003/DAC-48812, Chicago, Illinois (USA).

Dubey, V.N. and Dai J.S. (2001). Modeling and Kinematic Simulation of a Mechanism extracted from a Cardboard Fold, International Journal of Engineering Simulation, Vol. 2, No. 3, pp. 3-10.

Dubey, V.N., Dai, J.S. and Stamp, K.J. (1999a). Advanced Modeling and Kinematic Simulation of a Complex Carton, Science and Technology Report, Unilever Research, (UK).

Dubey, V.N., Dai, J.S. and Stamp, K.J. (1999b). Advanced Modelling, Design and Experimental Work of a New Reconfigurable System for Packaging with a Mult-fingered Robot Hand, Science and Technology Report, Unilever Research, (UK).

Dubey, V.N., Dai, J.S., Stamp, K.J. and Rees Jones, J. (1999). Kinematic Simulation of a Metamorphic Mechanism, Tenth World Congress on the Theory of Machine and Mechanisms (IFToMM), pp. 98-103.

Fu, K. S., Gonzalez, R. C. and Lee C. S. G. (1987). Robotics: Control, Sensing, Vision, and Intelligence, McGraw-Hill International, New York.

Lu, L. and Akella, S. (2000). Folding cartons with fixtures: a motion planning approach IEEE Transactions on Robotics and Automation, Vol. 16, No. 4, pp. 346-356.

Workspace 4, (1998). User's Guide, Robot Simulations Ltd., UK.

Arc Welding Robot Automation Systems

Beom-Sahng Ryuh[1], Gordon R. Pennock [2]
[1]*Division of Mechanical Engineering, Chonbuk National University, Jeonju, Republic of Korea, 561-756*
[2]*School of Mechanical Engineering, Purdue University, West Lafayette, Indiana, 47907-1288, USA*

1. Introduction

Robot automation systems are rapidly taking the place of the human work force. One of the benefits is that this change provides the human work force with the time to spend on more creative tasks. The highest population of robots is in spot welding, spray painting, material handling, and arc welding. Spot welding and spray painting applications are mostly in the automotive industry. However, arc welding and material handling have applications in a broad range of industries, such as, automotive sub-suppliers, furniture manufacturers, and agricultural machine manufacturers.
The number of arc welding automation robot stations is growing very rapidly. The two most common stations are the GMAW (Gas Metal Arc Welding) station and the GTAW (Gas Tungsten Arc Welding) station. These two stations are the most common because they are so well suited to robot systems. Typically, a robot arc welding station is comprised of a robot, a robot controller, arc welding equipment, a work clamp and motion devices to hold work pieces accurately in position (considering heat deformation), robot motion devices to move around the robot for a larger working range and better weld positions, sensors, and safety devices. A typical arc welding robot station is illustrated in Figure 1.

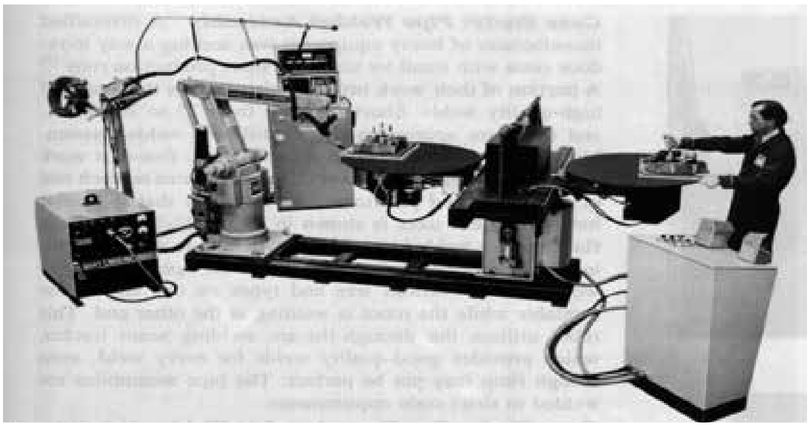


Fig. 1. A Typical Arc Welding Robot Station.

2. A Description of the System Components

2.1 The Robot and the Controller

A robot is programmed to move the welding torch along the weld path in a given orientation. The robot is typically comprised of a large number of links and linkages, which are interconnected by gears, chains, belts, and/or screws. The majority of industrial robots are actuated by linear, pneumatic, or hydraulic actuators, and/or electric motors. Most of the high-end robots currently use AC servo motors which have replaced the use of hydraulic actuators and, more recently, DC servo motors. AC servo motors are essentially maintenance free which is very important in industrial applications.

In an arc welding robot system, the torch is attached to the wrist of the robot which has two or three axes of motion. As technology develops, however, there is less application for a robot with a two-axis wrist motion. In the case of three-axis motion, the motion is composed of yaw, pitch, and roll, similar to the human wrist. The robot has the most effective motion when the center point of the wrist is aligned with the center line of the upper arm. A robot with a three-axis lower arm and a three-axis wrist will permit the torch action that is necessary for a complicated three-dimensional welding process. The torch can satisfy all the angle requirements, such as, the work angle, the transverse angle, the travel angle, and the longitudinal angle.

Important factors, when considering manufacturing performance are the frequency of failure, the mean time between failure (MTBF), the average time for repair, and the time for robot replacement. Also, with regards to system design, important issues are the robot work envelope, the reach of the robot tip, the number of joints (i.e., the degrees of freedom), the travel velocity, and the repeatability, accuracy, and resolution of motion.

The controller is the brain of the robot arc welding system. This is because the controller stores the robot programming and arc welding data, and performs the necessary computations for robot control, typically by a high-speed microprocessor. The controller provides a signal to the actuators and the motors by programmed data and position, speed, and other information obtained from various sensors. The controller is now integrated to govern not only the robot but also any peripheral devices, such as manipulators. When the system is required to weld a work piece that has a complicated geometry, the simultaneous coordinated control of the integrated controller is inevitable.

Memory backup devices, such as, a floppy disc drive, are recommended for storing important data as a safeguard in the event of a break down with the controller. In the case of a power failure, or some other unforeseen event, storing the data from an absolute resolver, or encoder, in controller memory will ensure that the robot can restore the programmed position without returning to the zero configuration.

2.2 The Welding Equipment (Power Source)

The welding equipment generates power to generate the arc for welding. One of the most important characteristics is stability of power. It is recommended that the welding equipment generates a short arc with less spatter for a good welding quality even at high speeds.

The arc sensor detects the current value so that the power source can supply the correct amount of power to the wire feeder, which then controls the wire feeding speed. The wire feeder has wheel rollers to advance the wire. Some feeders have four rollers speed sensors for more accurate wire feeding by push-pull action. Also, a wire feeder with shorter length to the torch is better in terms of a response time. Therefore, a good location for the wire feeder for a robot system is at the end of the upper arm of the robot.

A slender welding gun is better for maneuverability but, in case of a collision, sufficient strength must be guaranteed. It is also necessary to ensure that the torch is equipped with shock absorption devices such as springs. It is also important to have a cooling system (a circulating water, in general) to protect the torch against heat deformation. All the connections for welding, such as, electric power, the wire, and the coolant are usually integrated into one cable unit. It is recommended that the cable unit be as short as possible for a quicker response and a better reliability.

2.3 Manipulators

A robot has a limited working range and accessibility, therefore, in many cases a manipulator has to be considered. A manipulator is a device holding the work piece and is moved around (typically with linkages) for better access and welding positions. The advantages of a manipulator include:

(1) A manipulator can easily be moved around the work piece for the best welding positions.
(2) A manipulator can reduce the variation in the lead and the lag angles of the tip.
(3) Welding can be performed in a stable flat welding position by a synchronized and simultaneous control of a robot and a manipulator.
(4) Any hard-to-reach positions can be accessed more easily.
(5) A manipulator increases the working range of a fixed floor mounted robot or an inverted robot.

In general, a robot can maintain a better flat welding position which will produce a better deposition and, thereby, reduce any repair work by the cooperation with a manipulator. This also makes possible higher welding speeds and, thereby, an increase in productivity.

There are two types of actuation systems for manipulators; namely, the indexing type system, and the servo-controlled system. The indexing type system is for economic models and is commonly actuated by pneumatic and AC motors. This type of system is usually controlled by signals for target position with constant speed. The servo-controlled type system is for speed and direction. This can be an expensive system since it has a complex structure with servo motors, worm gear reducers, and encoders or resolvers. However, the servo-controlled type system has higher accuracy and variable speed control in both directions of rotation. Errors are compensated for by feedback control.

Various types of manipulators, depending on the types of motion and the degrees of freedom that are required, are shown in Figs. 2 and 3. Several examples of rather simple one-degree-of-freedom manipulators are shown in Fig. 2.

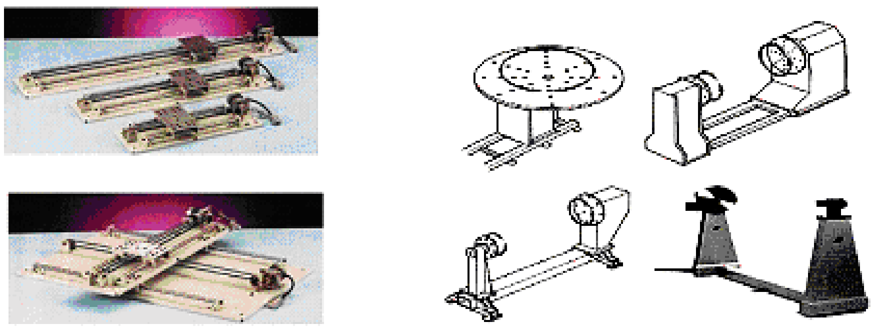


Fig. 2. Examples of One-Degree-of-Freedom Manipulators.

Figure 3 shows more sophisticated manipulators for higher maneuverability, but with an associated higher cost. In selecting the best type of manipulator, it is important to consider control types, load carrying capacity, and working environment. Also, repeatability, accuracy, mechanical configuration, and degrees of freedom are important issues that should be considered.

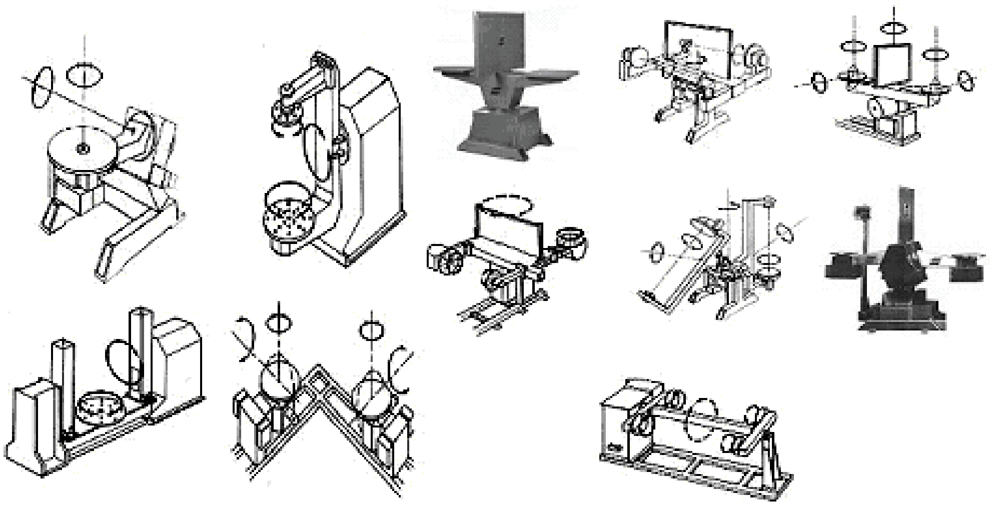


Fig. 3. Examples of More Sophisticated Manipulators.

A decision on the type of control depends on the condition of the weld that is required. In terms of the load carrying capacity, not only the mass or weight of the work piece, but also the moment caused by the off-center distance of the mass center of the work piece must be considered. For example, a work piece with a mass of 227 kg and an off-center distance of 50 mm is equivalent to a work piece with a mass of 1135 kg and an off-center distance of 10 mm. Typically, the load carrying capacity is evaluated at a distance between 76–152 mm. For purpose of illustration, Fig. 4 shows a plot of the load carrying capacity against the off-center distance of the mass center of a work piece.

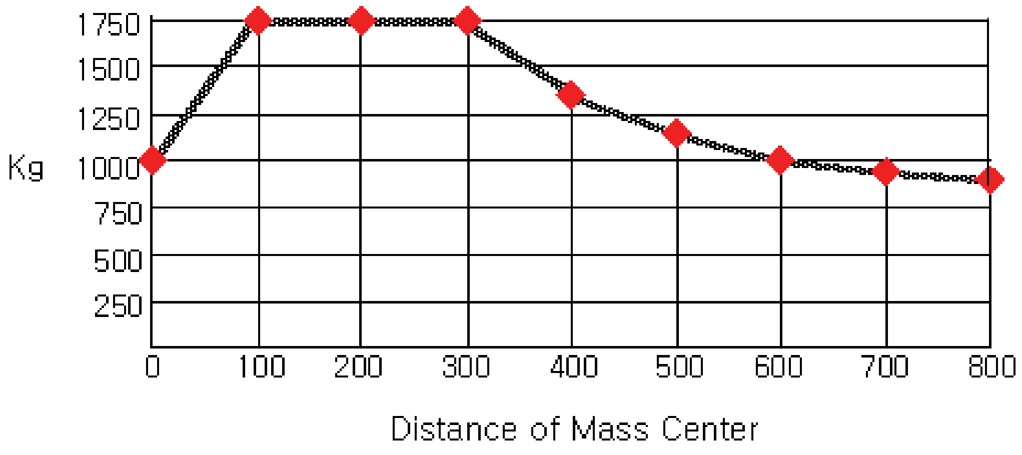


Fig. 4. A Plot of the Load Carrying Capacity.

2.4 Sensors

Sensors collect information from the robot, peripheral devices, and process and transfer this information to a controller. In arc welding, it is critical to consider deformation from high heat input and, therefore, a closed loop control with a sensor is necessary. Also, in an automatic welding system the errors caused by manufacturing tolerances of manipulator and work pieces have to be considered. Various types of sensors for robot arc welding Stations are available, see Table 1, and the right type must be chosen depending on the application.

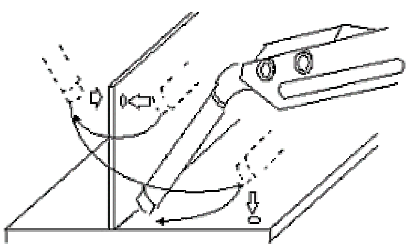
Sensor type	Sensors
Contact type (Weld seam tracking)	Mechanical Type - Roller Spring. Electromechanical type: 1) Two probes across the seam line. 2) A probe in the seam line. Electric control type with probe.
Non-contact type (Various Purposes)	A. Physical type: 1) Acoustic – arc length control. 2) Capacitance – distance control. 3) Eddy current –seam tracking. 4) Induction – seam tracking. 5) Infrared radiation – penetration control. 6) Ultrasonic – penetration and weld quality. 7) Magnetic – detecting electromagnetic field. B. Through-the-arc type: 1) Arc length control (arc voltage). 2) Weaving with electric measurement (GTAW, GMAW). C. Optical/vision (image capture and process): 1) Vision sensors. 2) Laser sensors. 3) Opto-electric sensors.

Table 1. Various Types of Sensors for Robot Arc Welding Stations.

As illustrated in Table 1, and shown in Fig. 5, there are two types of sensors; namely, a contact type sensor and a non-contact type sensor.

A contact type sensor. Figure 5(a) shows a contact type sensor. A gas nozzle, or a finger, is used as a probe to detect contact with the work piece. The nozzle senses the existence, location, and orientation, and, thereby, the location of the weld seam. A contact type sensor is less expensive and easier to use than a non-contact type sensor. However, this type of sensor can not be used for butt joints and thin lap joints.

A non-contact type sensor. Figure 5(b) shows a non-contact type sensor referred to as a through-the-arc sensor. This sensor detects changes of welding parameters while the torch is weaving during the arc welding process. The sensor may be used for tee joints, U and V grooves, or lap joints over a certain thickness. This type of sensor is appropriate for welding of bigger pieces with weaving when penetration control is not necessary. However, it is not applicable to materials such as aluminum.


(a) A Contact Type Sensor.

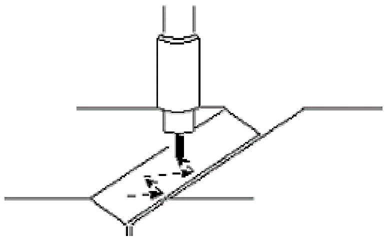

(b) A Non-Contact Type Sensor.

Fig. 5. Two Types of Sensors.

Optical systems for guiding a weld seam. Figure 6 shows a system that will detect an upcoming weld joint. This sensor may be used for grooves such as V and J, lap joints, fillet joints, butt joints, or corner joints. However, since it is located near the torch there is a limitation in access.

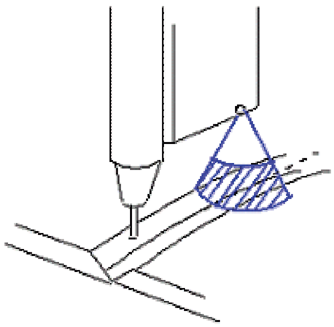


Fig. 6. Optical System for Guiding a Weld Seam.

The system in Fig. 7 is used for detecting minor changes of the joints that lie ahead. This system can be used for thin welds and at high speeds. The system is good for lap, fillet, or butt joints. However, it is important to note that there is the possibility of a collision with the work piece.

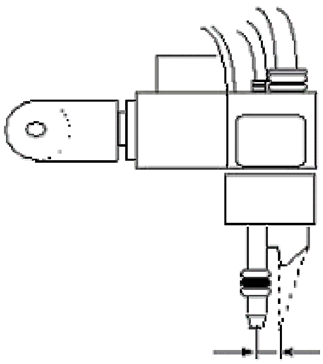


Fig. 7. A Sensor for Detecting Minor Changes of Upcoming Joints.

2.5 Track, Gantry, Column, and Peripheral Equipment

When a work piece is too large for the robot workspace, or a robot cannot reach some welding points, peripheral devices such as a track, a gantry, or a column should be considered. These devices have advantages of expanded work space, flexibility, and increased productivity. Also, it is possible that a robot may work on a multiple of work pieces and, thereby, increase the arc time. For efficient use of these devices, it is advantageous to provide all the axes of the system (including the robot and peripheral devices) with simultaneous or synchronized control function. The standardized and modularized system may be chosen based on load carrying capacity, stability, accuracy and repeatability, and the maximum number of axes that the controller can handle. However, the productivity proven by practice is the most important criteria.

A Track. To increase the working range, the robot is mounted on a track as shown in Fig. 8. The track also this provides flexibility for the future consideration of the size of the work piece. In addition, a track is useful when the breadth of the pieces is less than 1 meter and a travel distance greater than 1.5 meter is required. Most common work pieces for track application are automotive panels, rear axles, tractor frames, bed and furniture frames, window frames, container doors, and computer racks. For increased arc time and productivity, the concept of one robot with multiple work stations is used. This is also illustrated in Fig. 8.

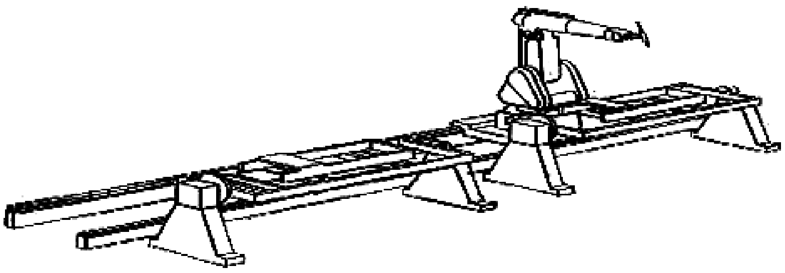


Fig. 8. A Robot Mounted on a Track.

A Gantry. A gantry is a steel structure where the robot is suspended and inverted. Using a gantry, a robot can weld work pieces of different sizes. A very large work piece can be welded with multiple robots suspended on a single gantry or multiple gantries. Figure 9 shows two robots suspended from a gantry (i.e., a grinding robot and a welding robot) working on a double station manipulator.

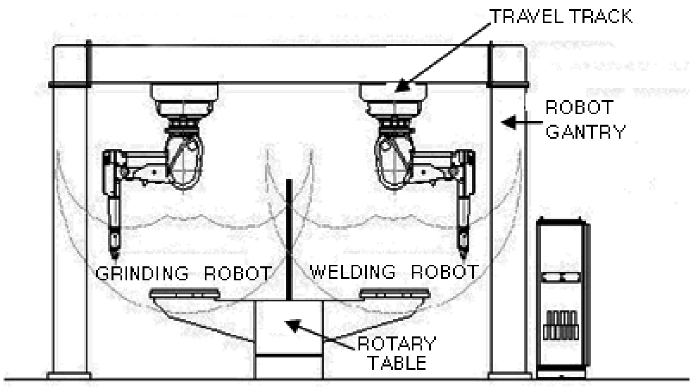


Fig. 9. Two Robots Suspended from a Gantry.

A Column. A column is developed from the concept of modularization. While the track moves in a single horizontal direction, the column may fix the robot or move the robot in a vertical direction and a horizontal direction. A column occupies less space in the plant and makes possible efficient utilization of the space. Also, the wirings of welding wire, power and signal cables may be placed in a duct inside the column to avoid unnecessary exposure. As shown in Fig. 10, a robot suspended from a column may have better accessibility and weld position than a floor mounted robot. There are stationary (fixed), traveling, rotary, and rotary/traveling types of columns.

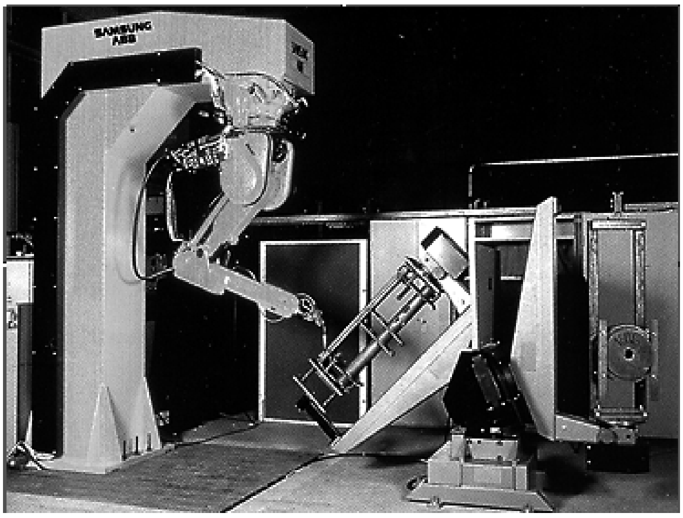


Fig. 10. A Robot Suspended from a Column.

Welding Fixtures. Manufacturing a welding fixture requires experience and know-how. A designer should have a profound knowledge of tolerances of work pieces before and after welding. Also, a designer should obtain information from experienced welding experts. The geometry of a fixture is based upon the geometry of the work piece and the clamping device of the manipulator. The fixture should guarantee a good welding position and should be protected against heat, smoke, and spatter. Figure 11 shows an example of a welding fixture.

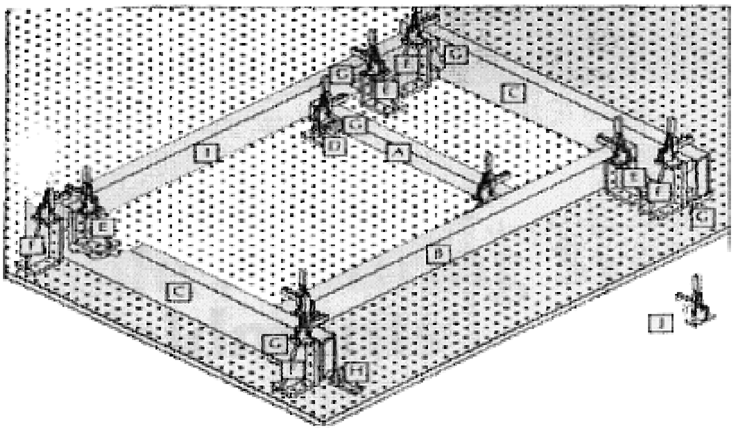


Fig. 11. An Example of a Welding Fixture.

2.6 Safety

An arc welding robot system should be on a firm foundation so that any vibrations will not produce a shaking effect on the system. Also, the emergency switch button (with colors of yellow and red) should be located at a position that is easily accessible. The switch should stop the robot without shutting off the power. There should also be a safety fence to protect the work force from spatter and arc flash. Figure 12 shows a complete system of a safety fence.

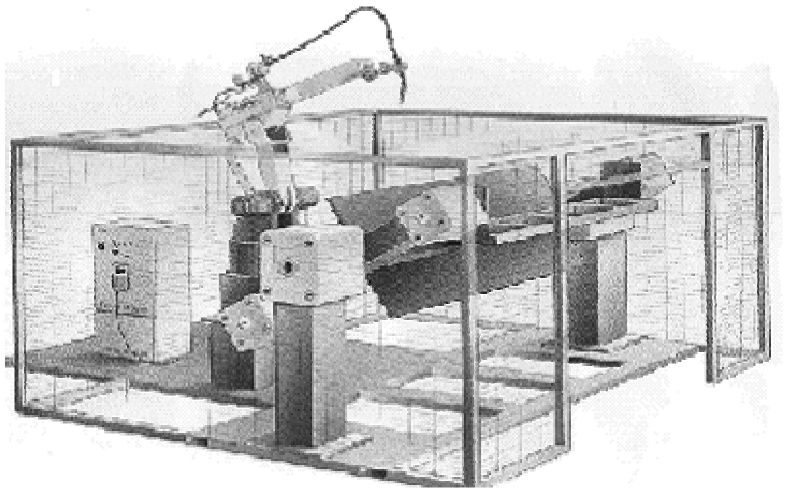


Fig. 12. A Complete System of a Safety Fence.

For safety, the robot operator should have a rigorous training on robot speed, working range, emergency stopping, and functions of teach pendant. The training should also provide the operator with the opportunity to become familiar with the robot system.

While designing the robot system, sufficient time for system modification and operator training is indispensable. Also, obstacles inside the working area of the operator should be eliminated and the system should be designed so that the welding is perfomed at a safe distance from the operator. For this purpose, a manipulator should be designed with dual stations with safeguards so that the operator can work safely on loading and unloading. This will also increase productivity.

After the system is designed, installed, and tested, all the detailed information in the design process should be documented and transferred to the operator.

3. Important Functions of an Arc Welding Robot

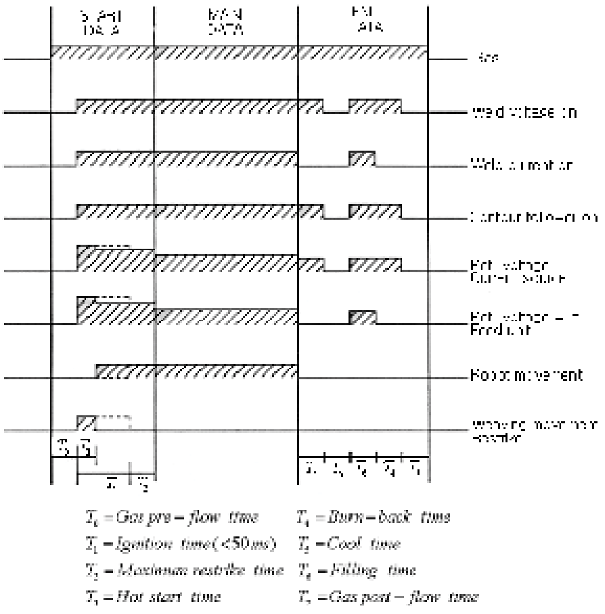
3.1 The Robot Program

(A) Programming Method. First generation robots were programmed by manual operation. However, in modern technology, there are four common methods for robot programming; namely:

(i) Programming by a Teach Pendant. The operator uses a dedicated teach pendant for robot teaching and program editing. Teaching is carried out for the tool center point (TCP) and the LCD display panel is adopted for menu guide. It is easy to use but restricted in application and extension. A teach pendant (such as the one shown in Fig. 13) is the most popular device in robot programming

Fig. 13. A Robot Teach Pendant.

(ii) Programming by Manual Lead-Through. A well trained welding expert will hold the holder near the torch and program by manual lead-through. This was common in first generation robots, however, in modern technology, this is only used for spray painting robots.
(iii) Programming by a Robot Language. The robot is programmed by a program language using a monitor and a keyboard. There exist several command, motion, and operation level languages. Commonly used robots can take advantage of a broad range of motion level languages. The operation level language only describes the final goal of the process, and the sequence of motion and data are generated automatically. This programming method still remains very much in the research stage.
(iv) Programming by a Simulator. A graphic simulation is performed and it is translated into the language of the robot. This is also referred to as off-line programming.
(B) Welding Data. Welding data is special data of parameters that are used for the welding process. The welding data is composed of start data, main data, end data, and weaving data. Figure 14 shows an example of welding data.


Robot movement
T_0 = Gas pre − flow time
T_1 = Ignition time(<50ms)
T_2 = Maximum restrike time
T_3 = Hot start time
T_4 = Burn − back time
T_5 = Cool time
T_6 = Filling time
T_7 = Gas post − flow time
Fig. 14. An Example of Welding Data.

Start data. Start data generates arc start data and stabilizes electric power. Start data contains the following parameters:

- Ignition voltage and current.
- Gas preflow time. The time between shield gas flow and arc start.
- Hot start voltage and current: To stabilize the arc in the first stage.
- Restrike amplitude: Minor change in the position of the torch to start the are.
- Restrike cross time: Time to stay in the restrike position.
- Arc welding start maximum time. The robot stops the process if the arc does not start in this time interval.

Main data. Main data contains the following parameters for the actual welding process:

- Welding voltage.
- Welding current.
- Welding speed.

For higher productivity, the welding speed should be increased to the maximum value. Therefore, a new system should be put through a number of tryouts until the parameters for maximum speed are determined. The above three parameters have interrelations with each other.

End data. At the end of the welding process there are craters, or cracks, that may be the cause of welding defects. Therefore, several parameters for appropriate finish are required. These parameters include:

- End voltage and current.
- Gas postflow time.
- Burnback time.
- Cool time.
- Fill time.

Weaving data. A large work piece, to be welded with large penetration repetitive welding, demands a long time and requies a motion plan to the weld start position. Carrying out this type of welding in one pass is possible due to weaving. Weaving has various patterns such as zig-zag, V-shape, triangular, and wrist weaving as shown in Fig. 15. Wrist weaving uses only the sixth-axis of the robot and enables weaving in a narrow space where weaving with the lower arms of the robot is impossible. Also, it is useful when high frequency weaving is necessary. However, there is an error in the weaving height, as shown in Fig. 15. Therefore, small amplitude weaving is recommended.

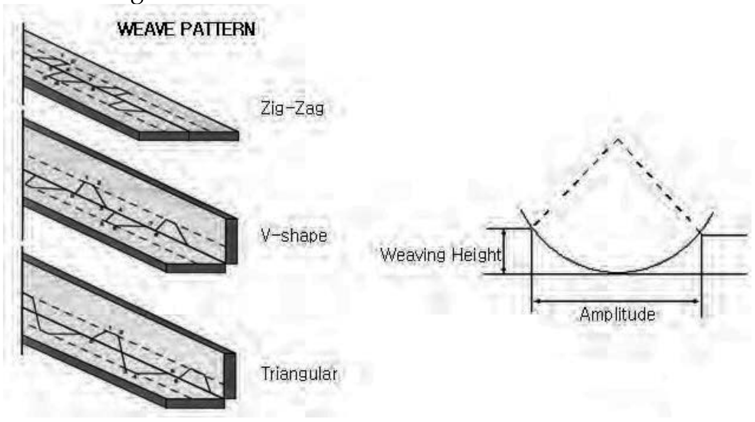


Fig. 15. Zig-Zag, V-Shape, Triangular, and Wrist Weaving.

3.2 Coordinates and Interpolation

(A) Coordinates. Typically, there are five types of coordinate systems in robotics; namely:

- A World coordinate system.
- A Base coordinate system.
- A Hand coordinate system (a mechanical interface coordinate system).
- A Tool coordinate system.
- A Goal coordinate system.

These five types of coordinate systems for a typical robot are illustrated in Fig. 16.

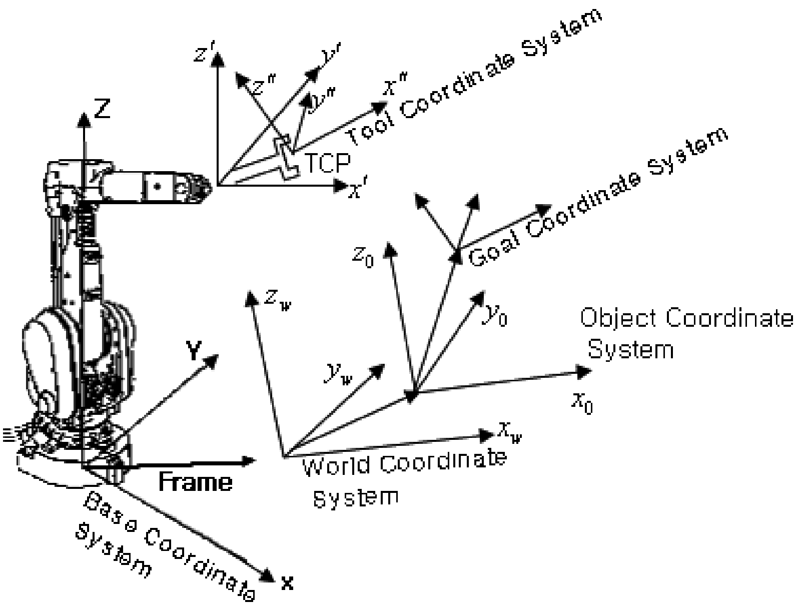


Fig. 16. The Five Types of Coordinate Systems.

The world coordinate system is the reference coordinate system for the other four types of coordinates. The world coordinate system relates the coordinate systems in a work cell where robots and other peripheral devices exist. The base coordinate system is the coordinate system for the robot. The hand coordinate system is at the face of the robot end-effector where tools are attached. The tool coordinate system is at the tip of the torch and defines the Tool Center Point (TCP) and the Tool Center Line (TCL). When several tools are used, it is convenient to define a TCP for each tool.

There are many advantages associated with defining the different types of coordinate systems, for example:

- Easy manual operation of a robot.
- Easy management of the tool data.
- Easy position data manipulation for work pieces of similar geometry.
- Easy application of offline programming.

(B) Interpolation Control Function. Robot path programming is based on interpolation technique. Basically, the interpolation is carried out either in joint space or Cartesian space. For more details, readers may refer to robotics text book. We introduce four techniques directly related to industrial application.

Point to point interpolation. This is the simplest and fastest interpolation technique. Point to point interpolation commands the TCP to move from one point to another regardless of the intermediate path. This technique is good when there are no obstacles in between. When there is an obstacle it may be avoided by defining intermediate via points.
Linear segments with parabolic blends. When a constant speed trajectory in the world coordinate system is needed, this interpolation technique is used. At the beginning, the speed linearly increases and near the end, the speed linearly decreases.
Minimum time trajectory This technique seeks the minimal time trajectory between two positions. The trajectory is obtained for maximal acceleration and deceleration and is called bang-bang trajectory.
Interpolation for weaving. When weaving is used for thick work piece welding, paths for weaving are calculated based on basic welding path.

3.3 Cooperation with a Peripheral Device

(A) External Axis Control. External axis control is used for peripheral devices such as manipulators, columns, tracks, and gantries which need analog signals for a servo motor or actuator control.
(B) Functions for Off-Line Programming. The final goal of an automated system is unmanned operation. Therefore, a higher level computer, or a controller, should be able to control the system. The higher level computer shown in Fig. 17 not only controls a system at the lower level but also stores data with larger memory storage playing the role of backup memory.

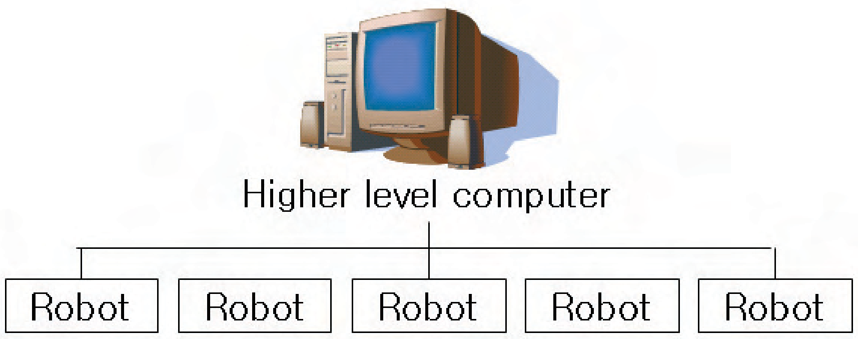


Fig. 17. A Higher Level Computer and the Robots.

4. Special Welding Automation

(A) TIG Welding Automation. TIG welding (GTAW) uses a tungsten electrode to generate arc with a base metal and a filler metal is supplied separately. TIG welding is used for various metals such as aluminum, magnesium, and stainless steel providing high weld quality. Also, TIG welding does not need post-treatment and does not have spatter. When a robot is used, the filler metal should be supplied ahead of the torch. Also, the process is sensitive to noise and, therefore, care has to be taken for various noises in the area. Figure 18 shows a TIG welding system welding thin stainless steel plates with nickel alloy for a jet engine part. There is a dual manipulator, a power source for 150A, and a robot with an arc welding software.

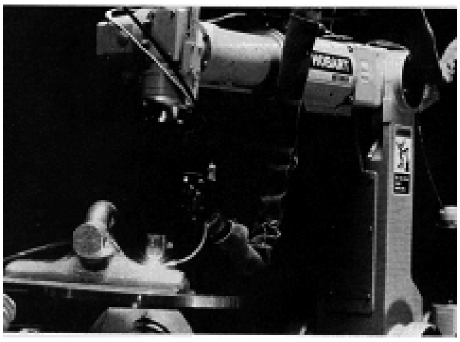


Fig. 18. A TIG Welding System of Stainless Steel Plates with Nickel Alloy.

(B) Aluminum Welding Automation. Aluminum is a metal that is very difficult to weld. DC reverse polarity needs to be used in MIG welding and AC needs to be used in TIG welding. Aluminum has the following unique properties:

- Aluminum oxide surface coating.
- High thermal conductivity.
- High thermal expansion coefficient.
- Low melting temperature.
- No color changes approaching melting point.

In order to weld aluminum, the following is a list of some critical considerations:

- Base metal and filler metal should be kept clean and dry and oxide surface should be removed.
- Appropriate filler metal should be chosen.
- Welding parameter should be accurate.
- High welding speed should be maintained.

For an arc welding robot system, a good coordination of peripheral devices with simultaneous control of external axes are very important issues.

5. References

[1] J. D. Lane, "Robotic Welding," IFS, 1987.

[2] H. B. Cary, "Modern Welding Technology," Third Edition, Regents/Prentice Hall Book Company, Inc., Upper Saddle River, New Jersey, 1994.

[3] N. S. Seo, "Modern Manufacturing Processes," Dongmyungsa, Korea, 2002.

[4] P. F. Ostwald, and J. Munoz, "Manufacturing Processes and Systems," John Wiley and Sons, Inc., New York, 1997.

[5] S. Kalpakjian, "Manufacturing Engineering and Technology," Addison Wesley Publishing Company, Inc., Reading, MA, 1992.

[6] J. J. Craig, "Introduction to Robotics, Mechanics and Control," Addison Wesley Publishing Company, Inc., Reading, MA, 1989.

[7] K. S. Fu, "Robotics," McGraw-Hill International Book Company, Inc., New York, 1987.

Robot Automation Systems for Deburring

Beom-Sahng Ryuh[1], Gordon R. Pennock[2]
[1]*Division of Mechanical Engineering, Chonbuk National University, Jeonju, Republic of Korea, 561-756*
[2]*School of Mechanical Engineering, Purdue University, West Lafayette, Indiana, 47907-1288, USA*

1. Introduction

Deburring is a significant area of research for robot application. However, it is difficult to obtain details on this important topic in modern text books and archival journal papers. Therefore, in general, the engineer is subjected to a time consuming decision making process that usually involves trial-and-error procedures. This chapter will attempt to organize important know-how and field experience, in a logical sequence, so that the engineer can use the information in a straightforward manner. The chapter will provide important details on various types of tools, material selections, robots and station concepts, and programming. This information will help reduce the trial-and-error approach, which is a significant drain on time and money. The contents of this chapter will also provide the opportunity to obtain an optimal solution to the problems associated with deburring.
The chapter is arranged as follows. Section 2 defines a burr and presents background information on the automation systems for the deburring process. The process is classified by the types of burr and the work material. Sections 3 gives a brief introduction to robots for the deburring process. Section 4 discusses the tools of a robot deburring system including, the spindle motor, tool holders, and tool selection. Section 5 focuses on the rotation-type burr tool; i.e., deburring by a solid rotating file. The rotating file is the most commonly used at the present time. Section 6 focuses on the deburring operation and includes a discussion of potential problems in the deburring process. Section 7 presents a discussion of the selection of tools for different applications. Finally, Section 8 addresses the problems associated with the cutting and deburring of polymers and die castings.
The material in this chapter is based on the reports and experiences of experts in Europe, the United States of America, Japan, and Korea.

2. Burrs and Deburring

A burr is defined as an undesirable effect generated at the edge of a work piece after the machining process. In a broader sense, burrs can also be generated during casting, forging, sintering, welding, cutting, plating, and painting. Usually, a burr is harder than the parent metal due to work hardening. Sometimes a burr is considered to be different from a flash but both are common in affecting the processes that follow. For example, a burr can be a

major hindrance in assembly work and the sharp edge of a burr can cut a workers hand. Also, in general, burrs give the appearance of a poorly finished product.
Deburring can occupy as much as five to ten percent of total man hours, yet in many cases, it is generally regarded as a minor problem or just a bothersome phenomenon. As the demand for quality becomes more rigorous, and the role of process automation becomes more important, it is necessary to devise a substantial and systematic prescription against burrs and flashes. The best solution is that a burr should not be generated in the first place, or the effects of a burr should be minimizal. Also, the direction of a burr should be controlled in the process planning stage so that it does not affect the upcoming processes. Finally, the burrs that are generated must be removed, in which case, a deburring methodology, automation, process planning, and justification of the financial implications must be thoroughly investigated.
Robot automation systems for the deburring process have been developed primarily for the automotive industry, kitchen products, and the manufacture of plastics. The motivation is due to the reduction in the available work force, industrial environmental problems such as dust, and the demand for cycle time reduction. The latter issue is very important in applications such as the production of crankshafts. The deburring process, however, is a relatively undeveloped process and it is difficult to obtain a comprehensive list of the literature on the subject. In general, information on deburring automation is not readily available, it has remained as the know-how of the experts in the field and in the reports of system houses. The robot automation system of deburring is composed of a robot (see Section 3) and peripheral equipment. The most important peripheral equipment is the fixture device of the work piece. This device may be a simple clamping table, an indexing table, or a sophisticated manipulator depending on the geometry of the work piece and conditions such as cycle time. When the size of the work piece is bigger than the robot work envelope then robot track motion should be used.
The selection of the correct deburring tool is very critical for the success of the process and, therefore, is an essential part of the deburring process. Also, more than thirty-five methods have been reported for the deburring process. The most important criterion in choosing one method over another is the minimization of total manufacturing cost. Primary items that must be considered are the production volume, the cycle time, the material properties, tolerances, geometry, and dimensions. Secondary items are the investigation of the process that generates a burr and the purpose of burr removal. The best solution can only be obtained when all of these conditions are thoroughly investigated.

3. Robots for the Deburring Process

In the robot automation of machining processes, robots are performing operations such as deburring, polishing, grinding, and light duty cutting. The robots for these processes require a higher stiffness and rigidity, especially in the wrist area, compared to robots for applications such as arc welding. Industrial robots, however, have a limited rigidity especially at the wrist and, therefore, are not suited to process applications. Different from applications such as arc welding, the robots are in direct contact with the work piece and have to sustain reaction forces and torques, and withstand undesirable vibrations. Usually, industrial robots designed at low cost prices are for less challenging applications using timing belts, harmonic drives, and cycloids. Sturdier robots are typically designed with speed reducers, made of gear assemblies that can be used for machining processes. In some cases, dedicated robots are developed for this specific purpose. These robots are generally designed to withstand the inevitable dust and chips.

The major application areas of industrial robots include:

- Deburring of automotive engine blocks, crankshafts, and camshafts.
- Deburring and finishing of die casting products.
- Polishing and buffing of kitchen products.
- Cutting and drilling of plastics.

4. The Tools of a Robot Deburring System

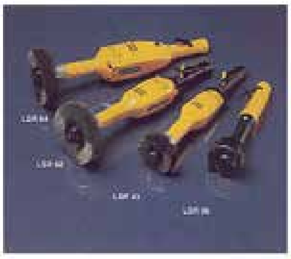
A robot deburring system is composed of a spindle motor, a deburring tool, and a tool holder. Peripheral devices for the robot and the work piece such as a manipulator, and a robot track, are based on the same principle as for arc welding robot automation (see the chapter on arc welding robot automation systems).

(A) The Spindle Motor for Deburring

A spindle motor rotates the deburring tool, oscillates the tool in the axial direction, and occasionally rotates the work piece when it is small and lightweight. The motors are classified by speed as follows: (i) a high speed motor (15,000 rpm and higher) for rotating files; (ii) middle speed motor (10,000 - 15,000 rpm) for operation by brush with 60 mm diameter or less; and (iii) low speed motor (500 – 1,500 rpm) for tools for special applications such as deburring the inside of a hole.

A pneumatic spindle is typically used on a low speed motor, and an electric motor and a high frequency motor are used for the higher speeds. The pneumatic spindle is the most commonly used. The maximum speed is 30,000 rpm with no load and the net power can be as high as 550 Watts. This spindle is relatively cheap, reliable, and a high power versus weight ratio. A high frequency motor is controlled by a converter to vary the speed between 5,000 rpm and 12,000 rpm with a maximum horsepower of 0.74 kW and a maximum torque of 0.15 Nm. For reciprocating oscillating files, special reciprocating motors are used, which operate at 2,000 ~ 4,000 strokes per minute with an amplitude of 1 ~ 9 mm.

For a brushing application, two different concepts can be used. The first concept is for the robot to hold the work piece and approach the brush wheel mounted on the floor. The second concept is for the robot to hold the spindle and brush unit and approach the work piece which is fixed on a clamp as shown in Fig. 1(a). In this case, the brush wheel usually has a diameter less than 60 mm and the speed of the motor does not exceed 15,000 rpm. When burrs are easily accessible, such as burrs on the outside edges or on the corners of a work piece, then small belt grinders attached to the robot, see Fig. 1(b) for example, will give good results.


(a) A Brush Grinder Tool.

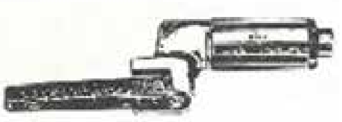

(b) A Belt Grinder Tool.

Fig. 1. Two Types of Grinder Tools.

For deburring the inside of a hole, there are two different cases. First, when the hole diameter is larger than 25 mm, then general rotating tools with high speeds (15,000 rpm and greater) may be used. Second, when the hole diameter is smaller than 25 mm then special tools, or counter sinking tools, with low speeds (30 - 60 m/min) are used. Figure 2 illustrates the procedure for deburring the inside of a hole.

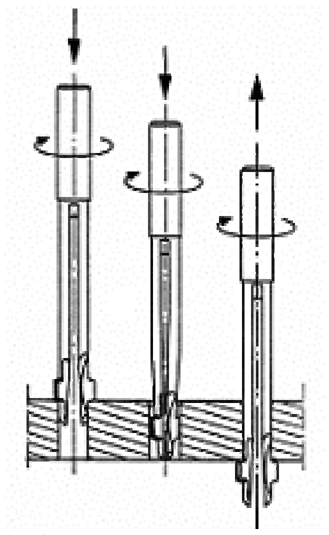


Fig. 2. Deburring the Inside of a Hole.

When selecting a spindle, the pneumatic spindle should be given first consideration since it is the most economic and reliable tool. When various speeds are required for several tools then a high frequency motor and a tool adapter with an automatic tool changer should be used. The high frequency motor is controlled by a converter and changes speed easily across broad range with quick response. However, the high frequency motors are expensive and if only two speeds not variable speed are required then the concept based on high frequency motors should be compared with using two cheaper pneumatic spindles.

(B) A Deburring Tool

The most common tools for robot deburring automation are flexible tools, solid burrs (rotating files), oscillating files, and internal deburring tools. There are belts and brushes in the category of flexible tools. The belts are made of coated abrasives with backing paper or cloth, or abrasive non-woven webs. The brushes are made of aluminum oxide or silicon carbide impregnated on nylon, or steel. The most effective work is at the tip of the brush. Generally, a belt is efficient for straight line edges and a brush is efficient for corners.

The most widely used tools in robot deburring automation are solid deburring tools which will be discussed in some detail in Section 2.3. Solid deburring tools are rotary files made from various materials that have many different shapes. The selection of the correct tools is based on geometry, material property, location, and the volume of the burrs to be removed. Oscillating files are used for small and slender burrs. There are files with various profiles as shown in Figure 3. The appropriate profiles are chosen based on speed. The stroke is based on the burr geometry and conditions such as oscillating tool performance.

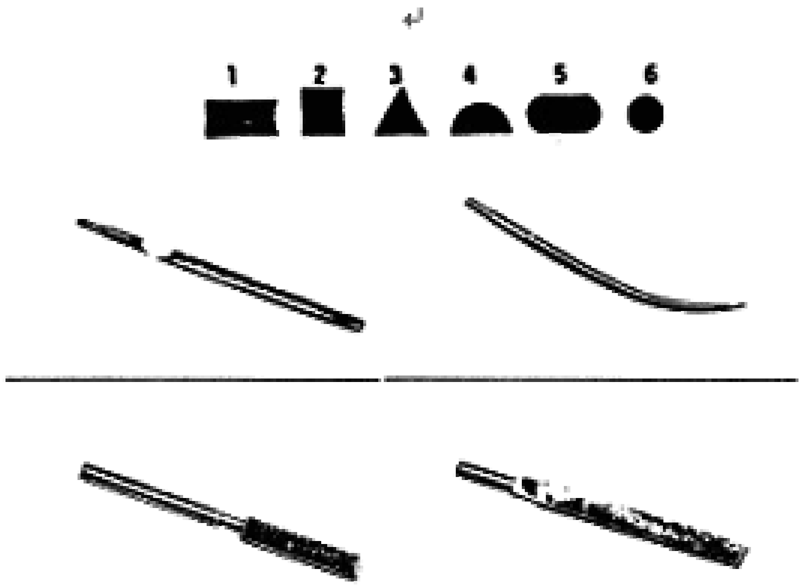


Fig. 3. An Oscillating File and a File Profile.

A tool for deburring the inside of a hole is shown in Fig. 2. The tool should have radial compliance so that the diameter can be extended inside the hole. A description of the spindle was presented in section (A).

(C) Tool Holders

Solid burrs do not absorb the reaction during the burr removing operation. Also, the volume and the hardness of the burrs are not uniform. Therefore, the results may not be uniform and the robot and spindle are exposed to unexpected shock. In such cases, the robot may stop running and provide an error message. To protect the system from this situation, tool holders should be provided with certain compliance functions. A simple solution is to use basic compliance function provided by the tool holder itself while a sophisticated system may use a specially designed integrated spindle and tool holder system.

General overviews of simple solutions to tool holder compliances are illustrated in Fig. 4. The figure shows that there are four types of tool holder compliance:

(i) Multi-directional compliance, see Fig. 4(a). The spindle is wrapped in an elastic material such as rubber. This concept is used when the spindle is under axial direction thrust but does not provide precision due to slow elastic recovery.

(ii) Two-directional compliance, see Fig. 4(b). The spindle holder is loaded with a spring for compliances in two directions perpendicular to the spindle axis.

(iii) One-directional compliance, see Fig. 4 (c). The spindle holder is loaded with a spring for compliance a direction perpendicular to the spindle axis.

(iv) No compliance, see Fig. 4(d). The spindle holder is rigidly fixed without compliance for soft material deburring.

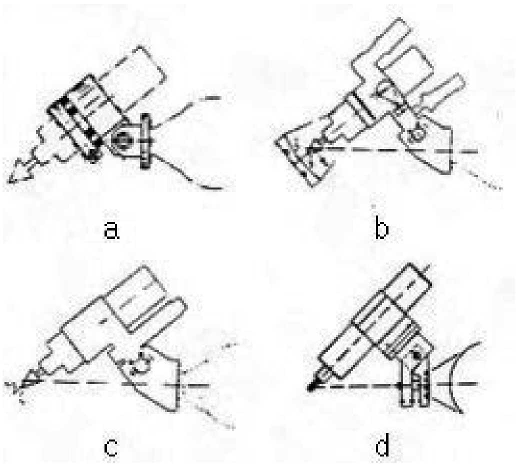


Fig. 4. Four Types of Tool Holder Compliance.

(a) A Rubber Suspended Holder. (b) A Two-Way Spring-Loaded Holder.
(c) A One-Way Spring-Loaded Holder. (d) A Rigid Holder.

(D) Tool Selection for Robot Deburring

The selection of robot deburring tools is based on the following criteria:

(i) The selection of tool profile based on geometric accessibility to burrs.
(ii) A decision of the correct tool compliance based on burr size and location.
(iii) A study of tool cutting data considering cutting force and speed, tool RPM, and material property.
(iv) A selection of spindle type and specification based on the cycle time and the manufacturing cost.

5.The Rotation-Type Burr Tool (A Solid rotating file)

The most common tools for robot deburring are rotating burrs. A rotating burr gives precise and predictable results. Also, the tool is small, has a long life, and various shapes and materials are readily available. Therefore, right selection of tools is very important.

(A) The Geometrical Shape of the Tools

Figure 5 shows the typical structure of a rotating burr. It consists of a tool body, a tooth and fluting pattern, and a shank.

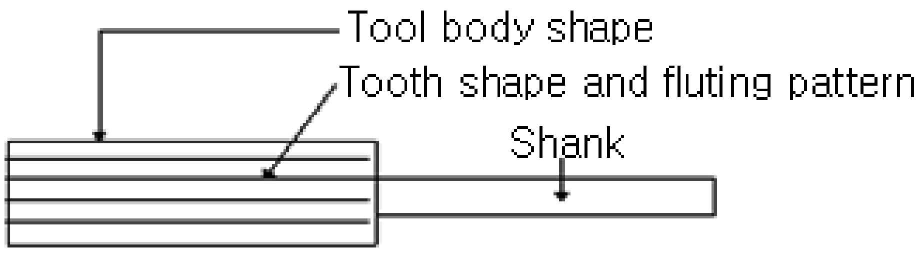


Fig. 5. A Rotating Burr.

The tool body has various shapes, such as wheels of cylinder, cone, and sphere, and saw type cutter depending on the direction and type of compliances. Figures 6, 7, and 8 illustrate various rotating burrs depending on the concept of compliance. When attaching a rotating burr to a robot, it is important to program the robot to apply power to the tool in the direction coincident with the direction of compliance. Figure 6 shows cylindrical shapes for radial direction compliance. The conical cutter with the straight ends offers an advantage in that it requires no dressing.

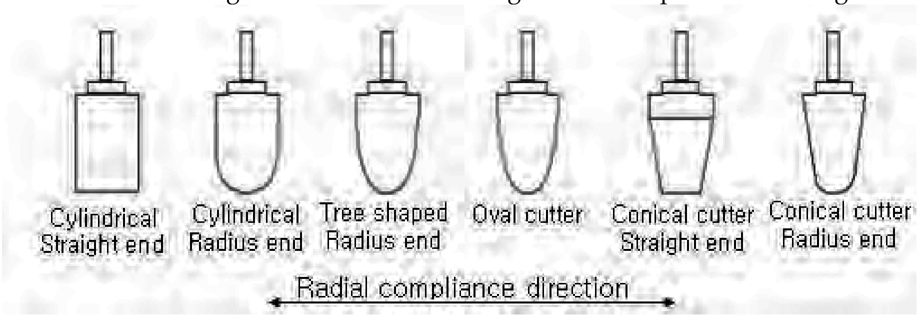

Cylindrical Straight end
Cylindrical Radius end
Tree shaped Radius end
Oval cutter
Conical cutter Straight end
Conical cutter Radius end
Radial compliance direction

Fig. 6. Cylindrical Shapes for Radial Direction Compliance.

Figure 7 shows conical shapes for axial direction compliance. The cone cutter offers an advantage because the robot does not need to be reprogrammed to modify the tool center point, even after it is worn down.

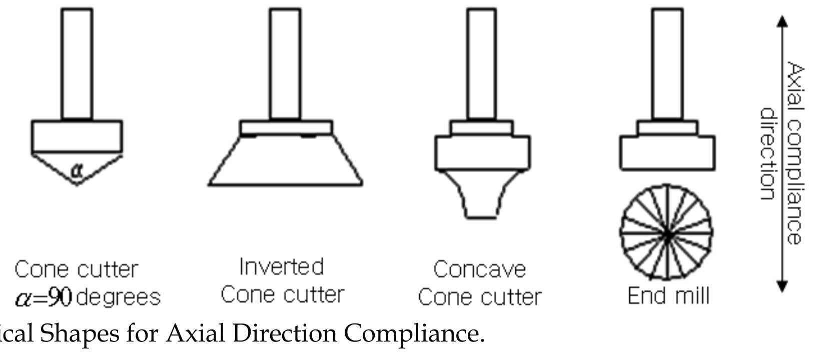


Fig. 7. Conical Shapes for Axial Direction Compliance.

Figure 8 shows the ball shaped cutter for multi-directional compliance and cutters for the case of no compliance.

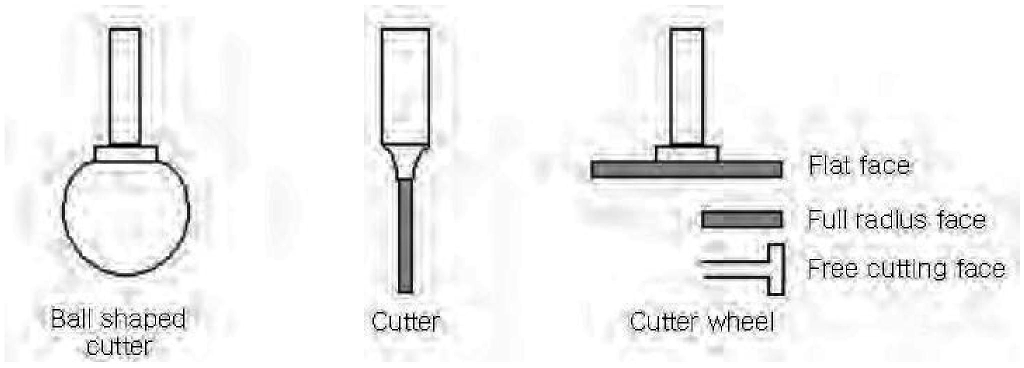


Fig. 8. The Ball Shaped Cutter for Multi-Directional Compliance and Cutters for No Compliance Applications.

The spherical shape of the ball shaped cutter is very efficient for deburring along smooth curves. The cutter wheels and the straight cutters are good for mild materials (such as plastics and aluminum) but need a device to protect them from the vibrations.
The fluting patterns and cutting edges shown in Fig. 5 may be classified as in Fig. 9. The patterns are categorized by the blade angle relative to rotation axis, number and size of blades, and cut mechanisms. It is important to choose right concept of the pattern to the size and hardness, and material of burrs. The standard concept is spiral cut, standard cut, and medium number and size of blades. It should be the starting point when designing a system. In the case where diamond cut, double cut, or chip breaker is used in deburring, then the chip from the process becomes different. The chip generation mechanism is the same as that of machining processes and, for more details, the reader is referred to books on the theory of machining.

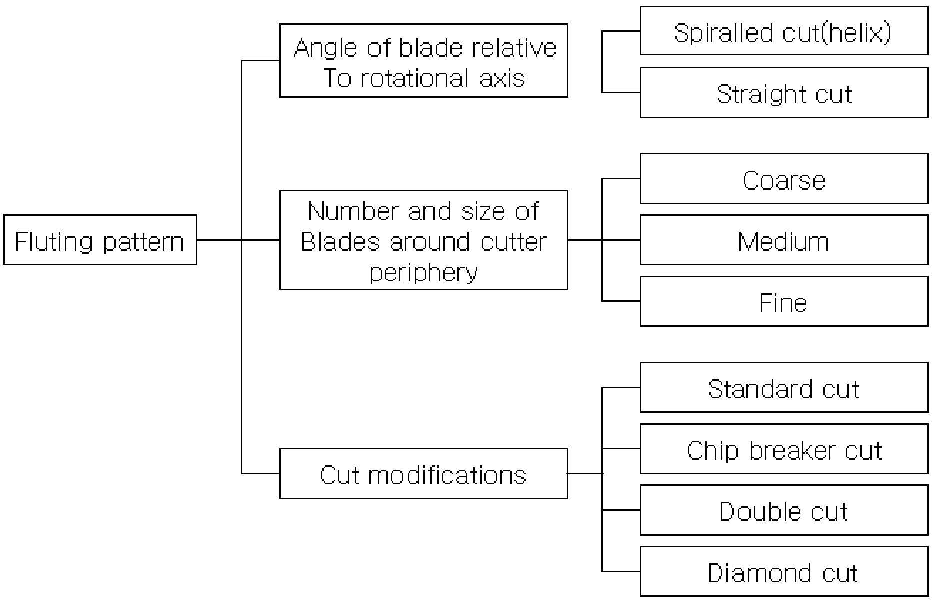


Fig. 9. Hierarchy of Fluting Pattern.

(B) Compliance of Tools

Compliance is the elastic deformation of the tool holder of the spindle due to the external forces that are exerted during the process. Since the active control of forces is not used, force on the tool is considered uniform regardless of robot position. There may be simple solutions to give compliance to tool holders. There are also more sophisticated solutions where spindle, tool, and suspension are integrated as a system.

The directions of compliance for the suspension axis may be classified as radial, axial, two-directional, three-directional, and semi-rigid, as shown in Fig. 10. There is also rigid suspension which are cutters with no compliance, see Fig. 8. An appropriate concept for the compliance may be chosen from the tool holders shown in Fig. 4.

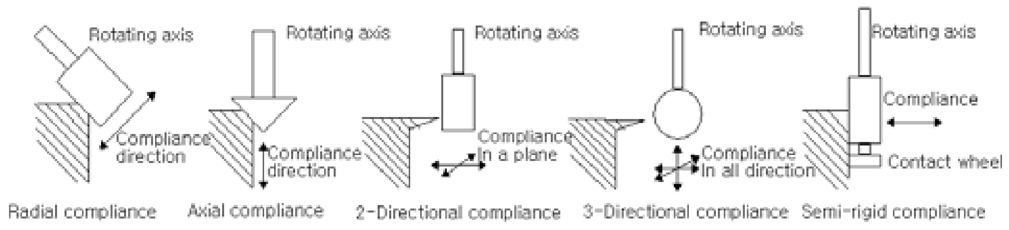


Fig. 10. Directions of Compliance for the Suspension Axis.

The most common compliance in robot deburring is radial compliance. It has a simple structure and it is easy to obtain the correct balance. Radial compliance is commonly useful for robots with six degrees of freedom. The sixth axis of the robot must be aligned such that the axis of compliance is perpendicular to the direction of burrs. A typical example is the deburring of the edge of a hole.

Axial compliance is a new concept which is experiencing an increase in applications. In the conventional concept, the entire motor and tool are moving in the axial direction, which makes the system heavy. In order for the contact force not to be affected by gravity, the system is used for deburring on the same plane. However, new integrated systems such as the tool system proposed by a company, Njuma Hiac [1], have been developed especially for axial compliance with very low weight and high maneuverability.

Two-directional and three-directional compliance are easy to use since they provide the robot with more directions to move, which saves much programming work. However, it is at a cost of potentially more oscillation. Therefore, robot programming speed is usually reduced. Also, if the burr size is uneven along the path the result could be uneven and, therefore, appropriate precautions are necessary such as preloading the tension to a certain value. In the case of semi-rigid compliance, the wheel is always in contact with work. This will give a good result when there are large unexpected burrs. The system is good for castings or machined parts.

Compliance may also be categorized by the elements. Common elements are rubber, pneumatic, spring, and gravitational elements. Rubber is used for two-directional and three-directional compliance. Rubber has a nonlinear characteristic between force and deformation. Also, the rubber must be replaced frequently due to age cracking, and each time the rubber is replaced the robot TCP (tool center point) should be modified. A pneumatic element can control the contact force easily by varying the pressure but attention must be paid to friction in the suspension system. A mechanical spring is a

simple and accurate element, and is good for one directional compliance. The contact force is controlled by a set screw. A preloaded screw can be used for uniform contact force in a limited distance of stroke. A gravitational element is mounted on the floor and the work gripped by the robot approaches the tool so that gravity generates the contact force.
In the performance of the compliance system, the most important factor is the stability of the contact force. It is especially important when the tool position is reversed, and in acceleration or rotation during high speed operation. If the system does not provide sufficient stability then uniform chamfer cannot be obtained during initial contact, at corners, or robot servo lag. The main cause of contact force instability is the low contact force versus inertia ratio. Figure 11 shows the hierarchy of deburring tool systems considering compliance.

(C) Tool Materials

Materials for deburring tools require hardness, toughness, wear resistance, and thermo-chemical stability. Tungsten carbide, high speed steel, and diamond or CBN coated tools are in common use. Table 1 shows typical materials with hardness data. Aluminum oxide and silicon carbide are brittle and used for coated abrasives and bonded abrasives. Tungsten carbide, tantalum carbide, and titanium carbide are very hard materials with good heat conductivity and may be sintered into various shapes of tools. They also perform very satisfactorily near the melting point. Titanium carbide and titanium nitride are also used to make laminated tools by diffusion bonding on base material. High strength steel is used in the temperature range lower than that of carbide tools, which has lower hardness but higher toughness. It is good for deburring of milder materials with lower strength than carbide tools. Diamond and CBN are known as the hardest materials, as shown in Table 1, and may be used as coated abrasive for solid flexible tools or wheel grinder. These materials have longer life and higher wear resistance than any other tool materials. CBN is next to diamond in hardness and better high temperature stability.

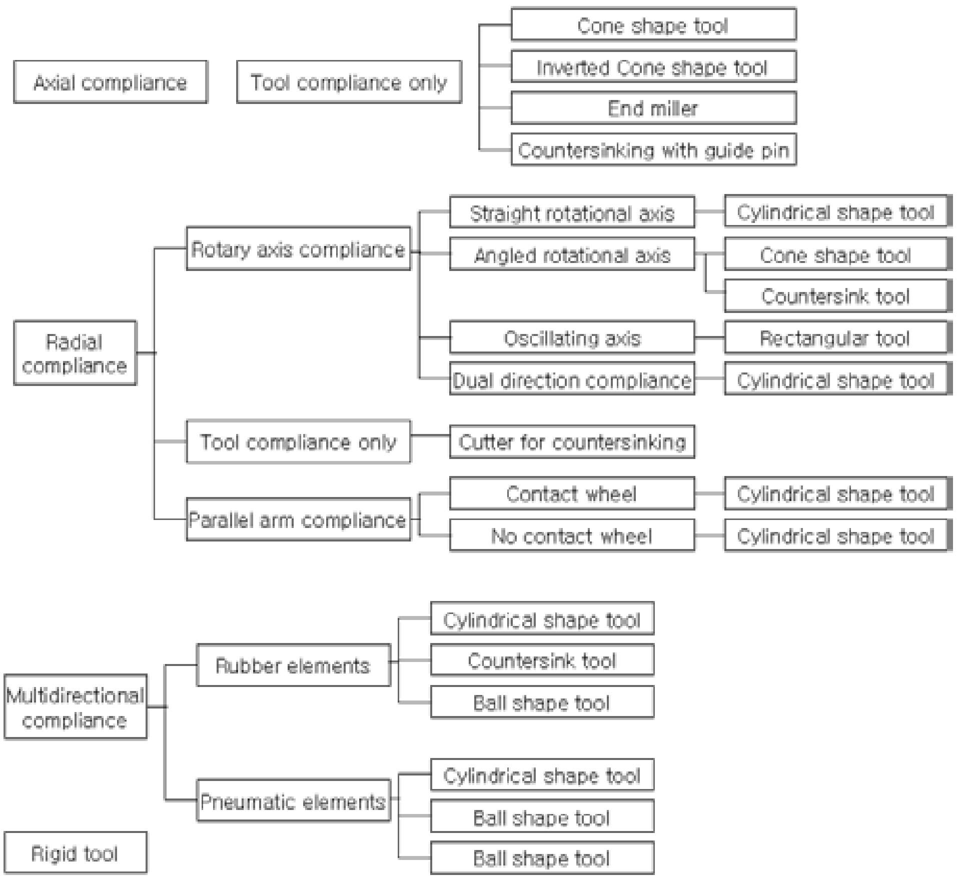


Fig. 11. The Hierarchy of Deburring Tool Systems.

Abrasives Basic elements	Hardness HK, kg/mm^2	Material	Hardness HK, kg/mm^2
Aluminium oxide (Al_2O_3)	2000~3000	Common glass	300~500
Silicon carbide (SiC)	2100~3000	Gray iron	200~300
Tungsten carbide (WC)	1800~2400	70~30 brass	100~150
Tantalum carbide (TaC)	1600~2200	High speed steel (HSS)	800~1100
Titanium carbide (TiC)	1800~3200	Hardened steel	700~1300
Titanium nitride (TiN)	2000		
Cubic boron nitride	4000~5000		
Diamond	7000~8000		

Table. 1. Hardness Properties of Tool Materials.

6. Deburring Operation

After the proper tool has been selected and the station is ready, the robot deburring process can commence. The following is a discussion of the important phases of the robot deburring automation process.

(A) Considerations before operation

The purpose of the deburring operation is to generate round chamfers of a certain radius and breadth. In the case where all the conditions (i.e., width, thickness, and material property) for deburring are uniform, then position control of the tool is the only requirement for a consistent output. However, there are many variables that influence the result; such as:

- The variety of burr size and shape.
- The approaching direction to the burr.
- The repeatability of the robot.

In order to solve the potential problems, the following ideas are proposed:

- Appropriate compliance must be devised.
- Robot speed must be controlled to avoid excessive contact force.
- Adaptive control algorithm of robot path must be applied to variable burr sizes.
- Force control of the robot should be adopted. This, however, is in the research stage.

(B) Potential Problems in the Deburring Process

Important problems in robot deburring process are path inaccuracy, chattering of the tool, proper programming of the robot, and tool wear.

Path Inaccuracy

When the path is not accurate then the chamfer cannot be uniform due to the following reasons.

- The burr size deviation should not exceed 1 to 4 mm. If the deviation is bigger that this then it should be removed in a previous stage.
- The robot is not securely mounted.
- The work piece is not properly clamped.
- Dust or dirt on the surface of the work piece prevents a tight clamping of the work.
- The tool is dislocated from its original position.
- In corner operation, robot servo lag causes inconsistent chamfer. Simpler tool geometry and slower speed may solve the problem.

Chattering of the Tool

Chattering may damage the deburring tool and break the rotating burr. Chattering can be caused by one of the following reasons.

- Multi-directional compliance such as rubber is used when burrs are not uniformly distributed, speed is too high, or contact angle is not appropriate.
- The robot or the tool is not securely mounted.
- There exist backlashes in the tool holder system.
- The contact force is too small compared with the capacity of compliance system.
- There is a chattering in the tool.
- The wrong concept of compliance is adopted.

Proper Programming of the Robot

The number of tools should be minimized for the robot. Also, the following items are important in robot programming:

- Singularity of the robot should be avoided by maintaining the fourth and sixth axis of robot not parallel. In axial compliance, the sixth axis of robot is oriented perpendicular to tool rotation axis while the sixth axis is parallel to the tool rotation axis in radial compliance.
- The tool center is kept not too far from the robot wrist center.
- Relative motion between robot and the work is utilized for quick and efficient operation.
- Heavy deburring tool is not recommended, which restricts speed, accessibility, and accuracy.

Tool Wear

Important factors to consider when investigating tool wear are:

- The robot should be programmed so as to obtain uniform wear of tool.
- When the file is damaged, it is a sign of poor working conditions. The damage should be investigated and work parameters such as RPM, speed, and contact force and angle should be reviewed. The possibility of tool chattering should be investigated.
- The wear mechanism of the tool material should be studied.

7. The Selection of Tools

The most important factor in tool selection is to decide on the tool that is best suited to the geometry of the burr. The second most important factor is the minimization of the number of tools by choosing the most common type of tools. However, if cycle time reduction is more important than optimal tool selection then proper number of tools may be used. In the reports of system houses, solutions to various situations are proposed based on field experiences over many years.

When addressing a problem, a number of potential solutions should be investigated. In some cases, it is possible to find the optimal solution using different concepts. The first criterion is to find the most common tool in the market. Then the right type of compliance may be proposed depending on the tool. This may not be a unique solution but without this guideline, the system engineer may need to go through a long process of trial-and-error and collecting information. The sections on tool selection in this chapter (see Sections 2.2 and 2.3) may be the first set of guidelines presented in the literature, and it is important to understand these guidelines thoroughly.

8. The Cutting and Deburring of Polymers and Die Casting

(A) Polymers

Plastics with high mechanical strength are formed from composite materials such as SMC (sheet mold compound) through molding processes. In this case, the parts are not finished by the process and must be subjected to a final finishing process. The excessive material should be removed by deburring or cutting. In the automation of plastic cutting, the robot must have a greater load carrying capacity (at least 30 kg), and a net power of 0.7 kW. An electric motor with a speed of 54,000 rpm is used as the spindle motor. A pneumatic spindle is too weak for cutting with too high speed. When a spindle is attached to robot, it should be aligned with the sixth axis of the robot in order to minimize the moment exerted on the robot wrist. The spindle holder should be sufficiently strong to withstand the strong reaction forces but care must be taken because the spindle bearing might be damaged if it is too tight. The tool length and the diameter are selected first based on the work geometry and accessibility. The pattern of the tool teeth is selected based on the work material. The fixture is tailor made depending on the work shape and size. The fixture should be sufficiently large and rigid to hold the work securely without tool chattering.

(B) Die Casting

A die cast product will have burrs or flash in various locations, which makes finishing operation difficult.

- In die casting, burrs around gate, overflow, and split line. A trimming press is appropriate since the burrs are uniformly distributed and the press can immediately handle the job.
- In the case of robot operation, the robot holds the work piece and approaches the rotary cutting saw mounted on the floor.
- Burrs along the parting line. When burrs are small and thin, the robot system with rotating burr with pneumatic spindle, oscillating files (scaler), or belt sander will be appropriate. If burrs are big and thick, a chisel may also be used. Figure 12 shows two examples of chisels; namely, a flat chisel and a blank chisel. The robot may hold the work and approach the belt sander or grinder that is mounted on the floor.

Fig. 12. A Flat Chisel and a Blank Chisel.

9. References

[1]"General Deburring Manual," ABB, Sweden, 1990.
[3] N. S. Seo, "Modern Manufacturing Processes," Dongmyungsa, Korea, 2002.
[4] P. F. Ostwald, and J. Munoz, "Manufacturing Processes and Systems," John Wiley and Sons, Inc., New York, 1997.
[5] S. Kalpakjian, "Manufacturing Engineering and Technology," Addison Wesley Publishing Company, Inc., Reading, MA, 1992.
[6] J. J. Craig, "Introduction to Robotics, Mechanics and Control," Addison Wesley Publishing Company, Inc., Reading, MA, 1989.
[7] K. S. Fu, "Robotics," McGraw-Hill International Book Company, Inc., New York, 1987.

Robot-based Gonioreflectometer

Andreas Höpe, Dirk Hünerhoff, Kai-Olaf Hauer
Physikalisch-Technische Bundesanstalt, AG 4.52, Reflectometry
Bundesallee 100, 38116 Braunschweig, Germany

1. Introduction

Measurements of diffuse reflection are important for a variety of applications in optical metrology. In reflectometry both the spectral and the spatial distribution of radiation diffusely reflected by solid materials are measured and characterised with the indication of classification numbers. In practice, reflection measurements of diffuse reflecting materials are accomplished predominantly relative to a standard. As primary standard for this purpose the perfectly reflecting diffuser (PRD) is established, which reflects the incoming radiation loss-free, completely diffuse and with a lambertian direction characteristic. The PRD is a theoretical concept only, which cannot be realised experimentally respectively materially. Since there is no material with these characteristics, the realisation of primary standards is carried out with physical methods, i.e. by the measuring apparatus itself, in the context of an absolute measurement. In contrast to the directed, specular reflection, the incoming light in diffuse reflection is distributed over all directions in space. If the radiance L of the reflected radiation is independent of the direction, one speaks of a lambertian emitter.
The classification numbers, in the case of reflection are the reflectance ρ, the radiance factor β and the BRDF "bidirectional radiance distribution function" f_r. These classification numbers are not material-specific but depend on a multiplicity of parameters, e.g. the wavelength of the radiation, the direction of irradiation and reflection as well as the aperture angle.
There are, in principle, nine different measuring geometries for a reflection measurement. They consists of all possible combinations, where the incident and the reflected radiation is in each case hemispherical, conical or directional. Therefore it is always necessary to specify the measurement geometry for a material classification number. The hemispherical measuring methods can be accomplished thereby with an integrating sphere whereas directional geometries are realised with a gonioreflectometer. The name gonioreflectometer is used in this context for a device which allows the precise control of the angles of the incident and reflected optical beams in a reflection measurement. This publication deals with the first time utilization of a commercial 5-axis industrial robot as a sample holder in a gonioreflectometer.

2. General information about reflectometry

At the Physikalisch-Technische Bundesanstalt (PTB), the National Metrology Institute of Germany, a new robot-based gonioreflectometer (Fig. 1) for measuring radiance factor and BRDF has been set-up (Hünerhoff et al., 2006). The facility enables measurements of the directed reflection characteristics of materials with arbitrary angles of irradiation and detection relative to the surface normal.
Measuring directional diffuse reflection allows one to describe the appearance of a material under user-defined lighting conditions. This includes calibrations for a variety of different customers, like paper, textile and colour industry, companies producing radiometric and photometric instruments, as well as measurements for radiometric on-ground calibration of remote sensing instruments for space-based applications on satellites.

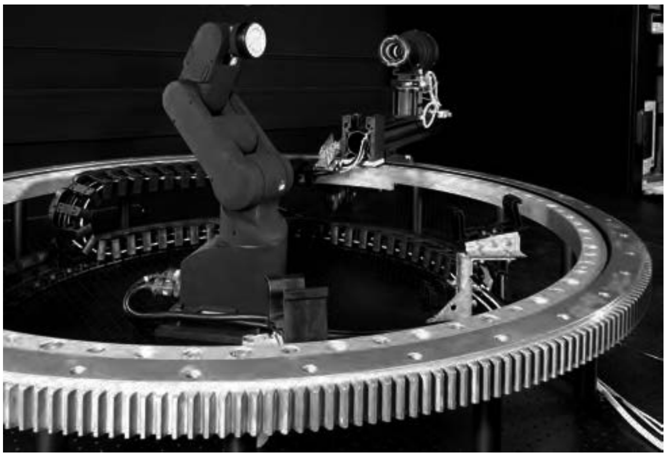


Fig. 1. Photograph of the gonioreflectometer showing the main components large rotation stage and 5-axis-robot.

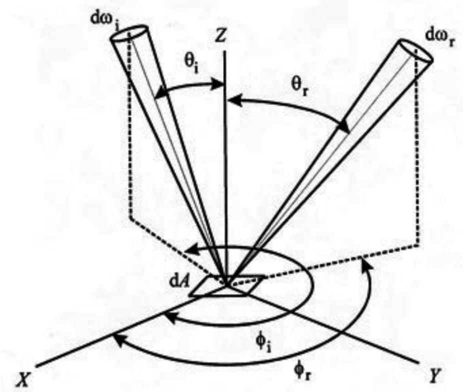


Fig. 2. Geometry of incident and reflected beams at the gonioreflectometer.

The former "old" gonioreflectometer of PTB was completely home-made (Erb, 1980). It was constructed from steel with a air bearing rotation stage of approximately 3 m in diameter. Also the sample holder was self constructed. Due to the layout of the instrument only specific reflection measurements were possible, where the vectors of the incoming radiation, the reflected radiation and the vector of the surface normal formed a plane. The same situation is true for a variety of gonioreflectometer facilities in other National Metrology Institutes (Proctor & Barnes, 1996), (Chunnilall et al., 2003), (Nevas et al., 2004). The measurement of the appearance of surfaces under arbitrary lighting conditions as in every day life, however, requires a gonioreflectometer with multi angle capabilities. Such a gonioreflectometer was set-up in PTB using standard laboratory and engineering equipment.

As mentioned in the introduction one important classification number for the diffuse reflection is the radiance factor. The radiance factor β is defined as the ratio of the radiance L_r of the specimen to that of the perfect reflecting diffuser L_r^{PRD} identically irradiated.

Fig. 2 shows the geometrical conditions of a gonioreflectometer. The incoming radiation is denoted with index *i* and the reflected radiation from the sample area dA is denoted with index *r*. For a measurement in an arbitrary geometry the spectral radiance factor β depends besides the wavelength on all of these geometrical quantities:

$$\beta(\theta_i, \phi_i, \theta_r, \phi_r, d\omega_i, \mathrm{d}\omega_r, \lambda) = \frac{L_r(\theta_i, \phi_i, \theta_r, \phi_r, d\omega_i, \mathrm{d}\omega_r, \lambda)}{L_r^{PRD}} \tag{1}$$

As mentioned above the radiance of the PRD cannot be measured directly but it has to be calculated from other quantities. This can be accomplished using the radiance L_i of the irradiating lamp.

$$L_r^{PRD} = \frac{\cos\Theta_i}{\pi} \cdot \frac{A_Q}{R^2} \cdot L_i \tag{2}$$

Here, the area of the radiator is denoted by A_Q and R is the distance between the irradiating lamp and the sample, where Θ_i is the angle between the incident beam and the surface normal of the sample. The radiance L_i of the irradiating lamp can be measured by looking with the detection system directly into the lamp. The lamp has to be rotated to the 180 degree position and the sample has to be moved out of the beam path. This is something which can be carried out easily with the robot (see Fig. 3).

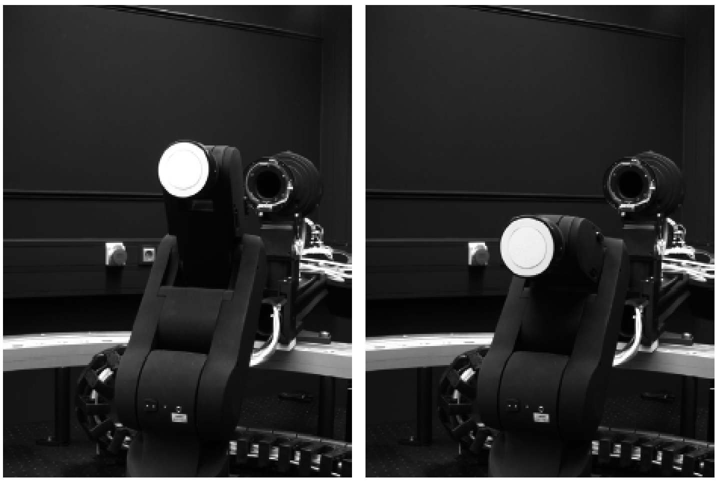


Fig. 3. Moving the robot head with a sample attached out of the detection beam path.

A Multi angle gonioreflectometer can be constructed either classically where several rotation and translation stages are assembled in order to achieve the desired degrees of freedom or using a robot as a sample holder for the device under test. It is expensive to stack several rotation and translation stages with lots of motor control units. We estimated the threefold price for such an arrangement. For economical reasons, we selected the robot solution.

3. Description of the gonioreflectometer facility

The gonioreflectometer facility consists of three major parts (see Fig. 1). A large rotation stage carrying the irradiating lamp, the 5-axis-industrial-robot as a holder for the sample under test, and the detection system (not visible in Fig. 1) consisting of a mirror-based imaging system and a monochromator for the spectrally resolved detection of the reflected radiance of the sample. In the following, these three major parts are presented in more detail.

3.1 The rotation stage with sphere radiator

The facility uses broadband irradiation of the sample with spectrally selected detection of the reflected radiation. The unfiltered broadband irradiation is generated by a special sphere radiator (Erb, 1979). This sphere radiator consists on a small integrating sphere 150 mm in diameter with an internal 250 W quartz tungsten halogen lamp. It is located on the large rotation stage with a diameter of 1.5 m and can be rotated 360° around the 5-axis-robot serving as the sample holder. The distance between sample and a precision aperture inside the lamp is 781.84 mm.
The combined adjustment of the rotation stage and the robot allows full angular control of the directed beams of incident and reflected radiation within the full half space above the surface of the sample, accomplishing also the measurement of out-of-plane reflection. The angular range of the directed radiation incident on and reflected from the sample is 0° to 85° for θ_i, θ_r and 7° to 353° for ϕ_i, ϕ_r (see also Fig. 2). The aperture angle of the source is 1.50° (solid angle $2.16 \cdot 10^{-3}$ sr), and the aperture angle of the detector 0.32° (solid angle $96.45 \cdot 10^{-6}$ sr). The rotation accuracy of the rotation stage is $\Delta\phi = \pm 0.0002°$ as measured with an indicating calliper.

3.2 The 5-axis-robot

The main part of the new gonioreflectometer is the small 5-axis-industrial-robot, model RV-2AJ from Mitsubishi Electric with an acromial height of only 550 mm. For the facility a small robot for table mounting in a conventional laboratory room was needed. To our knowledge it is the smallest 5-axis robot on the market. The robot serves as the specimen holder for the device under test, as shown in Fig. 1. The flexibility of the robot arm enables not only in-plane reflection measurements of characteristics of materials as is the case in many other national metrology institutes (Proctor & Barnes, 1996), (Chunnilall et al., 2003), (Nevas et al., 2004), but also out-of-plane configurations with arbitrary irradiating and detection angles are possible. The robot has three internal coordinate systems with the capability of making direct coordinate transformations between them.
These three coordinate systems are: A so called world coordinate system relative to the installation site, a base coordinate system relative to the footprint of the robot arm and a hand flange coordinate system relative to the hand flange. The hand flange system can be linear relocated in order to transform it into a tool coordinate system if required. The different axes of the robot are shown in Fig.4.

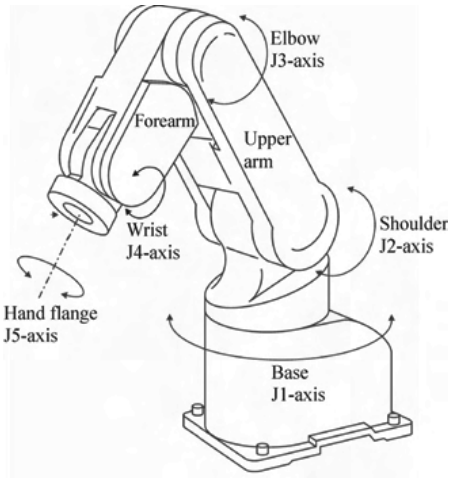


Fig. 4. Diagram showing the different axes of the five-axis-robot.

The coordinate system transform capability substantially facilitates the programming of the movements, translations and rotations of the robot system since no details like Euler's angles have to be considered. It is possible to address rotations of the 5 axis directly and also to use one of the three coordinate systems in order to execute compound movements of different axis simultaneously.
The mode of operation for the robot in the gonioreflectometer facility is atypical for an industrial robot. Normally a robot is working in front or beside his base unit, like in industrial production. In the present case the robot is part of a scientific measurement system and responsible for the positioning of the device under test, respectively the sample. In order to do that the hand flange of the robot is working above the base unit, one can say above his head. The robot is working in Master-Slave operation for our application. This is also unusual for an industrial robot. The standard case for a robot is to develop a sequence program and transfer it into the MCU (micro controller unit). The robot is than autonomous and is cyclical doing his work, only influenced by control inputs or the handing over of software parameter. In the given case we consulted the manufacturer, Mitsubishi Electric, and asked for expanded control command capabilities. They communicated us some special, normally unpublished control commands for Master-Slave operation. These are commands normally only used by the manufacturer for testing of the robot. Using these special control commands the robot system is now only one device upon the other measurement instruments forming the gonioreflectometer facility attached to it via RS-232 interface. The whole facility is fully automated and the measurement operation is controlled by a self-written routine programmed in Visual Basic running on a standard personal computed.
A mentioned above, the robot has to be operated in a slightly unusual position for an industrial robot, as shown in Fig. 1, with the hand flange above the basic platform collinear with the J1 base axis. The other robot axes (J2 to J4) are positioned in such a way as to ensure that the surface of the calibration sample is always located within the common rotation axis of the rotation table and the J1 axis of the robot which can only be achieved for a limited range with a sample thickness of up to 60 mm.
The robot is able to carry and position large samples with an outer diameter of up to 0.5 m and a maximum weight of up to 2 kg. The position accuracy is 0.02 mm for arbitrary movements within the whole operating range. The rotation accuracy of the J1 base axis is $\Delta\phi = \pm 0.002°$.

3.3 The detection system

The direction of the detection path is fixed due to the fact that a triple grating half-meter monochromator is used for spectral selection of the reflected radiation. The current wavelength range for measurements of diffuse reflection is 250 nm to 1700 nm.
The facility uses two-stage mirror-based 10:1 imaging optics to map a 20 mm circular area on the sample onto the 2 mm wide entrance slit of the monochromator. This results in a 3 nm spectral resolution within the spectral range 250 nm to 900 nm and a 6 nm bandpass within the 900 nm to 1700 nm range, depending on the gratings used.
Four different detectors behind the monochromator are used for detecting the radiance signals of the incident and reflected beams. All the signals are detected with a picoamperemeter and transferred to a computer for data storage and analysis.

4. Example for an out-of-plane measurement

As mentioned above, the flexibility of the robot arm allows for out-of-plane measurements of reflection characteristics of materials with arbitrary incident and reflection angle within the whole half space above the sample surface. Thus, the appearance of surfaces can be quantified under arbitrary angles between light source and observer.
In order to position the robot and the large rotation stage in the right angular position for such an out of plane measurement one has to calculate several dot products. These are the dot products between the vector of the direction of the incident radiance L_i, the vector of the surface normal of the sample N and the vector of the direction of the reflected radiance L_r. This has to be calculated in two different coordinate systems. In the first coordinate system with an orientation similar to Fig.2, the three different vectors are expressed in a simple way. The sample is located in the xy-plane with the vector of the surface normal in z-direction. The angles of the incident and reflected radiation can easily be expressed via the angle pairs ϕ_i, Θ_i and ϕ_r, Θ_r.
The second coordinate system corresponds to the specific geometrical conditions at the robot gonioreflectometer, see Fig. 5. The initial position of the sample is now tilted around 90° and located in the yz-plane.

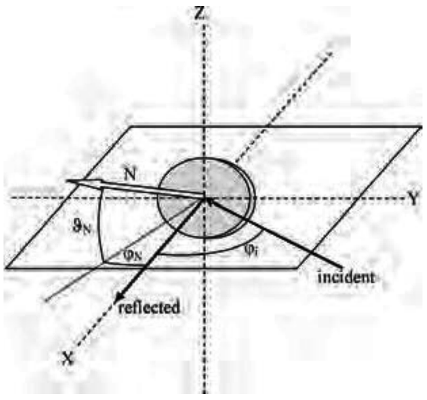


Fig. 5. Coordinate system and angles for out of the plane measurements at the robot gonioreflectometer facility.

The both vectors of the incident and reflected beams are restricted to the xy-plane (i.e. z = 0) according to the construction of the gonioreflectometer. The lamp generating the incident light is mounted on the large rotation stage and can only be rotated in this plane. The detection path for the reflected radiation lies also in this plane and is fixed due to the table mounted mirror optics. This fixed direction coincides with the zero degree position of the rotation stage. The orientation of the surface of the sample is characterised due to the vector N of the surface normal which is perpendicular to the surface and can be rotated with the robot to every angular position φ_N, ϑ_N within the whole hemisphere above the xy-plane. With the calculation of these three dot products in the two different coordinate systems and some conversion of the equations it is possible to compute the required angles φ_I, φ_N and ϑ_N which had to be adjusted at the facility for an off-axis measurement.

$$\varphi_I = \arccos(\sin\Theta_i \sin\phi_i \sin\Theta_r + \cos\Theta_i \cos\Theta_r) \tag{3}$$

$$\varphi_N = \arctan\left(\frac{\cos\theta_i - \cos\theta_r \cos\varphi_I}{\cos\theta_r \sin\varphi_I}\right) \tag{4}$$

$$\vartheta_N = \arcsin\left(\frac{\cos\theta_r}{\cos\varphi_N}\right) \tag{5}$$

Fig. 6. shows the results of two measurements in an out-of-plane configuration of the radiance factor of an opal glass (glossy and matt side). The incident angle is varied from 0° to 85° in 5° steps in the xz-plane perpendicular to the yz-plane where the reflection angle (fixed at θ_r = 25°) is located, see Fig. 7. It can be seen that the radiance factor of the matt and glossy side of the opal glass is quite different in this configuration.

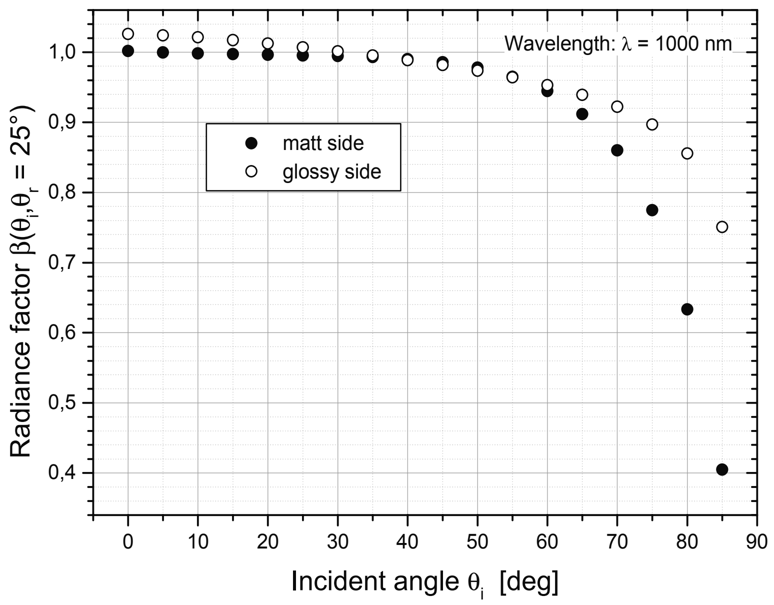


Fig. 6. Out of plane measurement on two sides of an opal glass reflection standard.

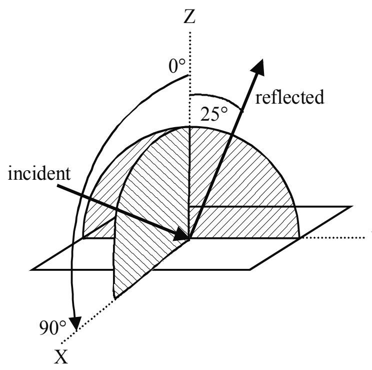


Fig. 7. Diagram showing the geometrical conditions of an out of plane measurement. The incident angle is varied from 0° to 85° in a plane perpendicular to the plane of the reflection angle (fixed at $\Theta_r = 25°$).

In order to vary the incident angle in this manner, three angles of the facility, the angle of the rotation stage φ_i and the angles of the sample surface normal φ_N and ϑ_N had to be adjusted simultaneously. For this measurement the angle of the rotation stage φ_i varies from 25° to 90°, the angle φ_N of the surface normal from 25° to 0° and the angle ϑ_N from 0° to 25°, see Tab. 1 and Fig. 8.

Incident angle of radiation on the sample [°]	Angle of rotation stage φ_i [°]	Angel of surface normal φ_N [°]	Angle of surface normal ϑ_N [°]
0	25	25	0
10	26.81	23.28	9.37
20	31.61	19.46	16.01
30	38.29	15.40	19.94
40	46.03	11.85	22.18
50	54.37	8.86	23.47
60	63.05	6.31	24.24
70	71.94	4.06	24.69
80	80.95	1.98	24.93
90	90	0	25

Table.1. Values of the involved angles (in 10 degree steps) for the presented out of plane reflection measurement.

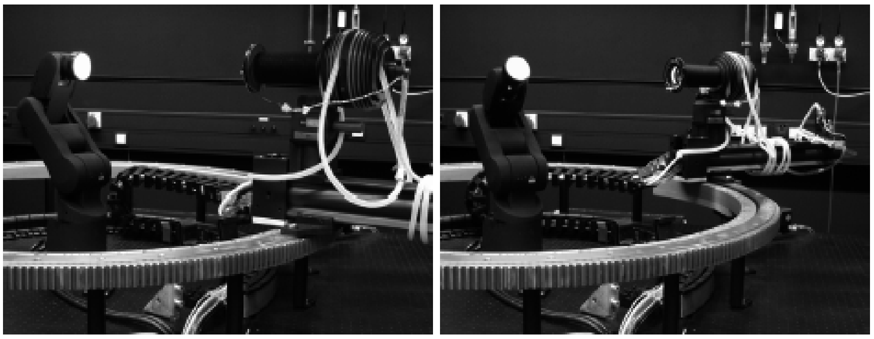


Fig. 8. Photo of the starting and end position for the explained out of plane measurement.

5. Rotational radiance invariance of diffuse reflection standards

Another measurement which can be easily realized with the robot as a sample holder are measurements of the rotational invariance of the sample reflection signal. In order to do that only the J5-axis of the robot has to be rotated around 360°.
Standards for diffuse reflection calibrations must not have any texture, i.e. their reflection behaviour should be independent of their rotational orientation. Explicit goniometric measurements on a selection of samples made of different materials showed variations in the reflected radiance of up to 2.5 % depending on the rotational orientation of the sample despite the absence of an obvious internal structure.
Different reflection standards were measured. Their radiance was recorded while they were rotated 360° around an axis collinear with their surface normal (J5-axis of the robot). The radiance signal was taken from a circular area 20 mm in diameter. The measurements were taken at a wavelength of 950 nm. The reflection geometry was 45/0, that is an angle of 45° between the irradiating radiation and the surface normal of the sample and a reflection under 0° degree from the sample, in an in-plane configuration for all of the measurements.

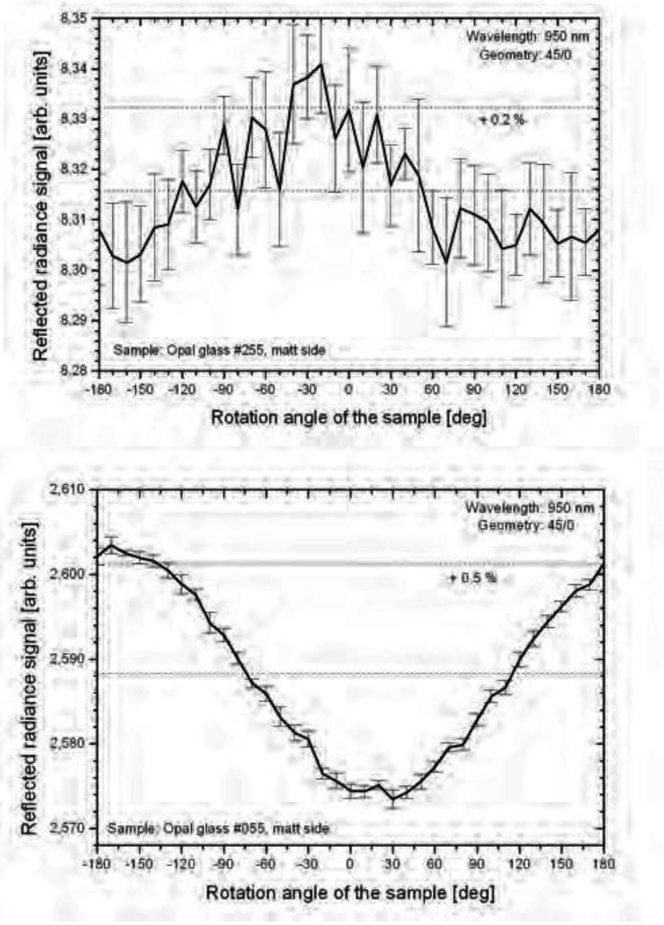


Fig. 8. Orientation dependence of the reflected radiance for two different opal glasses.

Fig. 8 shows the results of measurements on two samples. The samples are in both cases opal glasses from the same manufacturer but with different batch numbers. The reflected radiance signal from the samples should be in both cases independent of the rotation angle, which is only the case for the opal glass with #255, where the variances are in the range of the measurement uncertainty. The reason for this differences has to be analyzed in more detail in the future. Via visual inspection there is no obvious difference between the two samples.
From these measurements the following can be deduced. Standard materials of diffuse reflection should be handled with caution when there are used in directional reflection geometries. The absolute reflection depends on the rotational orientation of the sample relative to the plane spanned by the beams of the incident and reflected radiation. Even small rotations about the surface normal can lead to deviations in the reflected radiance almost in the percent range. This is more than the measurement uncertainties of most of the goniometric calibration facilities. This underlines the necessity to indicate the orientation of samples during their calibration and also to reproduce this orientation when making adjacent measurements using the calibration data.

6. Conclusion

The new gonioreflectometer of the PTB enables high precision goniometric measurements of diffuse reflecting samples within the whole hemispheric range above the surface of the sample, due to its out-of-plane capabilities, by using a 5-axis-robot.
The whole facility was build using standard laboratory and engineering equipment. The five-axis robot model RV-2AJ from Mitsubishi Electric could be used without any modification for this unusual field of application.
The current wavelength range for measurements and calibrations is 250 nm to 1700 nm. It is planned to extend this wavelength range up to 2500 nm in the near future. It is also planned to install an additional lamp with increased UV output power in order to improve the current uncertainty budget for measurements in the UV range.

7. Acknowledgments

The authors thank B. Campe and S. Teichert for the construction and manufacture of several parts of the gonioreflectometer facility.

8. References

Chunnilall, C.J.; Deadman, A.J.; Crane, L.; Usadi, U. (2003). NPL scales for radiance factor and total diffuse reflectance. *Metrologia*, 40, S192-S195, ISSN 0026-1394

Erb, W. (1979). Kugelstrahler mit homogener Strahldichte. *Phys-Techn. Bundesanstalt, Jahresbericht 1979*, 170-171, ISSN 0340-4366

Erb, W. (1980). Computer-controlled gonioreflectometer for the measurement of spectral reflection characteristics. *Appl. Opt.*, Vol. 19, No. 22, 3789-3794, ISSN: 0003-6935

Hünerhoff, D.; Grusemann, U.; Höpe, A. (2006). New robot-based gonioreflectometer for measuring spectral diffuse reflection. *Metrologia*, 43, S11-S16, , ISSN 0026-1394

Nevas, S.; Manoocheri, F.; Ikonen, E. (2004). Gonioreflectometer for Measuring Spectral Diffuse Reflectance. *Appl. Opt.*, Vol. 43, No. 35, 6391-6399, ISSN: 0003-6935

Proctor, J.E.; Barnes, P.Y. (1996). NIST High Accuracy Reference Reflectometer-Spectrophotometer. *J. Res. Natl. Inst. Stand. Technol.*, 101, 619-627, ISSN 0160-1741.

Serpentine Robots for Industrial Inspection and Surveillance

Grzegorz Granosik[1], Johann Borenstein[2], Malik G. Hansen[2]
[1]*Technical University of Lodz, POLAND*
[2]*University of Michigan*[*], *Ann Arbor, MI, USA*

1. Introduction

Urban search and rescue, industrial inspections, and military intelligence have one need in common: small-sized mobile robots that can travel across the rubble of a collapsed building, squeeze through small crawl-spaces to take measurements or perform visual inspections, and slither into the shelter of insurgents to gather intelligence. Some of these areas are not only difficult to reach, but may also present safety and health hazards to human inspectors. One species of mobile robots that promises to deliver such hyper-mobility is the so-called serpentine or snake robot (see Figure 1). Serpentine robots typically comprise of three or more rigid segments that are connected by 2- or 3-degree-of-freedom (DOF) joints. The segments typically have powered wheels, tracks, or legs to propel the vehicle forward, while the joints may be powered or unpowered. Desired capabilities for such a robot are:

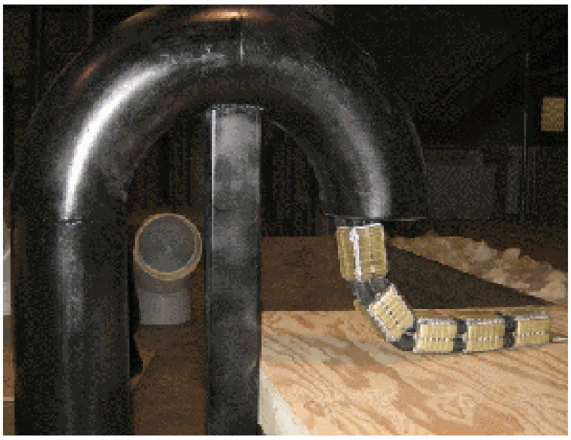


Fig. 1. The OmniTread Model OT-4 serpen-tine robot entering an "Inverted-'J' ventilation duct at SwRI[†].

[*] The OmniTread work was conducted at the University of Michigan where Dr. Granosik co-developed the "Omni's" as a Visiting Researcher from 2002-2004.
[†] The OmniTread robots were independently tested at the Southwest Research Institute (SwRI). Most of the OmniTread photographs in this chapter were taken at SwRI during the successful traverse of the shown obstacle.

- ability to traverse rugged terrain, such as concrete floors cluttered with debris, or unfinished floors such as those found on constructions sites;
- ability to fit through small openings;
- ability to climb up and over tall vertical steps;
- ability to travel inside and outside of horizontal, vertical, or diagonal pipes such as electric conduits or water pipes;
- ability to climb up and down stairs;
- ability to pass across wide gaps.

This chapter begins with an extended literature review on serpentine robots in general, and then focuses on the concept and features of the OmniTread family of serpentine robots, which were designed and built at the University of Michigan's (UM's) Mobile Robotics Lab. Along the way, we discuss the evolution of OmniTread robots (or "Omnis" in short), showing inheritance of valuable features, mutation of others, and elimination of disadvantageous designs. In the Experiment Results Section, photographs of successful obstacle traverses illustrate the abilities of the Omnis. The chapter concludes with our prognosis for future work in this area.

2. Serpentine Robots

Serpentine robots belong to the group of hyper-redundant articulated mobile robots. This group can be further divided based on two characteristic features: the way the forward motion of the robot is generated and the activity of its joints, as shown in Table 1. As our work is focused on serpentine robots we will limit the following literature review to this scope.

The first practical realization of a serpentine robot, called KR-I, was introduced by Hirose and Morishima (1990) and later improved with version KR-II (Hirose et al., 1991). This first serpentine robot was large and heavy, weighing in at 350 kg. The robot comprised of multiple vertical cylindrical segments on powered wheels (tracks in KR-I) that give the mechanism a train-like appearance. Vertical joint actuators allow a segment to lift its neighbors up, in order to negotiate steps or span gaps.

More recently, Klaassen and Paap (1999) at the GMD developed the Snake2 vehicle, which contains six active segments and a head. Each round segment has an array of 12 electrically driven wheels evenly spaced around its periphery. These wheels provide propulsion regardless of the vehicle's roll angle. Segments are interconnected by universal joints actuated by three additional electric motors through strings. Snake2 is an example of a robot that is inspired by the physiological structure of snakes where wheels replace tiny scales observed on the bodies of some real snakes. Snake2 is equipped with six infrared distance sensors, three torque sensors, one tilt sensor, two angle sensors in every segment, and a video camera in the head segment. Snake2 was specifically designed for the inspection of sewage pipes.

Another serpentine robot designed for sewer inspection was developed by Scholl et al. (2000) at the Forschungszentrum Informatik (FZI) in Germany. Its segments use only two wheels but the actuated 3-DOF joints allow full control over each segment's spatial orientation. The robot is able to negotiate tight 90° angled pipes and climb over 55 cm high obstacles. One segment and its joint are about 20 cm long. The sensor suite of this robot is similar to that of Snake2. The development of sewer inspection robots is continued in the joint project MAKROplus (Streich & Adria, 2004).

While wheeled serpentine robots can work well in smooth-walled pipes, more rugged terrain requires tracked propulsion. To this effect Takayama and Hirose (2000) developed the Soruyu-I crawler, which consists of three segments. Each segment is driven by a pair of tracks, which, in turn, are all powered simultaneously by a single motor, located in the center segment. Torque is provided to the two distal segments through a rotary shaft and universal joints. Each distal segment is connected to the center segment by a special 2-DOF joint mechanism, which is actuated by two lead screws driven by two electric motors. The robot can move forward and backward, and it can change the orientation of the two distal segments in yaw and pitch symmetrically to the center segment. One interesting feature is the ability of this robot to adapt to irregular terrain because of the elasticity of its joints. This elasticity is provided by springs and cannot be actively controlled.

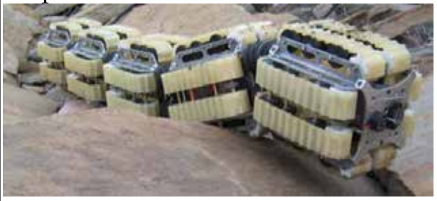
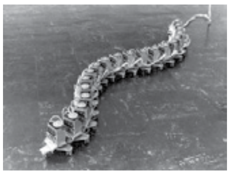
	External propulsion element: legs, wheels, tracks	Movement is generated by undulation
Active joints	Serpentine robots:  OmniTread Moira Kohga Soryu Snake 2 Robot MAKROplus Robot Pipeline Explorer	Snake-like robots:  ACM (Hirose, 1993) A CM-R3 (Mori & Hirose, 2002) Slim Slime Robot (Ohno & Hirose, 2000)
Passive joints	Active wheels – passive joints robots: Genbu 3 (imura & Hirose, 2002)	

Table 1. Articulated mobile robots.

A different concept using unpowered joints was introduced by Kimura and Hirose (2002) at the Tokyo Institute of Technology. That robot, called Genbu, is probably the only serpentine robot with unpowered joints. The stability of the robot and its high mobility on rough terrain are preserved by large-diameter wheels (220 mm). The control system employs position and torque

feedback sensors for the passive but rigid joints. Springs are used to protect the electric motors from impact, although the stiffness of the springs cannot be controlled during operation.
Another robot incorporating a combination of passive and active joints as well as independently driven and coupled segments is KOHGA developed by Kamegawa et al. (2004). This robot implements a smart design feature: besides a camera in the front segment, there is a second camera in the tail section that can be pointed forward, in the way a scorpion points its tail forward and over-head. This "tail-view" greatly helps teleoperating the robot.
The concept of joining several small robots into a train to overcome larger obstacles was used by researchers from Carnegie Mellon University in their Millibot Train (Brown et al., 2002). This robot consists of seven electrically driven, very compact segments. The diameter of the track sprockets is larger than the height of each segment, which allows the robot to drive upside-down. Segments are connected by couplers for active connection and disconnection, but the joints have only one DOF. Each joint is actuated by an electric motor with a high-ratio harmonic gear and slip clutch. It provides sufficient torque to lift up the three front segments. The robot has been demonstrated to climb up a regular staircase and even higher steps. However, with only one DOF in each joint the vehicle is kinematically limited.
A serpentine robot that uses tracks for propulsion and pneumatics for joint actuation is MOIRA (Osuka & Kitajima, 2003). MOIRA comprises four segments, and each segment has two longitudinal tracks on each of its four sides, for a total of eight tracks per segment. The 2-DOF joints between segments are actuated by pneumatic cylinders. We believe that the bellows-based joint actuators used in our OmniTread have a substantial advantage over a cylinder-based design, as the discussion of our approach in the next section will show.
The newest construction from NREC (National Robotics Engineering Center) is Pipeline Explorer – robot designed and built for inspection of live gas pipelines (Schempf et al., 2003). This robot has a symmetric architecture. A seven-element articulated body design houses a mirror-image arrangement of locomotor (camera) modules, battery carrying modules, and support modules, with a computing and electronics module in the middle. The robot's computer and electronics are protected in purged and pressurized housings. Segments are connected with articulated joints: the locomotor modules are connected to their neighbors with pitch-roll joints, while the others – via pitch-only joints. These specially designed joints allow orientation of the robot within the pipe, in any direction needed.
The locomotor module houses a mini fish-eye camera, along with its lens and lighting elements. The camera has a 190-degree field of view and provides high-resolution color images of the pipe's interior. The locomotor module also houses dual drive actuators designed to allow for the deployment and retraction of three legs equipped with custom-molded driving wheels. The robot can sustain speeds of up to four inches per second. It is fully untethered (battery-powered, wirelessly controlled) and can be used in explosive underground natural gas distribution pipelines.

3. The Omnis Family

3.1 Robots Description

Since 1998 the Mobile Robotics Lab at the University of Michigan (UM) has focused on the development of serpentine robots. Figure 2 shows our first serpentine robot, the OmniPede (Long et al., 2002). Although we conceived of the idea for the OmniPede independently, we later found that nature had produced a similar design: the millipede (see Figure 3a). In the OmniPede, UM introduced three innovative functional elements: (1) propulsion elements (here: legs) evenly located around the perimeter of each segment; (2) pneumatic power for joint actuation; and (3) a single so-called "drive shaft spine" that transfers mechanical power to all segments from a single drive motor.

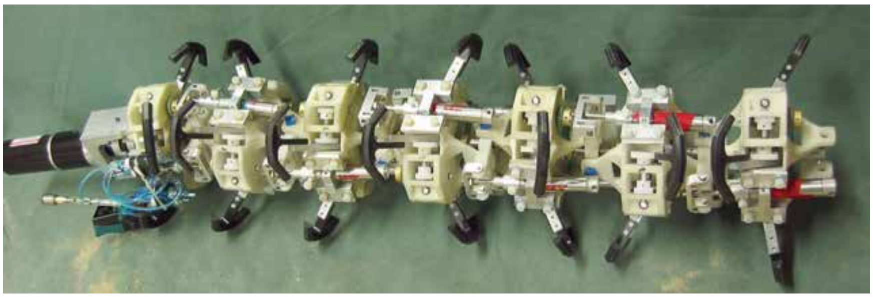


Fig. 2. OmniPede.

One of the key features in the design of the OmniPede is that each leg has only one degree of freedom (DOF). A "leg" and its associated "foot" look like the cross section of an umbrella. The trajectory of the foot and the orientation of the leg are determined by a simple mechanism as shown in Figure 3b. The geared 5-bar mechanism moves the leg so that the foot makes contact with the terrain while performing the backward portion of its motion (which is the portion that propels the vehicle forward). Then the foot disengages from the terrain while it performs the forward portion of its motion (as shown in Figure 3c). As a result the OmniPede moves forward.

By having only one DOF per leg instead of the two or three DOF that most other legged vehicles have, the number of required actuators is reduced. The price that is paid for the reduced complexity, weight, and cost is having less control over the position and orientation of the legs. However, we considered this to be a small sacrifice because with the OmniPede precise leg positioning is unimportant. Also, the reduced complexity of the legs offers further advantages, as described below.

The OmniPede consists of seven identical segments, with the tail segment housing the motor. Each segment has four of the legs shown in Figure 3b arranged circularly on its circumference and evenly spaced at 90-degree intervals. The legs are arranged this way so that no matter which part of the OmniPede is in physical contact with the environment, contact is always made through some of the feet. The segments are connected through articulated joints, which allow two DOF between the segments. These two DOF are each independently controlled with a pneumatic piston by means of a four-bar mechanism. This feature provides the OmniPede with the versatility that was lost by linking the legs kinematically. The joint actuators enable the OmniPede to lift its front end up and onto obstacles much the same way a millipede (or a worm, or a snake) does. Another key feature of the OmniPede design is that the motion of each leg is kinematically linked to a common drive shaft, called the drive shaft spine, that runs through the centre of the vehicle. This allows all of the legs to be driven by just one actuator, which supplies torque to the common drive shaft. Also, because the legs are all kinematically linked by the common drive shaft, the phase differences between all of the legs are fixed.

Unfortunately, the OmniPede never reached the mobility level of millipedes. Partially because of the scale factor (our robot is much larger than its natural counterpart) and mainly because we could not produce the same foot density (number of feet per side area of robot) as nature did. Therefore, our design needed modifications; we could say it was real evolution. We abandoned the idea of few discrete legs altogether, and instead adopted the abstract idea of a continuous, dense "stream of legs;" we noticed that trace of each foot can be seen as track, as schematically shown in Fig. 3. Using tracks (executing rotation) we improved efficiency of driving mechanism. We also improved design of robot's joints by introducing Integrated Joint Actuator (described in detail later). And finally, we preserved the idea of "drive shaft spine" with a single drive motor.

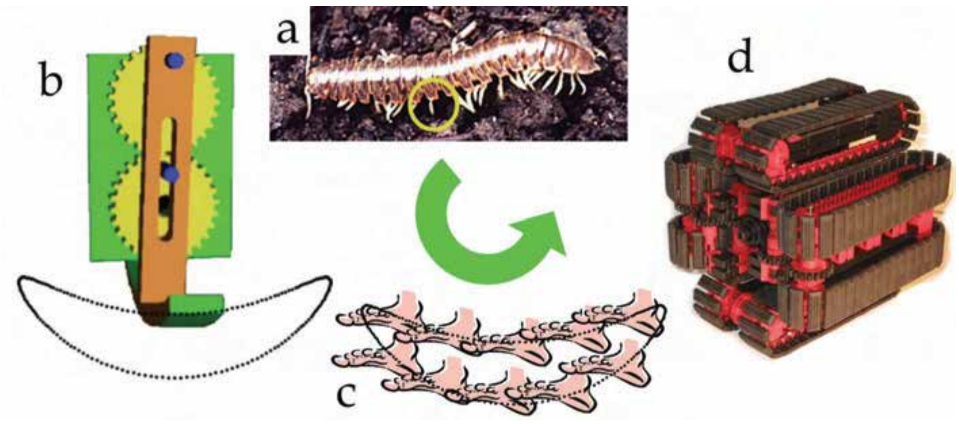


Fig. 3. Evolution of driving system: from the legged OmniPede to the tracked OmniTread. a – millipede, b – 1DOF leg of the OmniPede, c – propulsion idea of OmniPede's foot, d – Proof-of-concept prototype of the OmniTread: In an abstract sense, a moving track with grousers can be seen as a continuous stream of moving legs.

From the study of the OmniPede, and from the observed shortcomings of its legged propulsion system, we derived important insights about the design of serpentine robots. These insights led to the development of the far more practical "OmniTread" serpentine robot, shown in Table 1. This version of the OmniTread, later called "OT-8," has five segments and four pneumatically actuated 2-DOF joints. The size of each segment is 20×18.6×18.6 cm (length × width × height). Each joint space is 6.8 cm long. The entire robot is 127 cm long and weighs about 13.6 kg. The OmniTread is teleoperated and has off-board power sources (electric and pneumatic).

In May 2004 we began work on the latest and current version of the OmniTread, the OT-4. The number designation comes from its dominant design parameter: the OT-4 can fit through a hole 4 inches (10 cm) in diameter, whereas the OT-8 can fit through an 8-inch diameter hole.

The OmniTread OT-4 comprises seven segments and six 2-DOF joints, as shown in Figure 4. The segment in the centre is called "Motor Segment" because it houses the single drive motor. All other segments are called "Actuation Segments" because they house, among others, the control components for the pneu-matic actuators. Segments #1 and #7 are able to hold some payload, such as cameras, microphones, and speakers. Segments #2 and #6 can hold one micro air-compressor each, for pneumatic power. Segments #3 and #5 hold Li-Polymer batteries. The OT-4 can carry onboard energy resources for up to 75 minutes of continuous, untethered driving on easy terrain.

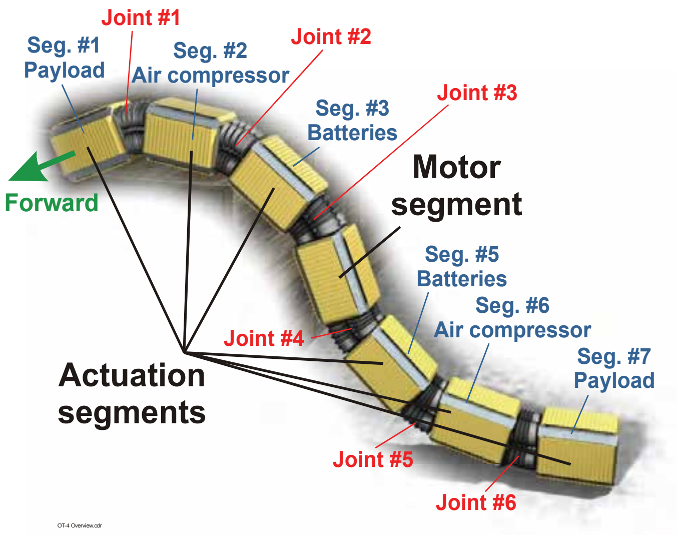


Fig. 4. Nomenclature for segments and joints of OmniTread OT-4.

The OT-8 and OT-4 share these mostly unique features:

1. Tracks-all-around each segment. This design aims at maximizing the coverage of the whole robot body with moving tracks. This feature is tremendously important on rugged terrain since the long, slender body of serpentine robots rolls over easily on such terrain. The disadvantage of this design is the greater complexity (each segment needs four drive systems) and the space needed for four drive systems.
2. The 2-DOF joints are actuated by pneumatic bellows, which produce sufficient torque to lift the three (two in case of OT-8) leading or trailing segments up and over obstacles. More importantly, pneumatic bellows provide natural compliance with the terrain. This feature assures optimal traction in bent pipes and on rugged terrain.
3. A single electric drive/motor in the center segment provides rotary power to each segment through a so called “drive shaft spine” that runs through the whole length of the robot. We believe this design to be more weight and power efficient than individual motors in each segment. The weaknesses of this design are a limit to the maximal bending angle of the joints of ~40 degrees, as well as inefficiency when articulating the joints.

3.2 Tracks All Around

Our doctrine in the design of both OmniTread models is the maximal coverage of all sides of the robot with moving tracks. This doctrine is based on two reasons:

1. Serpentine robots inevitable roll over when traveling over rugged terrain. Since terrain conditions may not allow the robot to upright itself immediately, only coverage of all sides with propulsion elements can assure continuation of the mission after a roll over.
2. Any contact between an environmental feature and a robot’s inert (i.e., not propelling) surface impedes motion or entirely stops the robot (i.e., the robot gets “stuck”). In contrast, any contact between an environmental feature and a

propulsion surface produces motion. On rugged terrain, such as the rubble of a collapsed building, it is quite common that not just the bottom side of the robot, but also its left and right side make contact with terrain features.

To express this relation quantitatively, we define the term "Propulsion Ratio" P_r. P_r is measured as the surface area that provides propulsion, A_p, divided by the total surface area, A_p+A_i

$$P_r = A_p/(A_p + A_i) \tag{1}$$

where A_i is the inert surface area of the body. To further clarify, A_p is the sum of all surface areas that could provide propulsion if in contact with the environment, while A_i is the sum of all surface areas that could not. P_r is not only a function of the robot's geometry, but also of the application domain. For example, on flat and hard terrain, P_r for a conventional automobile is 1.0 since only the wheels can be in contact with the terrain. That's because in a car no inert area of the periphery could possibly be in contact with the ground, that is, $A_i = 0$. However, on soft terrain the wheels sink into the ground and on rugged terrain obstacles protrude out of the ground, resulting in potential contact between the ground and portions of the inert body periphery. In this case the propulsion ratio P_r is undesirably low. In practice, serpentine robots with a low propulsion ratio get stuck very easily when trying to move over rugged terrain. In order to increase the propulsion area A_p and thus the propulsion ratio P_r, we cover all sides of the OmniTread with extra-wide tracks (as is also advised by Blitch, 2003). We also took extensive measures to reduce the space (and thus, the inert area A_i) between the segments. Environments, in which robots with high propulsion ratios excel, are dense underbrush, rubble, and rocks (see Fig. 5). In these environments contact can occur anywhere, and robots that have propulsion surfaces only on the bottom are always at risk of being stalled due to excessive, nonpropelling contact. The propulsion ratio for the OT-4 is 0.59 while that of our earlier OmniTread OT-8 is 0.42.

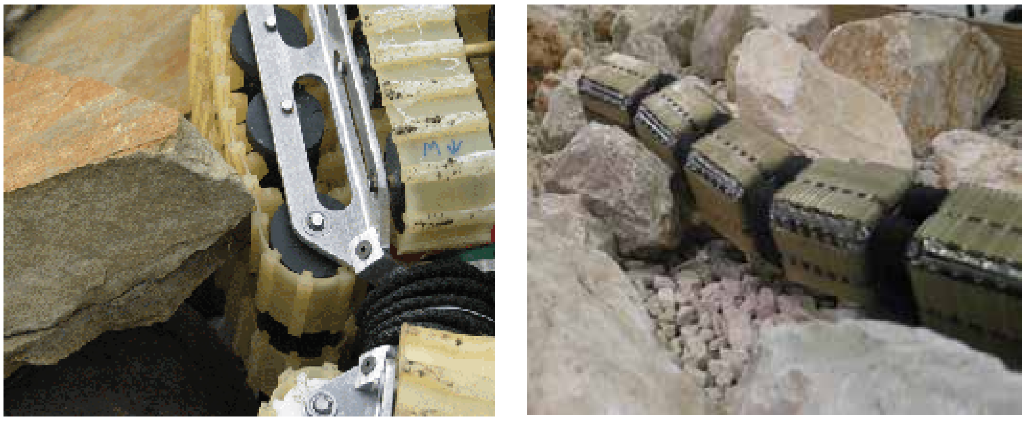

a b

Fig. 5. Tracks all around: As the OmniTreads drive through rockbeds, it becomes apparent that side tracks help provide forward propulsion. a – OT-8; b – OT-4.

3.3 Pneumatic Joint Actuation

During our work with serpentine robots, we spent a significant amount of time on the analysis and formulation of requirements for joint actuators in serpentine robots. Listed here are the four most important ones:

1. By definition, serpentine robots are relatively long compared to their diameter, so that their lead segments can reach up and over a high obstacle while still being able to fit through small openings, as shown in Fig. 6. However, lifting the lead segments requires a significant amount of torque, which is particularly difficult to generate in slender serpentine robots, where the lever arm for a longitudinal lifting force is limited by the robot's small diameter. One key requirement for serpentine robots is thus that they employ joint actuators of sufficient strength to lift up two or more of their lead or tail segments.
2. Another key requirement is that serpentine robots should conform to the terrain compliantly. This assures that as many driving segments as possible are in contact with the ground at all times, thereby providing effective propulsion. Serpentine robots that don't conform compliantly require extremely complex sensor systems to measure contact forces and to command a momentary angle for each non-compliant joint so as to force contact with the ground. Such actively controlled compliance has not yet been successfully demonstrated, and may well be unfeasible for many more years.

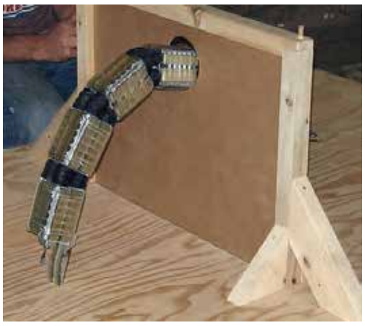


Fig. 6. OT-4 passes through a 10-cm (4") diameter hole, high above ground. Extendable "flipper tracks" in the distal segments allows the robot to straddle the hole without having to support its weight on the sharp edges of the hole in this test at SwRI.

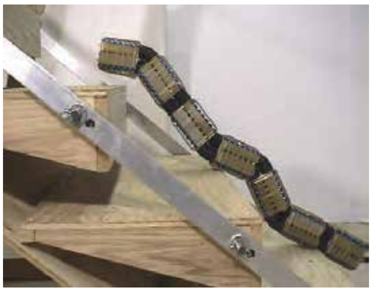


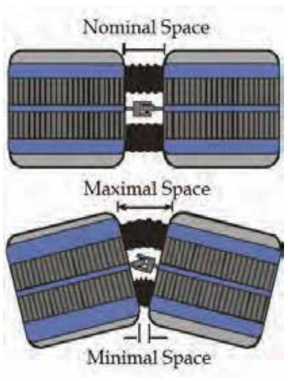
Fig. 7. Joint strength and stiffness: OT-4 lifting up three lead segments in order to reach the next step.

3. At times it is necessary to increase the stiffness of a joint, for example to reach over an obstacle, or for crossing a gap (see Fig. 7). Alternatively, it may be necessary to

adjust the stiffness to an intermediate level, for example, when the lead segment leans against a vertical wall while being pushed up that wall by the following segments. Thus, serpentine robots should be capable of adjusting the stiffness of every DOF individually and proportionally.

4. Large amounts of space dedicated to joints dramatically increase the amount of inert surface area. Therefore, joint actuators should take up as little space as possible, to reduce the size of space occupied by joints (called "Joint Space").

Moreover, it is obvious that the weight of the actuators should be minimal and joint angles in serpentine robots should be controllable proportionally, to provide full 3D mobility. Extensive studies of these requirements and of joint actuators potentially meeting these requirements led to the second unique design feature of the OmniTread, the use of pneumatic bellows for actuating the joints. Our research (Granosik & Borenstein, 2005) shows that pneumatic bellows meet all four of the above requirements better than any other type of actuator. In particular, pneumatic bellows provide a tremendous force-to-weight ratio, and they fit perfectly into the otherwise unusable (since varying) space between segments.



a

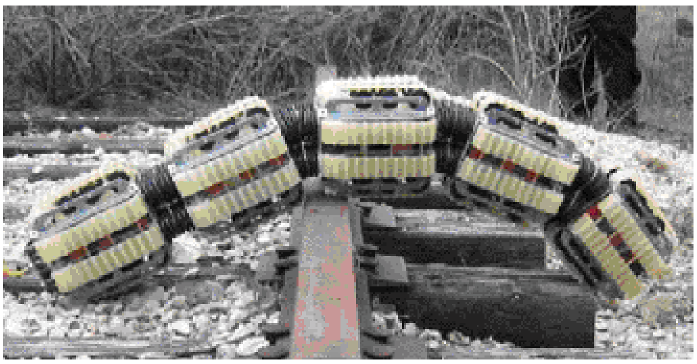

b

Fig. 8. Integrated Joint Actuator: a – In serpentine robots the shape of the so-called "Joint Space" varies with the angles of the joint. At extreme angles, there are some regions of Joint Space where there is almost no free space for mounting rigid components. However, the bellows of the OmniTreads conform to the available space. b – In the OT-8, Joint Space is only 6.8 cm long while segments are 20 cm long. This design helps produce a very favorable Propulsion Ratio P_r. The obvious advantage is the OmniTread's ability to rest its weight on some obstacle, such as this railroad rail, without getting its Joint Space stuck on it. The sharp edge of the hole in Fig. 6, however, was meant to penetrate joint space, as an additional challenge.

The latter point is illustrated in Figure 8a, which shows that parts of Joint Space may be small at one moment, and large at the next, depending on the bending of the joint. If we wanted to use Joint Space for housing electronics or other rigid components, then the size of that component would be limited by the dimensions of the "minimal space" shown in Figure 8a. Contrary to rigid components, pneumatic bellows fit into such varying spaces perfectly: bellows expand and contract as part of their intended function, and they happen to be smallest when the available space is minimal and largest when the available space is maximal. From the point of space utilization, pneumatic bellows are thus a superbly elegant solution, because joint actuators take up only Joint Space, and very little of it, for that matter. Therefore, we call our bellows-based pneumatic system "Integrated Joint Actuator" (IJA). In contrast, pneumatic cylinders or McKibben muscles, as well as electric or hydraulic actuators, would all require space within the volume of the segments or much larger Joint Space. To further illustrate our point about small versus large Joint Spaces, we included Figure 8b, which shows how the OT-8 successfully traverses a relatively narrow-edged obstacle, thanks to its very short joints. If the joints were longer than the rail's width, then the robot would necessarily get stuck on it.

3.4 Motor and Gear Train

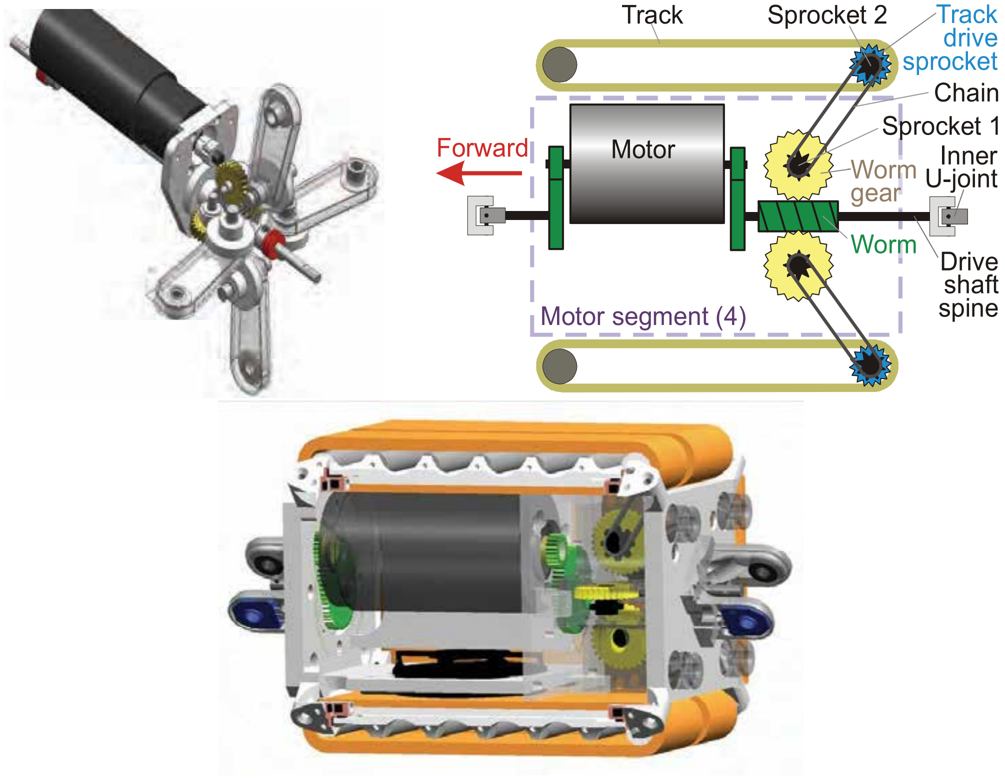


Fig. 9. CAD drawings of the OmniTread gearboxes: a – OT-8 gearbox, shown here for the motor segment. b – Schematic of the OT-4 drive system, shown here for the motor segment. c – CAD drawing of the OT-4 motor segment.

In case of OmniTread OT-8 a single 70W drive motor (Model RE36 made by Maxon) located in the central segment provides torque to all tracks on all five segments via the drive shaft spine. The drive shaft spine is an axle that runs longitudinally through all segments. Universal joints let the axle transfer torque at joint angles of up to 30°. Within each segment there is a worm on each driveshaft that drives four worm-gears offset 90° from each other, as shown in Figure 9a. Each worm gear runs two spur gears ending in chain drives to deliver power to the sprocket shafts. The purpose of the spur gears is to bring the chain back to center again so that the two tracks on each side can be of equal width. The chain drive is

very slim and therefore minimizes the gap between the tracks. The total gear reduction from the motor to the sprockets is 448:1. The drive system and chain drive is sealed to prevent dirt from entering the mechanism.
A similar drive train mechanism was developed for the OT-4. The drive shaft spine comprises seven rigid shafts that are connected by six universal joints. The universal joints are concentrically located within the gimbal joints that link the segments. On each shaft segment is a worm. Four worm gears feed off that worm on the drive shaft as shown in Figure 9b. Each worm gear drives a chain that drives the track sprocket. These worm gears can be engaged with worm or disengages from it by means of electrically actuated bi-stable clutches. The OT-4 has one such a clutch for each of its 7×4 = 28 tracks. These clutches allow the operator to engage or disengage each track individually. Thus, tracks not in contact with the environment can be disengaged to reduce drag and waste of energy. If the robot rolls over, the tracks that come into contact with the ground can be reengaged by the operator.
The drive shaft is supported by two ball bearings on each end of the gearbox to retain good tolerances within the gearbox. The other end of the drive shaft is floating and only supported by the universal joint. Not constraining the shaft at three points prevents the driveshaft from flexing too much, if the structure of the segment warps under high loads. The motor has too large a diameter to fit into the segment next to the drive shaft spine (see Figure 9c), as is the case in the OT-8. However, the OT-4 drive motor has two output shafts, so that each drives one half of the now split drive shaft spine.

4. Design Details OT-8 vs. OT-4

In this section we provide details on some the more important components of the OmniTreads. We also compare design features of the OT-8 with those of the newer and more feature-rich OT-4

4.1 Tracks

The OT-8 has 40 tracks and 160 sprockets and rollers. These components make up a significant portion of the overall weight of the system. In order to minimize system weight, we sought tracks that were particularly lightweight. In addition, the tracks had to offer low drag and they had to be able to accommodate debris (especially sand) that could get trapped between the tracks and the drive sprockets.
A solution was found in the form of a slightly elastic urethane material that would stretch to accommodate debris without mechanical tensioners, yet was strong enough not to slip over the sprocket teeth under stress. After testing different tracks designs we selected the section profile shown in Figure 10a. This design is an adaptation of the rubber tracks found in the Fast Traxx remote-controlled toy race car made by Tyco. The trapezoidal extrusion on the bottom of the track fits into a groove on the sprocket, ensuring that the track stays aligned on the sprocket. For further testing we rapid-prototyped tracks based on this design using 50 through 90 durometer urethanes.
In the much smaller OT-4 we had to simplify the gearbox; the chain is run off a sprocket mounted directly on the side of the worm gear. The chain drive is therefore off-center with respect to the driveshaft and the two tracks per side are therefore not of equal width (see Figure 10b). The tracks are molded in-house from a silicon mold. That mold is made from a Stereolithographic (SLA) rapid prototype, based on a CAD model, which we also developed in-house. The grousers have twice the pitch of the track teeth to better engage features of the obstacle being scaled. Keeping the grouser pitch a function of the tooth pitch reduces the

stiffness of the track as most of the flexibility of the track comes from the thin area between the teeth. In order to increase the stability of the robot to minimize roll-overs, we made the tracks as wide as geometrically possible while still allowing the robot to fit through a 4-inch hole. The track width is especially important considering the large deflection of the center of gravity we can impose on the robot by raising three segments in the air. To meet both goals, we had to minimize the sprocket diameter, as is evident from Fig. 10b.
Discussion:
There are several disadvantages to small sprockets: 1. greater roll resistance, 2. reduced ability to transfer torque between the sprocket and the track, and greater sensitivity (i.e., failure rate) when driving over natural terrains such as deep sand and underbrush. On these natural terrains sand and twigs are ingested between the tracks and drive sprocket, forcing the tracks to overstretch. In most industrial environments and urban search and rescue operations, surfaces are man-made and the OT-4 performs very well on them.

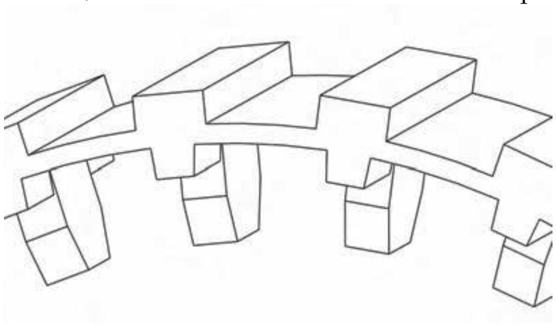
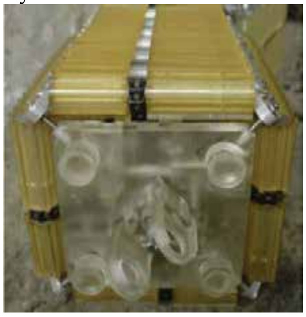

a

b

Fig. 10. a – Profile of the OT-8's urethane tracks, b – Front view of the OT-4. The extra-wide track areas add stability and reduce the risk of rollovers.

The diameter of the OT-8 track sprockets is much larger than that of the OT-4 track sprockets. Consequently, the OT-8 performed exceedingly well in deep sand and in underbrush. In order to transfer more torque, the tooth profile was kept similar to that of a timing belt, i.e., we maximized the number of engaging teeth.

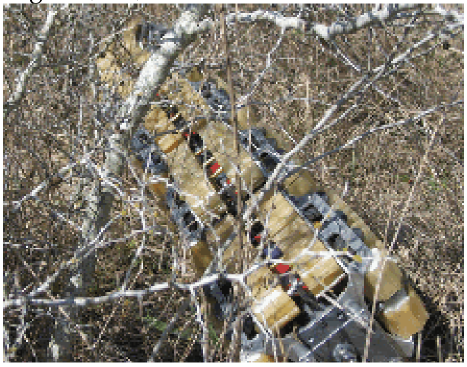
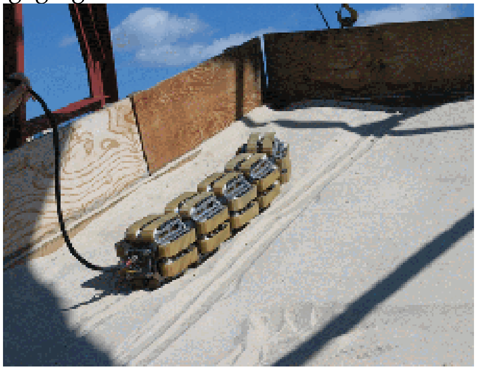
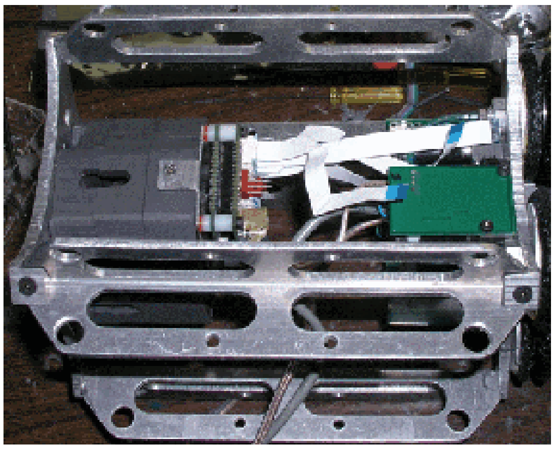

a

b

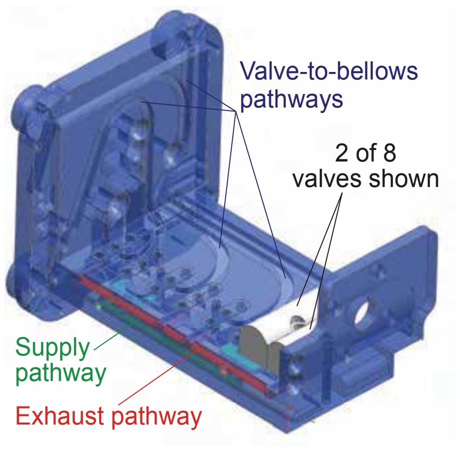
Fig. 11. The OT-8 outperformed the OT-4 on difficult natural terrains, mostly because of the OT-8 larger track sprocket diameter. a – The OT-8 literally plowed through SwRI's underbrush test area, aided, in part, by its more massive weight and greater power. b– Unaffected by sand, the OT-8 had enough power and traction to drive up a 15° inclined bed of deep sand.

4.2 Chassis

The chassis of the OT-8 consists of duralumin frame with attached gearbox machined from Delrin, as shown in Fig. 12a. Most of the components including sprockets and rollers were made in house. We spent some time to reduce the weight of metal parts and to optimize their shape. As a result we obtained easy to assemble segments with a lot of spare space inside. This space could be used for energy storage or payload.

Due to the small size of the OT-4, significant efforts had to be made to organize the internal components for space efficiency and accessibility (Borenstein et al., 2006). Cables and pneumatic lines are routed with these goals in mind. For example, the electronic circuit board on each segment has a connector on each end, with the wires coming from the neighboring segments plugging into the closer side. This design eliminated the need for wire runs all the way through the segment. Similarly, instead of using air hoses we integrated all pneumatic pathways into the chassis. This was possible thanks to SLA rapid prototyping techniques, which build the parts in layers and allows for such internal features. The chassis with integrated manifold and "etched-in" pneumatic pathways is shown in Fig. 12b. SLA rapid prototyping allowed us to create very complex, and otherwise difficult to machine structures. The SLA technique also allowed us to design parts for ease of assembly, maintenance, and space savings. However, SLA resins tend to warp with time, which is why they are normally used for prototyping only. In our early OT-4 prototypes, components that were under constant load would creep with time and would cause problems, especially in the case of the seal between the valves and the manifold. Aluminum reinforcements were therefore added to the endwalls, joints and manifold at key points where creep and deformation during load was becoming an issue. The endwalls were reinforced with a thin aluminum shell and the manifold was reinforced with a bar screwed on at both ends. The result was a much stiffer segment at a minor (2.5%) weight penalty.


a



b

Fig. 12. a – Aluminum frame of the OT-8 with gearbox, controller boards and manifolds with white flat cables visible inside, b – Manifold of the of OT-4 with two of the eight valves (white) mounted. Exhaust and supply pathways from and to the bellows are shown in red and green, respectively. This manifold is also partially the chassis of the six OT-4 actuation segments.

4.3 Joints

Between any two segments there are two concentric universal joints referred to as the "outer" and "inner" universal joint. The outer universal joint connects the two adjacent segments. It is made of two forks and a ball bearing-mounted gimbal connecting the two forks, as shown in Figure 13a. The inner universal joint (not shown) connects adjacent segments of the drive shaft spine and is concentrically located inside the gimbal. All components of the outer universal joint are made from aluminum and each fork is screwed onto the adjacent segment endwalls. Two Hall-effect angle sensors are mounted on one arm of each fork, respectively, provide position feedback for the control of the joint angles. The joint can be actuated at least 33° in any direction and up to 41° in the four principal directions (up, down and side to side). Wiring and pneumatic lines between the segments pass through four holes at the corners of the gimbal and the bases of the forks. Each joint is orientated with respect to each other in a way so as to compensate for gimbal error, the angular twisting deviation that occurs between the two ends of a universal joint as it is articulated. Without this, three fully articulated joints would lead to each progressive segment being twisted about the drive spine axis leading to instability and impeding obstacle traversal. It should be noted that the space available for the mechanical joints is extremely limited as it is shared with the bellows of the Integrated Joint Actuator (see Fig. 13b). Moreover, space is limited because we try to dimension the bellows with the largest possible diameter to increase their force capacity, as explained next.

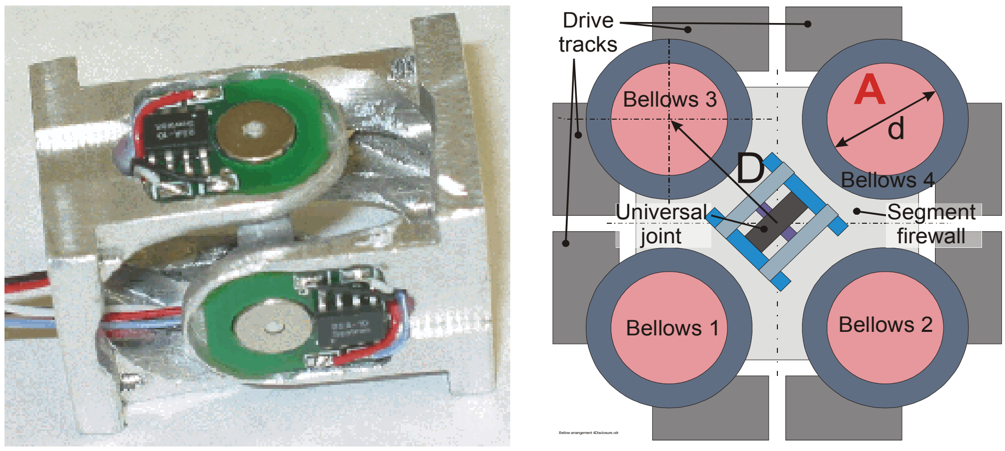


Fig. 13. Joints in the OmniTread robots: a – Outer universal joint, with Hall-effect joint angle sensor as used in OT-4. b – Cross-section of the Integrated Joint Actuator.

4.4 Pneumatic Bellows

Pneumatic bellows develop axial force according to

$$F = PA \qquad (2)$$

where P is the pressure of the compressed air and A is the area of the bellows surface that is normal to the axial direction, that is, the area of the cross section. One problem with Eq. (2) is the difficulty in determining exactly what the area A is. For example, in the bellows shown in Figure 14a, the convolutes change the diameter and thus the area of the cross section along the bellows. Of particular concern is the minimal cross section area, A_{min}, which corresponds to the inner whorl of the convolutes. For a given pressure P, the axial force that the bellows can apply is limited by the cross section area of the inner whorls, A_{min}. Yet, the volume of space that the bellows requires is determined by the diameter of its outer whorls. In the relatively large OT-8, the ratio between inner and outer diameter of the whorls (we refer to this ratio as "bellows efficiency") is fairly close to 1.0. However, in the smaller bellows of the OT-4, the bellows efficiency is much smaller than 1.0. In many conventional bellows the diameter of the inner whorl increases when inflated, thereby improving that bellows' efficiency. However, our OT-8 bellows design uses a metal ring around the inner whorls to prevent the bellows from ballooning. At the same time, these rings prevent the inner whorls from growing in diameter, thereby keeping the bellows efficiency low. To overcome this problem in the small-sized OT-4 bellows, we abandoned the metal rings altogether. Instead, we encased the OT-4 bellows in a tubular polyester mesh. To distinguish between these parts, we call the airtight, elastic part of the bellows "liner," and the outer part "mesh."

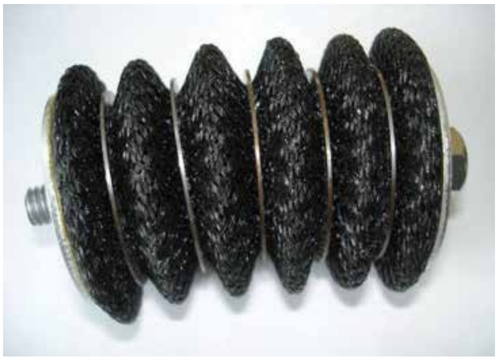
The new two-part bellows of the OT-4, shown in Figure 14b, has the significant advantage of allowing the diameter of the inner whorl to grow when pressurized, until the inner whorl is practically flush with the mesh. The result is a bellows that has an efficiency of close to 1.0, when fully pressurized.

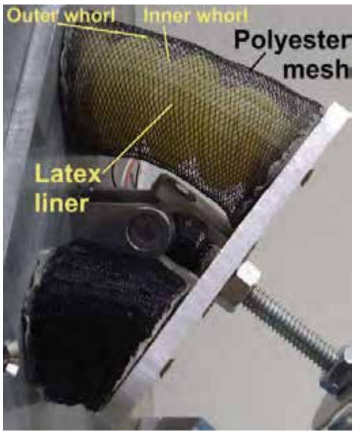
There is, however, one problem with all bellows designs: When the bellows extends beyond the natural length of the liner, the axial extension force $F = PA$ has to work against the elasticity of the liner. Similarly, when a bellows in a joint is compressed beyond a certain limit (e.g., because the bellows on the opposite site is expanding), its liner and mesh develop elastic forces that resist further compression with increasing force. Conversely, the bellows on the side of the joint that is being compressed resist further compression, thereby making it harder for the opposing bellows to expand. As a result of these effects, the moment produced by the bellows when installed inside a joint is neither constant nor depending only on the applied pressure differential. Rather, the produced moment is a non-linear function of the joint's momentary angle. For extreme joint angles, the moment produced by the joints may be reduced by as much as 50% compared to the maximal moment that's available when the joint is in its neutral position. In the OT-4, however, we dimensioned the bellows so as to be powerful enough to lift three segments even at near-maximal joint angles.

4.5 Power

In the OmniTread OT-8 prototype, electric and pneumatic energy, as well as control signals are provided through a tether – a 1 cm thick and 10 m long cable comprising six wires and a pneumatic pipe. Compressed air is supplied from an off-board compressor and distributed to the control valves from a single pipe running through the center of the robot. In the experiments described in the next section the compressor provided variable pressure from 85 to 95 psi but the control system limited the maximum pressure in the bellows to 80 psi.

Of course, a tether is highly undesirable for most inspection and surveillance tasks. That's why, when we designed the OT-4, we incorporated all energy resources onboard, and provided it with wireless communication. The OT-4 has Li-Pol batteries in Segments #3 and #5. Pneumatic energy, that is, compressed air, is produced onboard by two miniature air compressors, one each in Segments #2 and #6. Fully charged Li-Pol batteries allow the OT-4 to drive for 75 minutes on flat terrain. On very difficult terrain the run time is shorter.


a



b

Fig. 14. a – UM-developed rubber bellows used in the OT-8.
b – OT-4 bellows comprising a liner and a mesh. We chose yellow latex liner material for this photograph because it contrasts better with the black mesh. However the actual OT-4 bellows have neoprene (black) liners.

A unique means for preserving electric power are the OT-4's micro-clutches, which allow it to engage or disengage each individual track from the main drive shaft. Disengagement of tracks not in contact with the ground results in a significant saving of electric energy. For instance, on flat ground only three tracks need to be engaged: the ones on the bottom of Segments #2, #3 and #6, while the other segments are slightly lifted off the ground. This configuration provides stable, 3-point contact and steering is accomplished by the joint between Segments #2 and #3.

5. Control System

5.1 The Simplified Proportional Position and Stiffness Controller

In our paper (Granosik & Borenstein, 2004) we proposed a control system called "Proportional Position and Stiffness" (PPS) controller. The PPE system is designed to do what its name implies: it allows for the simultaneous and proportional control of position and stiffness of pneumatic actuators. The PPS controller is further optimized for use in mobile robots, where on-board compressed air is a valuable resource. To this end, the PPS employs a uniquely designed system of valves that assures that compressed air is consumed only during commanded changes of pressure or stiffness, but not while an actuator is held at a constant pressure and stiffness.

However, the PPS controller as described in (Granosik & Borenstein, 2004) is based on an approximated model of cylinders and requires the real-time measurement of certain system parameters. For example, the polar moment of inertia of masses that are being moved by the joint must be known at all times, as well as the torque needed to move the joint. In complex environments where the serpentine robot may be laying on any side, additional sensors would be needed to measure these parameters.

Because of these difficulties we simplified the control system so that these sensors are not needed, while maintaining acceptable performance. In order to distinguish the simplified control system from the proper control system, we call it "Simplified Proportional Position and Stiffness" (SPPS) controller. The SPPS controller uses a PID position controller with a stiffness control subsystem, as shown in Figure 15.

The task of the control system is to control the position of a joint, as well as its stiffness. The controlled parameters are the pressures p_A and p_B in the bellows-pair that actuates the joint. In order to control p_A and p_B, the PID controller generates the control signal u as input for the valve control subsystem. This subsystem realizes the stiffness control and air flow minimization by activating the four pneumatic valves according to the flow chart in Figure 16. In every control cycle only one of the four valves is active, i.e. generates airflow to or from one of the bellows.
The SPPS assigns higher priority to stiffness when conflicts between position control and stiffness control arise. However, the SPPS control system cannot change the stiffness of a joint in steady state because the valve control subsystem is activated by position errors only. As our experiments showed, however, this limitation can easily be accommodated by the teleoperators or by an onboard computer, in future, more advanced OmniTread models.

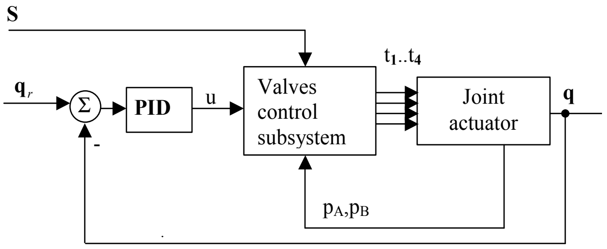


Fig. 15. Block diagram of the Simplified Proportional Position and Stiffness (SPPS) system with zero air consumption at steady state.

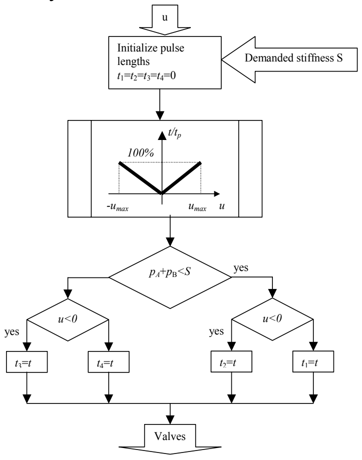


Fig. 16. Flow chart for the valve control subsystem.

5.2 Control System Hardware

The four joints of the OT-8 prototype are actuated by a total of 16 pneumatic bellows. With this design the OT-8 has 25 individually and proportionally controllable parameters, namely: 4 × 2 DOF position, 16 × pressure, and 1 × speed forward/ backward.
In order to control the OmniTread we developed a microprocessor-based distributed control system consi sting of five local controllers – one for each IJA and one for the motor. Each local controller is based on a 16-bit Motorola MC9S12DP256B micro-controller and all five controllers communicate with a master PC via CAN bus. A schematic diagram of the control system is shown in Figure 17.

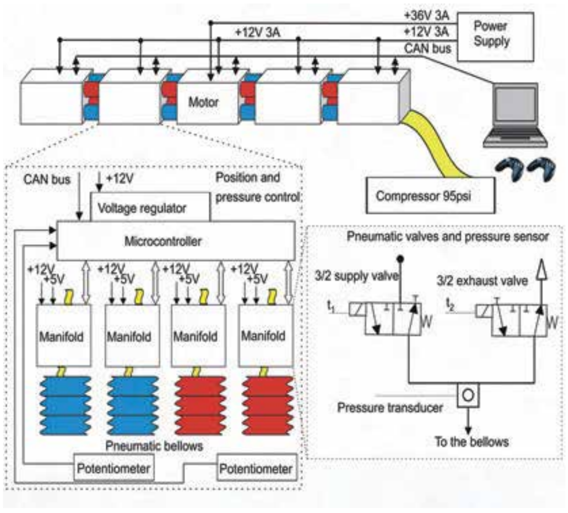
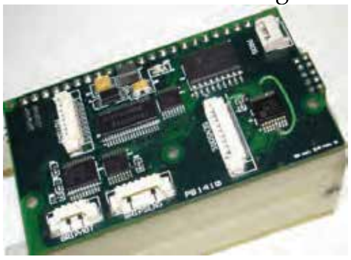


Fig.17. Schematic diagram of the OmniTread control system, shown for the OT-8.

We used the same structure of control system in case of OT-4. However, as this robot has six joints the number of controllable parameters is even larger that in the OT-8. To reduce the number of controllable parameters we decided to control the stiffness of each IJA (i.e., a group of four bellows) instead of the stiffness of each individual bellows. This reduces the number of controlled parameters to 6 joints × 2 DOF + 6 × stiffness + 1 × speed = 19 parameters. It was also necessary to fit the controller boards into the very small space of the OT-4 segments. Moreover, additional functions in the OT-4, such as micro-clutches, require additional circuitry.

In order to accommodate all of this circuitry, we split the functions into one Main Controller board and another Auxiliary Controller board, as shown in Fig. 18. Each Main Controller board manages the position and stiffness control for its adjacent 2-DOF joint. The microcontroller can receive new position and stiffness commands and return feedback data (two positions and four pressures) every 10 ms. Each microcontroller sends digital PWM signals to eight on-off pneumatic valves (two for each bellows) to implement the Simplified Proportional Position and Stiffness (SPPS) controller described in detail in Section 5.1. Each microcontroller also reads positions from two Hall-effect angle sensors and pressures from four pressure transducers.


a

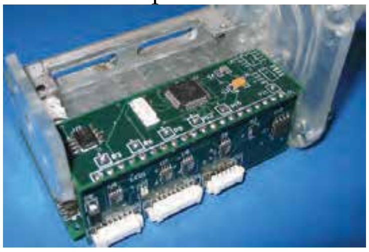

b

Fig. 18. Electronic control circuit boards in the OT-4.

a Main Joint Controller board. One each of these double-sided boards is installed in each to of the six Actuation Segments. These boards perform (i) communication via the CAN bus, (ii) PWM control of the eight valves per segment, and (iii) control of the micro-clutches, as well as (iv) measurements of joint angles and bellows pressures.

b Auxiliary Controller board. One each of these angled PCBs is installed in each of the six Actuation Segments.These boards provide added sensing and control capability, including: System pneumatic pressure sensor, PWM compressor control switch, four (4) force sensor inputs (for future work), PWM servo control output. Not all of the outputs are needed in all of the segments.

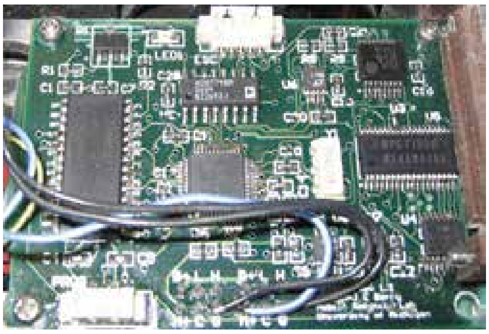


Fig. 19. Drive Motor Controller Board. This board generates the PWM signals for the off-the-shelf digital power amplifier for the OT-4 motor.

6. Features Found only in the OT-4

Up to this point we discussed mostly features and properties that are common to the OT-8 and the OT-4. However, the OT-4 has several unique features that are not found in the OT-8. We discuss the most salient of these features in this section.

6.1 Completely Tetherless Operation

The OT-8 requires three resources to be supplied to the robot through a tether: Electric power, compressed air at 80 psi, and control signals. In order to make the OT-4 entirely tetherless, these resources had to be supplied onboard. We will briefly discuss here how these onboard supplies were implemented.

6.1.1 Electric power

The OT-4 has two electric power circuits: a motor power circuit and a control power circuit.

a. The motor power circuit powers the drive motor and the two onboard compressors. This power is supplied by two 7.4 V, 2,000 mAh Li-Pol batteries, one each stored in Segments #3 and #5 (Fig. 20). The two batteries are connected in series to provide 14.8 V and their total energy storage capacity is 29.6 Wh. These batteries take up a volume of 84 cm^3 and weigh a total of 160 g.
b. The control power circuit powers the electronics control boards and pneumatic valves, as well as the wireless communication system. This power is supplied by two 7.4 V, 730 mAh Li-Pol batteries, one in Segments #3 and one in Segment #5. The two batteries are connected in parallel to provide 1,460 mAh at 7.4 V and their total energy storage capacity is 10.8 Wh. These batteries take up a volume of 34 cm^3 and weigh a total of 76 g.

In the endurance test (driving as far and long as possible on one charge on flat concrete floor) the motor power and the control power batteries lasted roughly the same time (75 minutes). On extremely difficult obstacles, where joints are actuated a lot and all tracks are engaged, the motor battery can be depleted in as little as 25 minutes.

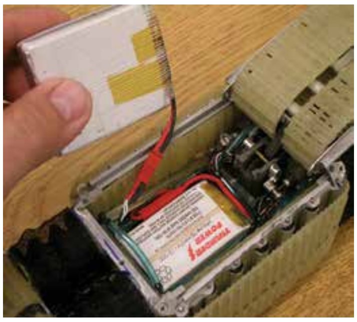


Fig. 20. The 730 mAh Li-Pol battery is in its place in Segment #5. The remaining space will be completely filled by the 2,000 mAh Li-Pol battery in the engineer's hand.

6.1.2 Pneumatic power

Pneumatic power is supplied by two off-the-shelf Hargraves CTS mini-compressors, shown in Fig. 21. In order to increase the flow rate we added another compressor head to the motor so that in one revolution of the crankshaft, two pistons go through a compression cycle. Since the stock motor was therefore under a larger load the flowrate wasn't doubled but it was increased significantly. We then further modified these compressors by replacing the stock motor by a more powerful one, the Faulhaber Model 2232 012 SR. The Faulhaber motor is coreless, slightly larger (22 mm as opposed to 20 mm) and has a higher power rating. Because of that higher rating, it draws only 1/3 the current of the stock motor, which was somewhat overloaded when running two heads.

In this configuration, the compressor provides about 25 psi (less when flow rates are high). This maximal pressure is sufficient for most ordinary tasks with the OT-4, since its bellows were specifically designed for a much lower operating pressure than that of the OT-8. However, for extreme task, such as the vertical climb in large-diameter pipes and other tasks with vertical motion requirements, a higher pressure would is desirable. That's because higher pressures translate into proportionally greater joint actuation torques. To achieve this higher pressure we connected the two heads in series, increasing the effective output pressure of one compressor up to 50 psi. Implementing the pneumatic diagram of Fig. 22, it is possible to switch the two compressor heads between series and parallel mode, by means of a single solenoid valve and a check-valve. In principal, switching can be done anytime during operation, without stopping the compressors, but at the time of writing this chapter we have not yet found a small enough solenoid valve with high enough flow rate to make the design beneficial.

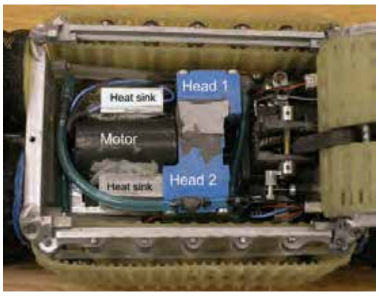


Fig. 21. (above) Modified dual-head air compressor installed in Segments #2 and #6.

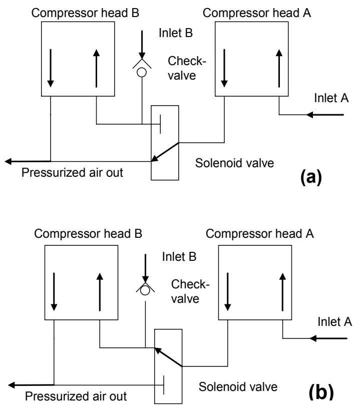


Fig. 22. (above) The dual-pressure compressor system. The compressor heads can be switched between (a) Parallel Mode and (b) Series Mode by switching the state of the solenoid valve.

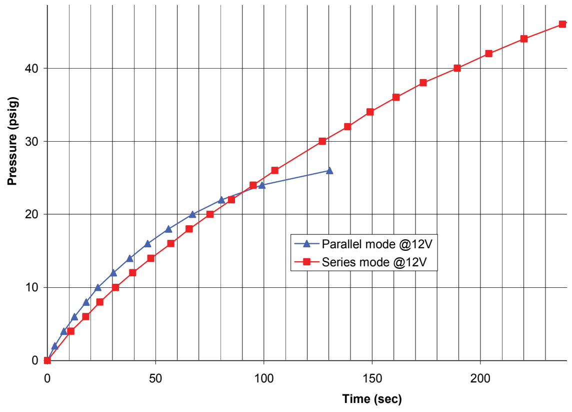


Fig. 23. Plot of pressure versus time required to fill a fixed test volume (2.1 liter) with air at different pressures. Comparison of dual-pressure compressor working in series (blue) and parallel (red) mode. This chart is for one compressor, although the OT-4 uses two compressors.

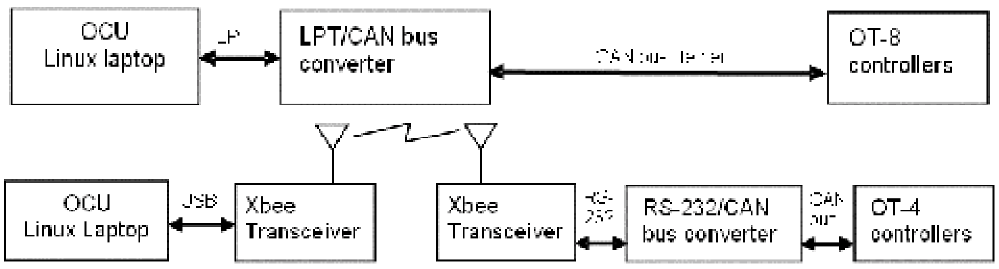


Fig. 24. Control communication system components. a – tethered system for OT-8.b – wireless system for OT-4.

We measured the output (pressure and flow rate) of both compressor modes by connecting the outlet of one of the two compressors to a 2.1-liter container and timing how quickly it built pressure. The results are shown in Fig. 23. As can be seen, low pressures up to 15 psi can be built faster in Parallel Mode. Pressures above ~15 psi build up faster in series mode.

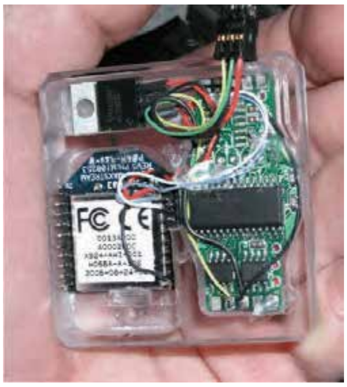


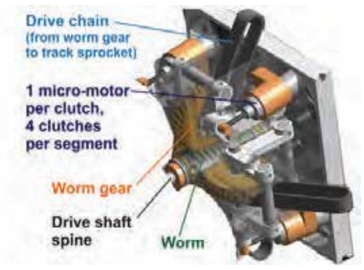
Fig. 25. Onboard components of the OT-4's wireless control communication system.

6.1.3 Wireless Control Communication System

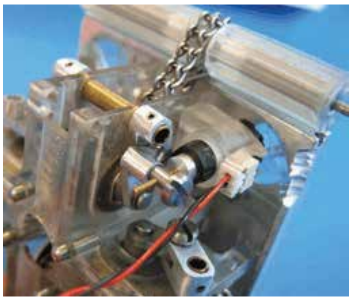
In the OT-8 the communication of control signals from the joysticks (via a laptop) to the robot and sensor signals from the robot to the off-board laptop were sent through the tether, as shown in Fig. 24a. In the OT-4, we implemented the wireless communication system of Fig 24b.
Our solution involved removal of the housing and other components from a Lawicel CAN-to-RS-232 converter to reduce it's volume, and wiring it to a Maxstream Xbee transceiver. Despite the complexity of the multiple-conversions system, we managed to integrate the components into the OT-4's tail segment such that most of the payload space in that segment remained available. Fig. 25 shows the on-board components of the wireless communication system.
The range of the system is approximately 20 m through two walls with no apparent problems. The CAN message throughput of the wireless system is slightly lower than that of the OT-8's tethered system despite similar transmission baud rates (115.2kB vs. 125kB). This is because the messages are transmitted in the wireless system as ASCII strings rather than as binary ones.

6.2 Electrically actuated micro-clutches

One reason for then OT-4's impressive motor battery run time (75 minutes on benign terrain) is a unique feature in the OT-4: all 28 of its tracks can be engaged or disengaged from the drive train individually. To motivate the utility of the clutches, let's consider some figures. A torque of T_f = 0.09 Nm is needed to drive a freely spinning track, that is, a track that is not engaged with any environmental feature. At the other extreme, the largest possible legitimate torque that a track may have to transfer is needed during vertical pipe climbs. During such climbs, one track of the center segment is pressed against the inside wall of the pipe and has to support half the robot's weight. Under this condition the torque required to turn that track is T_m = 0.21 Nm. Comparing these two extreme torque requirements shows a ratio of $q = T_m/T_f = 2.3$. The significance of this ratio is that driving 2.3 tracks at the lightest possible load (i.e., spinning freely) requires the same amount of torque a driving one track under the largest possible load condition. Since torque is roughly proportional to power consumption, we conclude that idling 2.3 tracks consumes as much power as driving half the robot's weight vertically. It is thus obvious that *not* driving an idle track will save a substantial amount of onboard electric power.



a



b

Fig. 26. Gear box and micro-clutches. a – CAD drawing, b – photograph

In practice we implemented the micro-clutches as shown in Fig. 26. To disengage a track, a micro motor moves one link of a four-bar mechanism so that the worm gear is lifted off the worm. Micro-switches (not shown here) stop the micro-motor in two stable, self-locking positions. These positions correspond to the worm gear being fully engaged or disengaged from the worm.

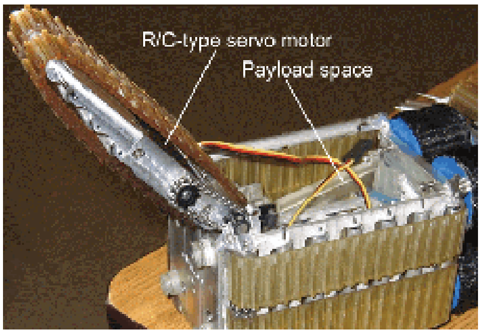


Fig. 27. Flipper track during deployment to its fully extended position.

6.3 Flipper tracks

We equipped the OT-4 with two so-called "flipper tracks." These tracks, located in the lead and tail segments, can be "flipped out" or "flipped in" to extend the reach of the OT-4. The extended reach is useful in two maneuvers: (a) to cross gaps and (b) to reach up and over high obstacles. The flipper track uses a small servo embedded in the track tray to extend the flipper 180° or to retract it. An additional locking actuator locks the track in either position. The servo is slightly wider than the height of the track, resulting in the outward bulge of the track, apparent in Fig. 27. Yet, the bulge does not interfere with the robot's ability to pass through a 10-cm hole. The new design functioned very well when we used it to overcome the knife-edge hole obstacle (see Fig. 6, back in Section 3.3), as well as in other tests, not documented here.

7. Experimental Results

In order to assess the performance of our robots under realistic and objective conditions, the OmniTreads (OT-8 and OT-4) were tested at the Small Robotic Vehicle Test Bed at the Southwest Research Institute (SwRI) in San Antonio, Texas. The OT-8 robot was tested in 2004, while the OT-4 was tested in 2006.

During these tests the OmniTread OT-8 was continuously controlled by two operators who had audio and visual contact with the robot, allowing them to monitor the robot's behavior at all times. The OT-4 required three trained operators to run. The SwRI test facility is well designed and allows for the objective assessment of vastly different small robot designs. Among the tests were tasks such as climbing over high steps, ascending through the inside of pipes, traversing wide gaps, and many more. A detailed description of all tests and their results would far exceed the scope of this chapter. Rather, we refer the reader to the many photographs in this chapter and point out that in each of the depicted scenarios, the robot successfully completed the run or passing the obstacle. The same is true for the photographs in Figure 28, which shows some more test environments and obstacles (some from SwRI, some from our own lab).

In addition, Table 2 provides some technical specifications and comments on performance. Lastly, video clips are very particularly helpful in conveying a sense of the capabilities of serpentine robots. We therefore refer the reader to our large video library on the web at http://www.engin.umich.edu/research/mrl/OmniTread_Video.html.

Parameter	OT-8	OT-4
Specifications		
Smallest hole robot can pass through	20 cm (8 inches) diameter	10 cm (4 inches) diameter
Dimensions (length, width, height) Number of segments Joint segment length Motor segment length Joint length	127×18.6×18.6 cm 5 20 cm 20 cm 6.8 cm	94×8.2×8.2 cm 7 10.3 cm 10.9 cm 3.6 cm
Weight	13.6 kg	(Incl. batteries & flippers) 4.0 kg
Power	Off-board lead-acid batteries and air compressor (80 psi). Both resources brought to robot via tether	On-board: Li-pol batteries (43 Wh), 2 air compressors (45 psi) On-board resources sufficient for 75 minutes driving on smooth terrain
Control	2 operators, 2 gamepads, off-board laptop	3 operators, 3 gamepads, off-board laptop
Controls signals	CAN bus, via tether	Wireless serial link, no tether

Special features		
Common features to both OT-8 and OT-4	Extra-wide tracks on all four sides of robot	
	Pneumatic joint actuation with position & stiffness control	
	Single drive motor, drive shaft spine	
	Requires tether	Completely untethered
	No micro-clutches needed, since electric power "free" with tether	Electrically actuated micro-clutches to engage/disengage each track
		Flipper tracks extend reach of lead & tail segment by 8 cm
	Supply pressure variable since compressor off-board	On-the-fly switchable maximal supply pressures: 30 & 45 psi
	Very strong components, since not optimized for weight & space	Mech. overload protection for each branch of drive train by shear pins
Performance		
Maximal velocity	10 cm/s	15 cm/s
Minimum turning radius (inside)	53 cm	16 cm
Performance on rocks or rubble	Excellent. Never failed a test.	Excellent. Never failed a test.
Performance on stairs	Poor. Can climb only stairs with small slopes, long treads	Excellent. Successfully climbed different combinations of rise/tread, up to 40°
Performance in deep sand	Excellent, able to climb 15° slope can drive indefinitely	Very poor. Cannot climb any slope. Stalled after 5 meters
Performance in underbrush	Excellent, plows through, can drive indefinitely	Poor. Ingests twigs, grass, stalled after 12 meters
Climbing in PCV pipe, diameters Maximal inclination of pipe	30 cm 22°	10 cm, 15 cm, 20 cm 90° (vertical)
Maximal height of vertical wall climbed	46 cm (2.5× own height)	40 cm (4.9× own height) Higher with flipper (not tested)
Width of largest gap crossed	66 cm (52% of own length)	49 cm (52% of own length)
Enter hole in wall, smallest diameter Height of center of hole above ground	20 cm Not tested	10 cm 42 cm

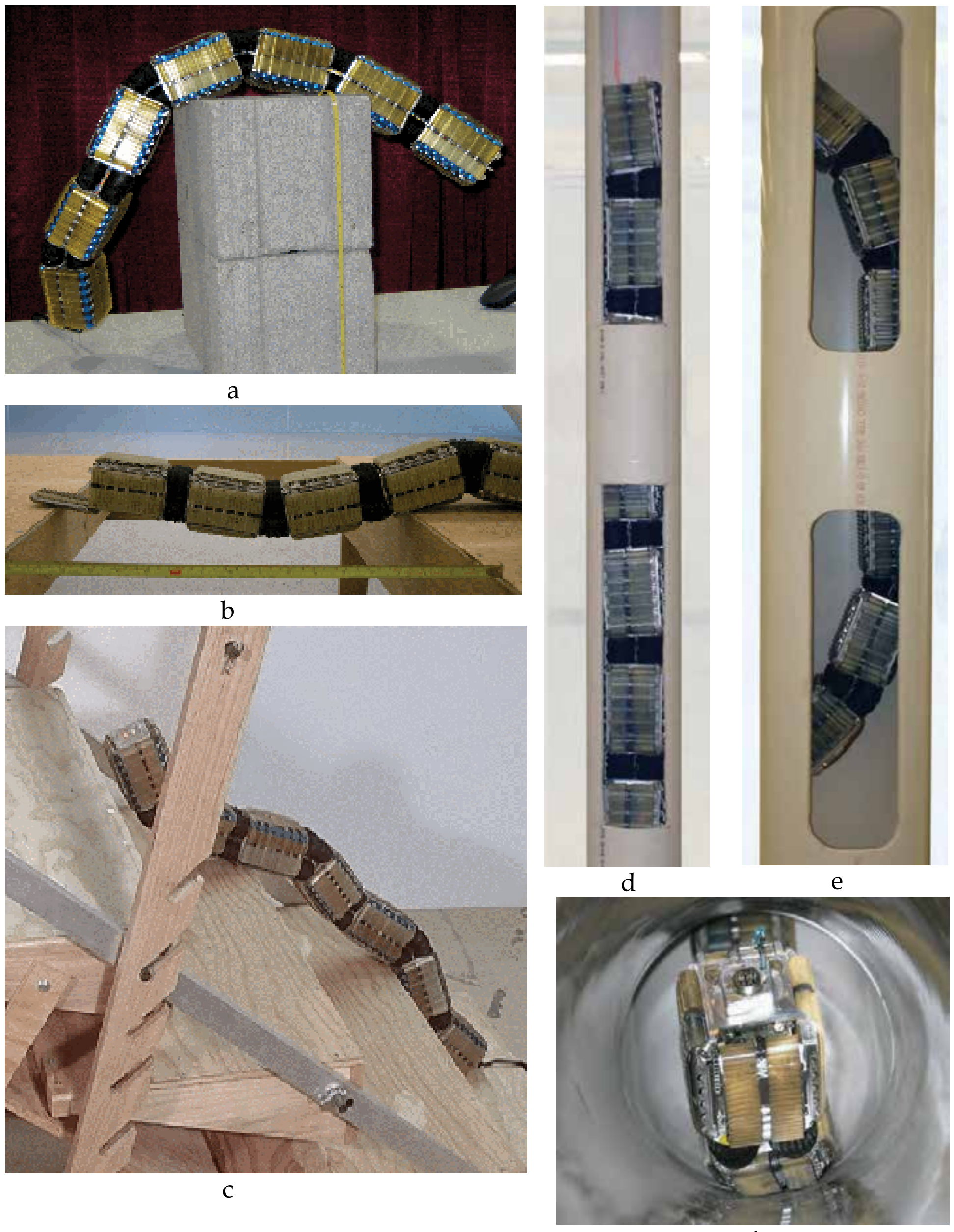


Fig. 28. OmniTread testing in different challenging environments:
a – OT-4 climbing up and over a 40-cm (15.5”) high wall (5× the robot’s heigth).
b – OT-4 traversing a gap 49 cm (19.25”) wide. This is 52% of its length.
c – OT-4 climbing up inside a 4-inch diameter PVC pipe. Speed: 8 cm/s.
d – OT-4 climbing up inside a 8-inch diameter PVC pipe. Speed: 6 cm/s.
e – OT-4 climbing up steep stairs sloped 40°
f – OT-4 inside an 8-inch diameter pipe, halfway through a 90° elbow

8. Conclusion

This chapter introduced the Omnis family of serpentine robots. Serpentine robots have the potential to provide hitherto unattainable capabilities, such as climbing over high steps, travel inside horizontal or even vertically pipes, or traversing wide gaps. While individual tasks of this nature have been tackled in the past by special-purpose mobile robots (e.g., pipe crawlers), it appears that only serpentine robots may be able to perform a large variety of difficult tasks.

We started our project with the design of the legged OmniPede. Technical evolution then let to the design of the OmniTread OT-8, and finally to our most advanced model, the OT-4. We believe that the OmniTread robots have a particularly high potential to become truly practical.

Notable in the OmniTread are several innovative features as summarized here:

- Pneumatically actuated 2-DOF joints – The 2 DOF joints of the OmniTread are actuated pneumatically. Pneumatic actuation provides natural compliance with the terrain for optimal traction and shock absorption, as well as a very high force-to-weight ratio.
- Bellows used as pneumatic actuators – Bellows are ideal for serpentine robots because they fit naturally into the space occupied by the joints. This minimizes the need for space, especially space not covered by propulsion elements. In addition, bellows can expand to four times their compressed length. (US Patent #6,870,343).
- Proportional Position and Stiffness Control system – The pneumatic control system, developed at our lab especially for serpentine robots, allows simultaneous, proportional control over position and stiffness of each individual bellows. In addition, the system uses compressed air only when changing the position or the stiffness of a bellows. Thus, during long stretches of straight travel no compressed air is used at all. (US Patent #6,870,343).
- Maximal coverage of robot surface with moving tracks – In this chapter we identified and formalized the need for maximizing the so-called "Propulsion Ratio," P_r. We implemented this design doctrine by covering all sides of all segments with extra-wide, moving tracks. The cost for this approach is additional complexity. The cost for not using this approach is mission failure when the robot gets stuck on troublesome terrain (US Patent #6,774,597).
- Single drive motor for all segments and drive shaft spine – Our analysis shows that a single drive motor is more energy, weight, and space-efficient than multiple motor configurations (i.e., one motor in each segment). Motor power is transferred to the segments via a drive shaft spine that runs the length of the robot. Within each segment the drive shaft spine is a rigid axle, connected to the axle in neighboring segments via a universal joint (not a flexible shaft). (US Patent #6,512,345).

We are currently working on the higher level control system for Serpentine robots. A problem with high-degree-of-freedom (HDOF) serpentine robots is that they often require more than one human operator. In the case of the OT-4, three operators simultaneously control the robot using six individual joysticks as well as auxiliary instrumentation.

In order to reduce the number of operators needed, we developed a "Haptic Operator Console" (HOC), which we call the "Joysnake" (as in joy-stick.) The premise of the Joysnake is that the fastest and most intuitive method for a human operator to command a pose for a High Degree of Freedom robotic mechanism is to shape an adjustable replica of the mechanism into the desired pose (Baker & Borenstein, 2006). In most simple to moderately difficult task the Joysnake works sufficiently well to replace the three operators by just one. However, in very complex task the three operators perform better.

9. Acknowledgements

This work was funded by the U.S. Dept. of Energy under Award No. DE FG52 2004 NA25587 as well as by the Intelligence Technology Innovation Center under Grants #F013596 and F013019.
The authors are grateful to the OmniTread OT-4 development team for their full-time efforts: Jack Phan, Russ Miller, Jim Berry, Justin Tesmer, Adam Borrell, Oke Liimatta, Hung Nguyen, and John Baker. We are also grateful to David Thomas, Jenny Weinberg and the many machinists who worked on the OT-4 part-time.
We are also grateful to Dr. William Hutchison who developed the "7G" computerized learning program. This program produced the sophisticated and complex motion strategy used for traversing the steep staircase in Fig. 7 and Fig. 28c.

10. References

Baker, J. & Borenstein, J. (2006). The Joysnake A Haptic Operator Console for High-Degree-of-Freedom Robots. *2006 International Joint Topical Meeting: "Sharing Solutions for Emergencies and Hazardous Environments,"* February 12-15, Salt Lake City, Utah, USA

Blitch, J.G. (2003). Adaptive Mobility for Rescue Robots, *Proc. of SPIE. Sensors, and Command, Control, Communications, and Intelligence (C3I) Technologies for Homeland Defense and Law Enforcement II*, vol. 5071, pp. 315–321.

Brown, H.B., Jr.; Vande Weghe, J.M.; Bererton, C.A.; Khosla, P.K. (2002). Millibot trains for enhanced mobility, *IEEE/ASME Transactions on Mechatronics*, Vol. 7, Issue 4, pp. 452-461.

Borenstein, J.; Hansen, M.G. & Nguyen, H. (2006). The OmniTread OT-4 Serpentine Robot for Emergencies and Hazardous Environments, *2006 Int. Joint Topical Meeting: "Sharing Solutions for Emergencies and Hazardous Environments,"* February 2006, Salt Lake City, Utah, USA.

Granosik, G. & Borenstein, J. (2004) Minimizing Air Consumption of Pneumatic Actuators in Mobile Robots, *Proc. IEEE Int. Conference on Robotics and Automation*, New Orleans, pp. 3634-3639.

Granosik, G. & Borenstein, J. (2005). Integrated Joint Actuator for Serpentine Robots, *IEEE/ASME Transactions on Mechatronics*, Vol. 10, pp. 473-481, October 2005.

Hirose, S. (1993). *Biologically Inspired Robots (Snake-like Locomotor and Manipulator)*, Oxford University Press.

Hirose, S. & Morishima, A. (1990). Design and Control of a Mobile Robot With an Articulated Body, *The International Journal of Robotics Research*, Vol. 9, No. 2, pp. 99-113.

Hirose, S.; Morishima, A.; Tukagosi S.; Tsumaki T.; Monobe, H. (1991). Design of Practical Snake Vehicle: Articulated Body Mobile Robot KR-II, *Fifth Int. Conference on Advanced Robotics, 'Robots in Unstructured Environments'*, Vol. 1, pp 833 -838.

Kamegawa, T.; Yamasaki, T,; Igarashi, H.; Matsuno, F. (2004). Development of the Snake-like Rescue Robot KOHGA, *Proc. IEEE Int. Conference on Robotics and Automation*, New Orleans, LA, April, pp. 5081-5086.

Kimura, H. & Hirose, S. (2002). Development of Genbu: Active wheel passive joint articulated mobile robot, *Proc. IEEE/RSJ Int. Conference on Intelligent Robots and System*, Vol.1,pp. 823 -828.

Klaassen, B. & Paap, K.L. (1999). GMD-SNAKE2: A Snake-Like Robot Driven by Wheels and a Method for Motion Control, *Proc. IEEE Int. Conference on Robotics and Automation*, Detroit, MI, May 10-15, pp. 3014-3019.

Long, G.; Anderson, J.; Borenstein, J. (2002). The OmniPede: A New Approach to Obstacle Traversion. *Proc. IEEE Int. Conf. on Robotics and Automation*, USA, pp. 714-719.

Mori, M. & Hirose, S. (2002). Three-dimensional serpentine motion and lateral rolling by active cord mechanism ACM-R3, *IEEE/RSJ International Conference on Intelligent Robots and System*, pp. 829-834 vol.1.

Ohno, H. & Hirose, S. (2000). Study on slime robot (proposal of slime robot and design of slim slime robot), *Proc. IEEE/RSJ Int. Conference on Intelligent Robots and Systems*, (IROS 2000), pp 2218-2223, Vol 3.

Osuka, K. & Kitajima, H. (2003). Development of Mobile Inspection Robot for Rescue Activities: MOIRA, *Proc. IEEE/RSJ Int. Conference on Intelligent Robots and Systems*, Las Vegas, Nevada, pp. 3373-3377.

Paap, K.L.; Christaller, T.; Kirchner, F. (2000). A robot snake to inspect broken buildings, *Proc. IEEE/RSJ Int. Conference on Intelligent Robots and Systems*, pp. 2079-2082.

Schempf H., Mutschler, E., Goltsberg, V. , Skoptsov, G., Gavaert, Vradis, G. (2003). Explorer: Untethered Real-time Gas Main Assessment Robot System, *Proc. of Int. Workshop on Advances in Service Robotics, ASER'03*, Bardolino, Italy.

Scholl, K.U.; Kepplin, V.; Berns, K.; Dillmann, R. (2000). Controlling a multi-joint robot for autonomous sewer inspection, *Proc. IEEE Int. Conference on Robotics and Automation, ICRA '00*, vol.2, pp. 1701-1706.

Streich, H. & Adria, O. (2004). Software approach for the autonomous inspection robot MAKRO. *Proc. IEEE Int. Conference on Robotics and Automation*, New Orleans, LA, USA, pp. 3411-3416.

Takayama, T. & Hirose, S. (2000). Development of Souryu-I connected crawler vehicle for inspection of narrow and winding space, *26th Annual Conference of the IEEE Industrial Electronics Society, IECON 2000*, vol.1, pp 143-148.

Using Parallel Platforms as Climbing Robots

Oscar Reinoso[2], Rafael Aracil[1], Roque Saltarén[1]
[2]Universidad Miguel Hernández de Elche
[1] Universidad Politécnica de Madrid
Spain

1. Introduction

Primates are the living beings with a greater capacity of manipulation. This skill derives from they have two legs equipped with elements adapted to the manipulation and grasping. The simultaneous use of both legs and arms confer to these living beings their special features to manipulate and manage different objects. As many technical developments inspired when nature laws are observed, parallel robots are conceived in a similar way. In this way, the mechanical structure of a parallel robot is composed by a closed chain in which the end effector is linked to the basis at least by two independent kinematic chains.
This definition can be in conflict with the developments about coordinated robots that also constitute closed kinematic chains. Parallel robots simplify these chains in such a way that every one has only one actuator. So, the complexity of the mechanism can be reduced and it is possible to make good use of the energy provided by the actuators to obtain a higher payload capacity or to increase the speed of movement of the end effector.
The first theoretical works on mechanical parallel structures appear long time ago, even before the notion of robot. In this way the first parallel mechanism was patented in 1931 (US Patent Nº 1789680) and was designed by James E. Gwinnett (Gwinnett 1931). In 1940 Willard Pollard presented a robot with 5 degrees of freedom dedicated to painting tasks. The robot was composed of three legs of two links each one. The three actuators of the base drive the position of the tool.
However, other more significant parallel mechanisms have been achieved from then. In this way, in 1947 Eric Gough designed the most popular parallel platform. Nowadays numerous platforms can be found with the name of MAST (Multi-Axis Simulation Table). In 1965, Mr. Stewart (Stewart, 1965) described a movement platform of 6 degrees of freedom (dof) designed to use as a flight simulator. On the contrary to the general belief the Stewart mechanism is different to the previously presented by Gough. The work presented by Stewart had and have a great influence in the academic world, and it is considered one of the first works of analysis of parallel structures.
At the same time, Klaus Cappel carried out in the Franklin Institute Research Laboratory a lot of researches with parallel platforms of 6 degrees of freedom. In 1967 Cappel patented a motion simulator based on a hexapod (Cappel, 1967). Later, in 1978 Hunt (Hunt 1978) suggested that the mechanisms of the flight simulators can be used as parallel robots. Also Hunt pointed out that parallel robots required a more detailed study in the context of robotic applications due to the advantages of accuracy and rigidity of these platforms.

Besides an excellent relation payload/dead weight, parallel robots have other interesting features. In 1979, McCallion and Pham (McCallion and Pham, 1979) suggested to use the Stewart platform as a parallel robot in an assembly robotics cell, basically due to the end-effector is much less sensitive than serial robots. The rigidity assures that the distortions in the actuators are minute and produces a high accuracy in the pose of the robot.
In this paper we present some parallel robots designed and developed to climb along several structures as pipes, tubular or metallic structures. The parallel robots proposed can climb on inside or outside tubular structures or palm trunks. Also a parallel robot with light modifications has been developed to climb on metallic structures of bridges or dome of buildings. Therefore, the purpose of this paper consists of showing the promising applications that are possible to achieve using parallel robots as climbing robots.
The remainder of the paper is structured as follows. In the next section the main features of parallel robots will be presented and discussed. Different possible configurations and morphologies of the parallel robots will be suggested. In the third section the kinematics and dynamics analysis of the parallel robots are analysed. In a following section some prototypes of parallel climbing robots will be shown. The main features of these robots will be analyzed and discussed. In section five, other parallel robots developed taking into account the initial structure of the first prototype of parallel climbing robot developed will be presented. Finally, some conclusions and future works will be proposed.

2. Main Features of Parallel Robots

Without a doubt, the present supremacy of the serial robots as opposed to parallel robots is patent taken into account their sales volume. However, parallel robots present a constant increasing tendency since some years ago. Nowadays, parallel robots are used in multiple applications due to present many advantages that make them especially suitable to resolve many problems in which serial robots have some important limitations.
With the purpose of comparing both platforms we can express the following advantages of the parallel robots:

- Power actuators are directly connected to the base of the robot with the end effector. So, power actuators serve as structural elements conferring high load capacity even more than its own weight. This way, these platforms have a high proportional ratio of its payload and deadweight providing a high energetic eficiency.
- Parallel structures are platforms capable to reach high velocities and develop big forces with a very important advantage: the low cost of manufacturing (Lazard, 1992).
- Parallel platforms are mechanically less complex than serial robots.

However, also parallel robots present some features that, depending on the application can be considered as disadvantages:

- Kinematics of parallel robots is more complicated. In some occasions redundant sensors are necessary to control the system.
- Working space is difficult to calculate due to the position and orientation of the end effector are extremely coupled. Several works have been reported about the position and orientation workspace of these platforms (Huan et al., 1999 & Almonacid et al. 2001).

- Possible singularities are very complex to analyze. Singularities should be analyzed specifically for every topology of parallel robot.
- A general dynamic model for the parallel robot is difficult to obtain in opposite of linear robot. For these reason, parallel robots are controlled nowadays in a decoupling manner.

2.1 Possible configurations of parallel robots

Many possible configurations of parallel robots can be designed. The high volume of possible combinations of kinematic chains, joints, restrictions of the joint movements, etc., make actually impossible to systematize all possible structures of parallel robots.
Merlet (Merlet, 1997) proposed a first sorting taking into account the movements that parallel robots can to accomplish. Two groups can be differentiated: planar robots and 3D robots. The movements of planar robots are reduced on a plane. Therefore they can have 2 or 3 degrees of freedom (a linear movement in the plane and a rotation movement along one axis perpendicular to the plane). Figure 1(a) shows some of possible configurations of planar parallel robots. However, 3D parallel robots can work in all 3D space. Figure 1(b) shows some of them with different degrees of freedom. In all of them universal (U) or spherical (S) joints with two and three dof respectively have been considered.

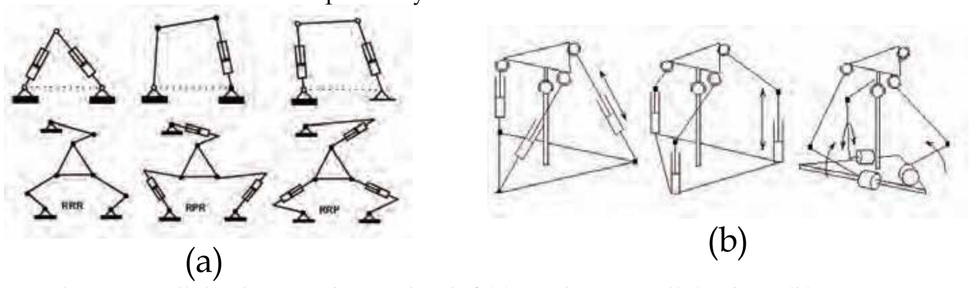


Fig. 1. Planar parallel robots with 2 and 3 dof (a); and 3D parallel robots (b).

2.2 Morphology of the parallel robot proposed as climbing platform

The morphology of the parallel robot proposed to climb along different structures is formed by two parallel rings of 6 dof. The main structure of the robot is similar to the classic structure of the parallel robot based on the Stewart-Gough (S-G) platform (Galt 1997; Merlet 1997). This platform consists of two rings linked by six linear actuators as UPS kinematic chains (Universal-Prismatic-Shperical joints) (see Fig. 2).

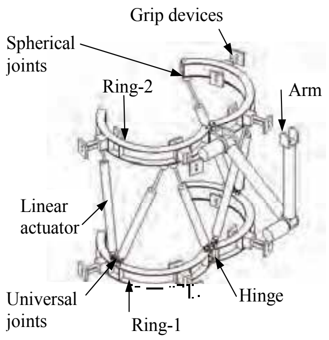


Fig. 2. Mechanical design of a parallel robot to climb along tubular structures.

On this general morphology several modifications can be accomplished with the purpose the robot can climb along other structures. In this way different legs can be added on each one of the external rings of the robot with the purpose the parallel robot can work outside wall of pipes. These legs allow fastening one ring to the pipe while the other ring moves along the structure. These modifications will be discussed in Section 4.

3. Mathematical analysis of parallel robots

Kinematic models of parallel robots present essential differences with regard to serial robots. In serial robots a systematic methodology exists providing a direct kinematic model of the robot independent of its configuration. Inverse kinematic models are more complex and it is necessary to take into account geometric restrictions for each configuration, or to resolve equations by numerical methods and so providing different solutions in some occasions. Otherwise parallel robots present the inverse problem. Inverse kinematic model is easy to obtain taking into account geometric restrictions, while direct kinematic model presents a greater complexity.

Many authors have been proposed different methods to provide kinematic models for parallel robots (Fitcher, 1986; Merlet 1990). In this way Fitcher provided the kinematic equations of general platform (Stewart platform) and also dynamic equations were anticipated. Merlet considered some design aspects of the Stewart platform providing some guidelines to resolve the kinematic equations and the jacobian. Also the dynamic equations were extended.

In the following subsections kinematics and dynamics aspects of parallel robots will be discussed. We have chosen for this discussion a parallel platform of six dof as presented in figure 3.

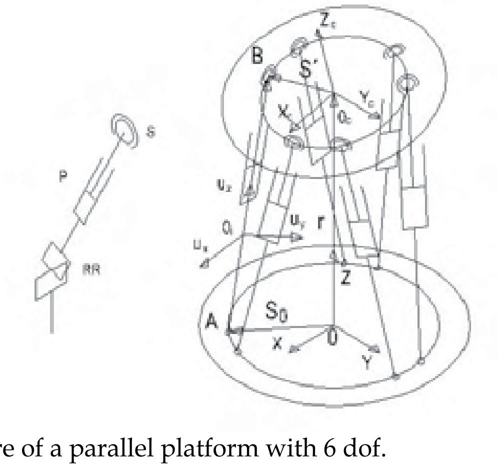


Fig. 3. General structure of a parallel platform with 6 dof.

3.1 Inverse Kinematic Model

The inverse kinematic solution in a parallel robot as described in previos section, can be defined by the set of equations that express the joint values of the kinematic chain depending on the configuration of the end effector. The approach of such equations can be expressed in a vectorial way. Taking as reference the system of figure 3 representing a parallel robot with 6 dof with UPS kinematic chains, for each chain we can express:

$$\vec{r}^{\,i} = \vec{r}_1 + A_1 \vec{s}_1^{\,i} - \vec{s}_0^{\,i} \qquad i = 1,\ldots 6 \tag{1}$$

where:

- A_1 is the rotation matrix that represents the orientation of the basis
- $\vec{s}_1^{\,i}$ and $\vec{s}_0^{\,i}$ are position vectors that locate universal and spherical joints with respect to base reference frame and mobile ring
- $\vec{r}_1$ is the position vector of the mobile ring expressed in the reference frame of the base
- and finally $\vec{r}^{\,i}$ is the vector of each one of the linear actuators.

These six equations constitute the inverse kinematics model in which the length of the actuators are the module of the vector $\vec{r}^{\,i}$. As can be observed, this methodology produces, except singular configurations, the solution in an easy way. In some singular cases, several solutions can be provided in a similar way than the serial robots.

3.2 Direct Kinematic Model

The direct kinematic model, or in other words, to calculate the position and orientation of the movil base from the length of the actuators is a more complicated problem. Analytical solutions to this problem only have been found in some easy configurations making use of the Denavit-Hartenberg formulation o with geometric restrictions (Innocenti & Parenti-Castelli, 1993; Ait-Ahmed & Renaud, 1993). To achieve the solution for more complex configurations it is necessary to employ numerical methods (Almonacid et al., 2003). These methods provides higher degree polynomical solutions with multiple solutions. Other authors (Bonev & Ryu, 1999) proposed a practical solution consisting of employing redundant sensors to resolve the direct problem. These extra sensors allow to achieve the pose of the robot. However this solution entails aditional problems and is dificult to carry out.

A better solution consists of employing multibody formulation (Saltaren et al., 2004). This method provides an easy computational solution at the same time that provides a systemathic methodology independent of the robot configuration. Also in case of exit the method provides only one solution. However the method presents important disadvantages due to it is necessary to assure the convergence and the solution is very sensitive to initial conditions.

3.3 Dynamic Model

As well as the kinematic model, the dynamic model of parallel robots presents important differences with regard to serial robots. Parallel robots not present a general equation that defines its dynamic behaviour as serial robots are:

$$M(q)\ddot{q} + C(q,\dot{q})\dot{q} + G(q) = \tau \tag{2}$$

Some suggestions have been presented based on Newton-Euler formulation (Dasgupta & Mruthyunjaya, 1998; Tsai 1999), but all of them take into account some distinctive features and lead to computational algorithms with an indeterminate efficiency.

The most interesting and employed formulation, comes from the multibody dynamics using the Newton-Euler equations along with the Lagrange restrictions. Through this formulation the inverse and direct dynamics problem can be resolved in an efficient manner. This formulation employs the following equations:

$$\begin{aligned} &M\ddot{q} + \Phi_q^T \lambda = Q \\ &\Phi_q \dot{q} = v \\ &\Phi_q \ddot{q} = \gamma \\ &\Phi = 0 \end{aligned} \tag{3}$$

Where:

- q is the generalized position of the system
- Q are the external generalized forces applied to the system
- M is the generalized mass matrix of the system
- Φ is the movement constraint vector
- λ is the vector of variables of Lagrange
- v is the velocity vector
- γ is the acceleration vector

To resolve the last equation it is necessary to employ numerical methods. As a consequence, an analytic dynamic model can not be derived.

3.4 Control of Parallel Robots

The non-existence of analytics dynamics models of the parallel robots produces a great difficulty to approach general control algorithms for them. The control systems proposed until now are relatively simple (not couple systems): the kinematics model generates references for the joints of the robot. In (Liu et al., 1992) some aspects with regard to the control parallel robots in the working space are discussed. However, the direct extension of control scheme to real time systems presents several problems due to the high computational cost to resolve the direct kinematic model. Another option consists of planning the path in the Euclidean space due to only the inverse kinematic model is required and this is easier to resolve.
In [Aracil et al., 2003] a control algorithm for a climbing parallel robot is discussed. The proposed scheme makes use of the information provided by ultrasonic sensors along with a double feedback loop, one for the control of the position of each actuator (designed with a PD controller), and the other one to centre de robot.

4. Climbing Parallel Robots

Using parallel robots as climbing robots offer a lot of possibilities. During last years several climbing parallel robots have been developed for using them in different applications. So, the use of Stewart-Gough platform as climbing robot to perform tasks in tubular structures such as oil pipes, bridge steel cables, towers and trunks of palm trees is very promising. The basic morphology is shown in figure 2. The platform is formed by two rings joined with 6 linear actuators as UPS kinematics chains (where the U degrees of freedom belongs to an universal joint, P is a prismatic degree of freedom that belongs to the linear actuator and S is the spherical joint). The robot assembly around the tubular structure is carried out through a system of hinges. The holding systems are based on a series of grip devices built in each ring. Those grip devices hold the reference ring firmly attached to the tubular structure while the free ring is displaced by the control system.

The automatic control of the robot that climbs along tubular structures should take into account geometrical changes in the path of the tubes. As a consequence, three 120^o ultrasonic sensors were installed in every ring. The three sensors allow calculating the difference between the centre of the ring and the tube (Aracil et al., 2003). Based on this estimate, an algorithm to control the displacement of the moving ring can be done by maintaining it centred and following the curve of the tube automatically (Almonacid et al. 2003). The climbing process is composed by four steps working in sequence. The steps are shown in figure 4. It consists of holding around the tube and moving up by using holding systems attached to the rings. It displaces along the tube as one ring holds up and the other free ring moves on.
On this basis a climbing parallel robot was developed to climb on a palm trunk. In figure 5 several images show the robot in different positions of the palm trunk. The first version of this prototype designed to climb this kind of structures moved to a velocity of 0.4 m/s. It was composed of 6 pneumatic cylinders. Every cylinder is controlled through a proportional valve FESTO MPYE-5. A linear encoder measures its displacement. The gripping system, which is activated pneumatically, can be seen in every ring. A multi-axis Delta Tau PMC-VME card has been used for the control structure of the robot.

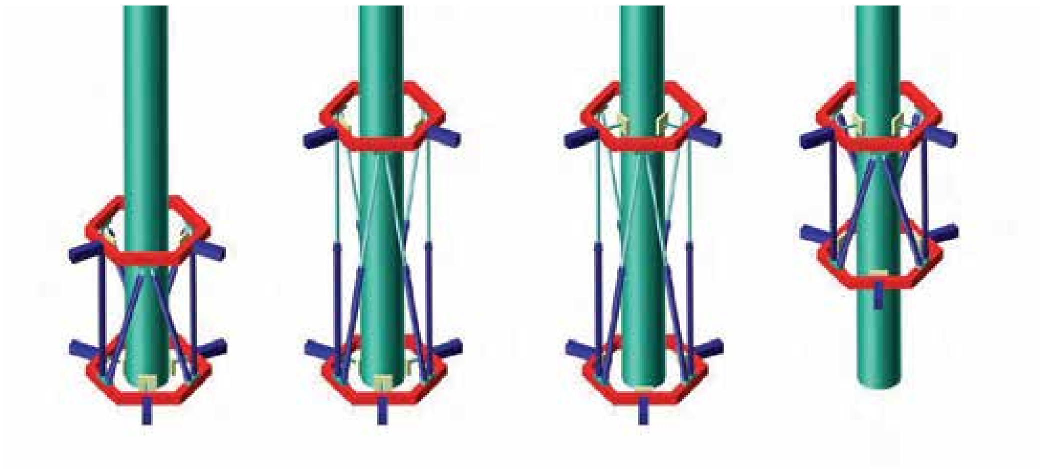


Fig. 4. Steps that define the climbing process of the climbing parallel robot.

But parallel robots also can climb inside tubes or pipes. To perform these tasks, it is necessary to modify some aspects like the holding systems and also the robot universal joints with the aim the robot was capable to have big rotations as observed in figure 6. These joints should be more mechanically robust in contrast to the standard universal joints (Aracil et al., 2006). As can be observed, the gripping system is radial to the rings.
Another possibility to climb along tubular structures as tubes or pipes is by means of a couple of arms that can be extended and retracted. This couple of arms is connected to each ring and serves as holding device. The main advantage of such holding system consists of the robot is capable of climb along complicated structures with obstacles due to its structural design. In this sense, the robot can adopt several forms and postures to overcome complicated zones as can be observed in figure 7.

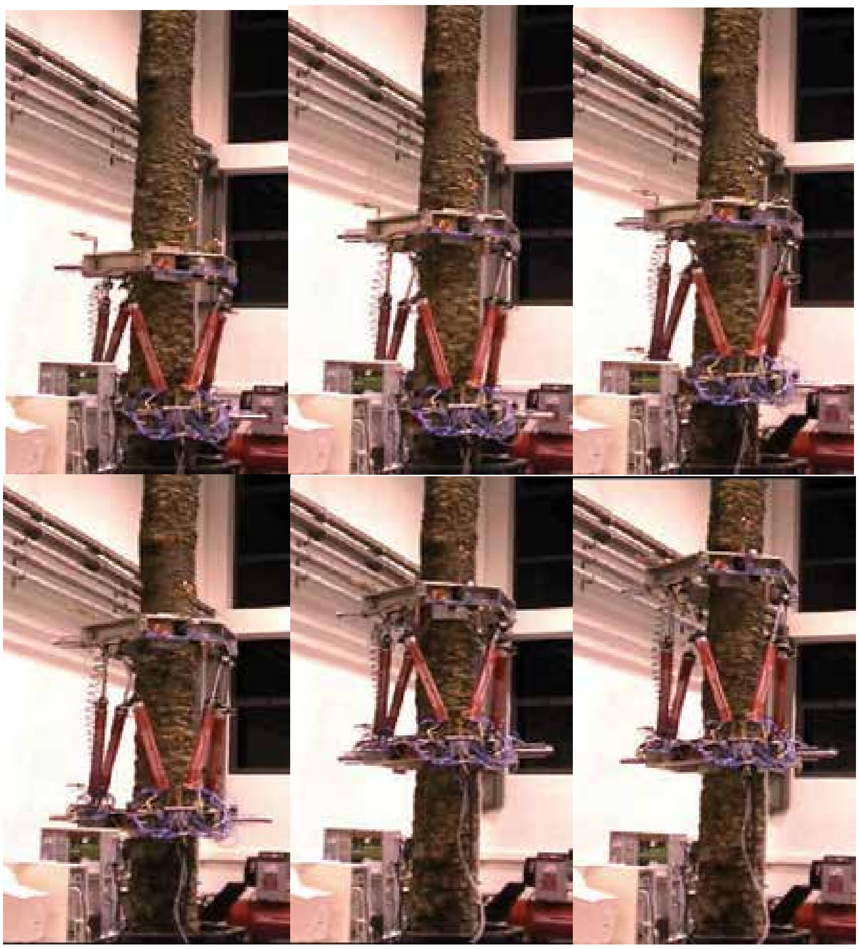


Fig. 5. Different images of the parallel robot climbing o a palm tree trunk.

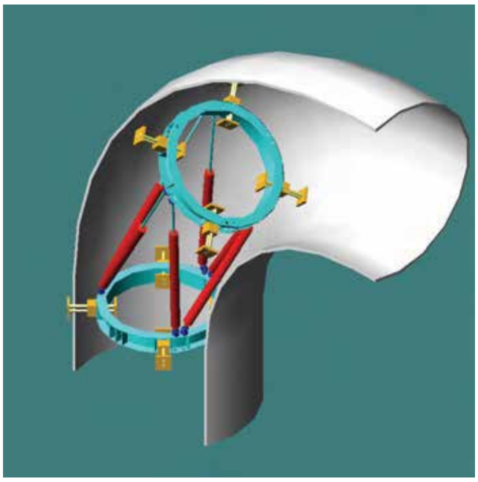


Fig. 6. Mechanical adaptation an parallel robot for climbing inside of tubes.

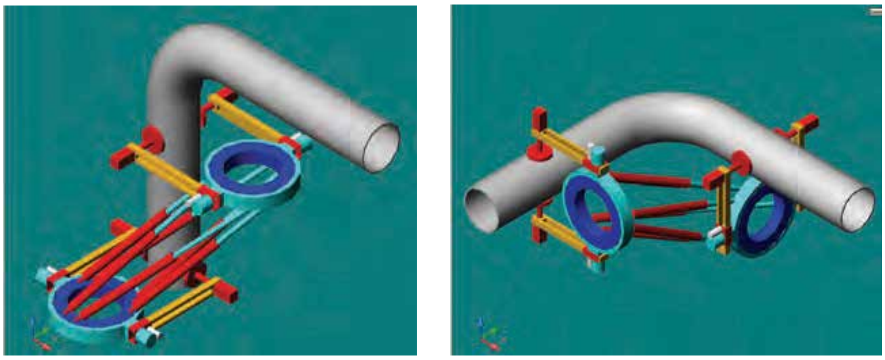


Fig. 7. Parallel robot climbing outside tubular structures.

A solution for the problem of displacement and climbing along metallic structures should theoretically be based on a mechanism whose power actuators are part of the structure, connected directly to the end effector and with a large geometric dexterity to overcome obstacles using minimal movements. Also, the robot should be lightweight, mechanically simple, with a large payload capacity and high velocity. Taking into account these features we have proposed a climbing parallel robot capable to develop tasks on such metallic structures. Following with the previous adaptations of the basic climbing parallel robot, also is possible using this parallel robot to climb along structural frames or metallic structures. In the development of this climbing parallel robot (CPR) it is necessary to carry out some modifications on the general platform with the purpose of facilitating the movements of the robot on such structures (Saltaren et al., 2005).

In contrast with the previous climbing robots used to work on palm trees with an interior clamping device to hold and climb by palm and pipes, the clamping devices of the robot should fold and extend at least in two predefined positions. The possibility of folding or extending the clamping legs allows reducing possible collisions between the movement ring and the environment. Moreover in some sequences of displacement it is necessary to orientate the legs of each one of the rings of the robot to predetermined positions (-90º, 0º, +90º), because the rotation of the exterior ring with its clamping device may reduce the rotation requirements of both rings around its axes. For this reason it is possible to avoid the collisions between the linear actuators originated when they cross themselves in the displacement of each one of the two rings of the robot. So, with the purpose of allowing configurations of 90º between the rings that constitute the basis of the robot, the spherical and universal joints have been adapted and redesigned (see figure 8).

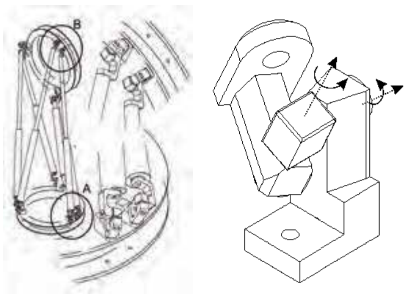


Fig. 8. Detail of the joints adapted with the purpose of the robot can accomplish postures of 90º.

The displacement of the robot can be carried out as a sequence of four simple movements. Figure 9 shows a sequence of these movements. This displacement is composed of four steps:

- The robot is grasped to the beam with both rings legs
- The robot is held by ring 1 legs. Ring 2 legs are released and folded. Linear actuators are commanded allowing ring 2 to acquire the required position.
- The ring that has been displaced (ring 2) is held through its clamping devices.
- With ring 2 grasped to the beam, ring 1 is released. Ring 1 acquires the new position. Once ring 1 has achieved the position, the robot is ready for a new cycle.

So, the postures to generate movements through a right path are simple and easily reachable. However, when it is necessary to overcome and pass structural nodes in the metallic structures, the robot must be able to acquire more complicated postures. A structural node is composed by three beams making a corner. In such structural nodes the robot can change the direction of its movement or can keep the same direction when passes it. In figure 10 several postures that the robot can adopt are presented. This sequence shows that the robot requires a minimum number of postures to pass a structural node.

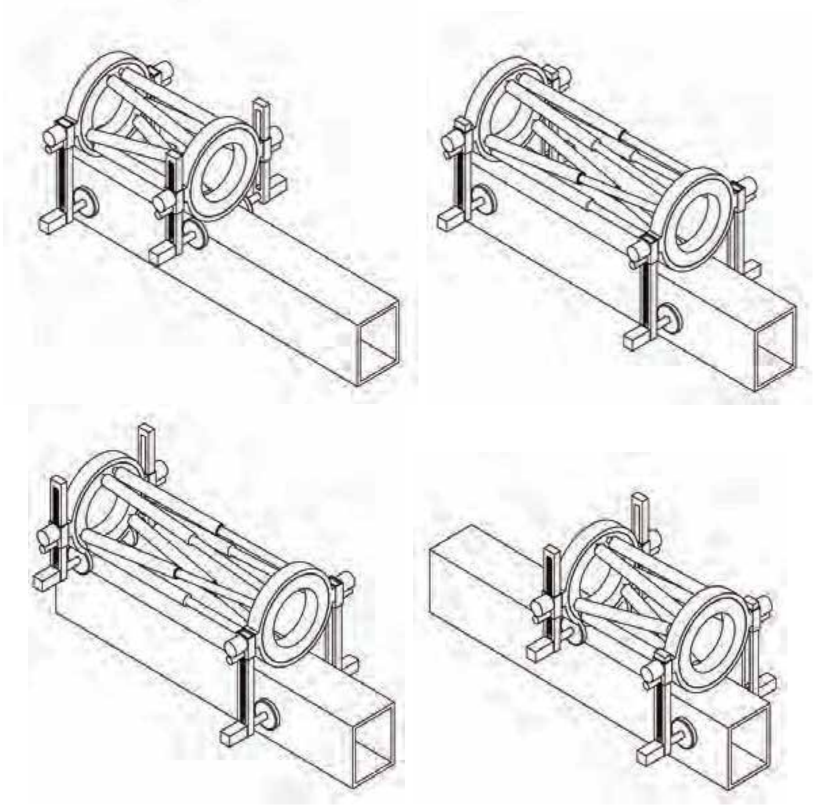


Fig. 9. Sequence of displacements of the robot along the length of a straight trajectory.

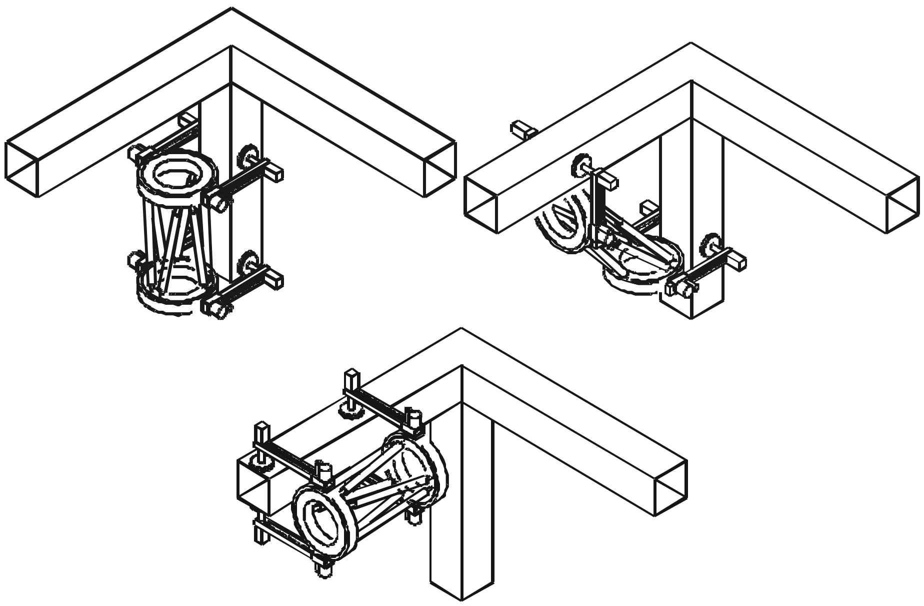


Fig. 10. Example of postures evading a structural node.

A detailed analysis of such postures is presented in (Saltaren et al., 2005). The Jacobian of the robot about each one of the postures is calculated. Also, based on the jacobian analysis, the angular limits of the upper ring can be calculated for each one of the postures of the robot. Such analysis shows the feasibility of using the presented parallel robot as climbing robot on metallic structures. Also the dynamic analysis of the postures needed to evade a structural node is presented and analysed.
Taking into account the results reported in (Saltaren et al., 2005), an electrical climbing parallel robot prototype has been developed. The main features of this prototype are: 300 mm of ring diameter, 490 mm of length actuators and a working velocity of 0.2 m/sec. The experimental analysis with this prototype showed the big potential of using parallel robot as climbing robot.

5. Other applications with parallel robots

The great variety of possible configurations of parallel robots produces that light modifications of such mechanisms allow their use in multiple applications. So, we can find parallel robots from micro robots to high platforms with a high payload, or from medical to spatial applications. Taking as basis the robot presented in figure 2 we have developed other parallel robots with special features depending of the application field.
Such is the case of ROBOTENIS parallel robot designed and developed with the purpose of making experiments of visual control systems (Angel et al., 2005). This robot can achieve a velocity of 2m/sec.
Also, other parallel robots making use of the capacity of adaptation due to their variable geometry have been designed, developed and tested. Such is the case of the submarine robot REMO (Aracil et al., 2005). The adaptation can be achieved through the six dof parallel platform. This parallel platform allows a great manoeuvrability

besides the possibility to achieve large depths due to its watertight volume is much reduced.
We have derived a second prototype of this robot (REMO II) with several improvements. We have added one motor in each ring to produce the driving forces. So, the navigation is based in the generation of driving forces due to the combination of forces produced by the motors mounted in the rings. This procedure allows achieving significant advantages with regard to the previous prototype.

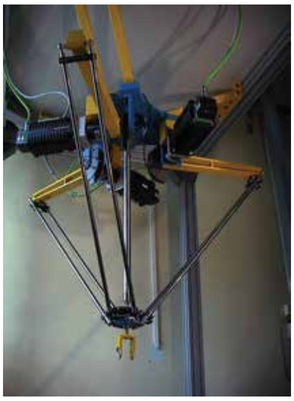
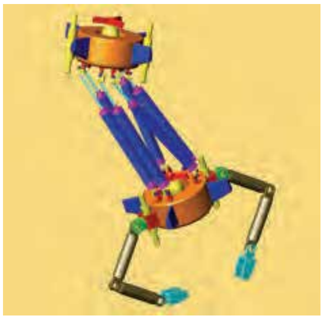
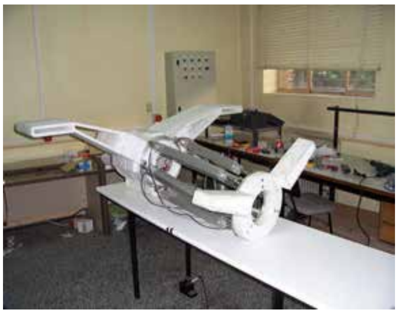




Fig. 11. Another parallel robots: ROBOTENIS (UPM), REMO (UPM) and REMO II (UPM).

6. Conclusions and Future work

At present, parallel robots show a great progress in their development due to their behaviour in multiple applications. In this sense, the Stewart-Gough platform with proper mechanical adaptations could be used for a climbing parallel robot. A climbing parallel robot with 6 degrees of freedom has been proposed and analysed. Parallel robots have great advantages compared to serial robots with legs using as climbing robots. Some advantages can be cited as the high weight payload capacity, robustness, simplicity of its mechanical design, etc. However, some difficulties have to be resolve too as the adaptation to the structure meanwhile the robot accomplishes the task or the singularities that can be produced in the movement on the structure.
In this paper several options of a climbing parallel robot based in the S-G platform have been presented and analysed. First of all, an initial prototype to climb along tubular structures as pipes or palm trunks has been shown. The main features of this robot are the pneumatic actuators and the external control loop that allows centring the robot to the tubular structure. Taking into account this first prototype, another parallel robot to resolve the problem of displacement and climbing along metallic structures has been presented. The main features of this robot are the electrical actuators and the clamping device developed with the purpose of reducing possible collisions between the movement ring of the robot and the environment. The results achieved with both prototypes show the validity of these platforms to undertake tasks in which are necessary to climb on. In short, this paper tries to show the big potential of using parallel robots as climbing robots.
Also, we are working in introducing teleoperation techniques to manage these parallel robots. So, the robots will be controlled in a semi-automatic way when it displaces along the tubular or metallic structures.

7. References

Ait-Ahmed, M. & Renaud, M. (1993). Polynomial representation of the forward kinematics of a 6 dof parallel manipulator, *Proceedings of International Symposium on Intelligent Robotics*, Bangalore Ed.

Almonacid, M.; Saltaren, R.; Aracil, R. & Reinoso, O. (2003). Motion planning of a climbing parallel robot. *IEEE Transactions on Robotics and Automation,* 19, 3, 485-489

Almonacid, M.; Agrawal, S.; Aracil, R.; Saltaren, R. (2001). Multi-body dynamics analysis of a 6-dof parallel robot. *Proc. of the 2001 ASME Inern. Symposium on Advances in Robot Dynamics and Control,* New York, November

Angel, L.; Sebastián, J.M.; Saltaren, R.; Aracil, R. & Sanpedro, J. (2005). Robotenis: optimal design of a parallel robot with high performance. *IEEE/RSJ Int. Conf. on Intelligent robots and systems IROS 2005*, I.

Aracil, R.; Saltaren, R. & Reinoso, O. (2003). Parallel robots for autonomous climbing along tubular structures. *Robotics and Autonomous Systems*, 42, 2, 125-134

Aracil, R.; Saltaren, R.; Ferre, M.; Yime, E. & Alvarez, C. (2005). REMO project: design , modelling and hydrodynamic simulation of a robot of variable geometry for actuations on marine disasters. *Vertimar: Symposium on Marine accidental oil spills*

Aracil, R.; Saltaren, R. & Reinoso, O. (2006). Climbing Parallel Robot CPR. A robot to climb along tubular and metallic structures. *IEEE Robotics and Automation Magazine*, 13, 1, 16-22

Bonev, I.A. & Ryu, J. (1999). A simple new closed-form solution of the direct kinematics using three linear extra sensors. *IEEE/ASME Ing. Conf. on Advanced Intelligent Mechanism,* Atlanta 19-23 Sept., 526-530

Cappel, K.L. (1967). Motion simulator. *US Patent Nº 3295224*

Dasgupta, B. & Mruthyunjaya, T.S. (1999). Closed form dynamic equations of the general Stewart platform through the Newton-Euler approach. *Mechanism and Machine Theory,* 33, 7, 993-1011

Fichter, E.F. (1986). A stewart platform based manipulator: general theory and practical construction. *Int. Journal of Robotic Research,* 5, 2, 157-181

Galt, S. & Luk, M. (1997). Evolutionary design and development techniques for an 8-legged robot. *Genetic Algorithms in Engineering Systems: Innovations and Applications,* 446, 259-263

Gwinnet, J.E. (1931). Amusement devices. *US Patent Nº 1789680*

Fichter, E.F. (1986). A stewart platform based manipulator: general theory and practical construction. *Int. Journal of Robotic Research, 5, 2*, 157-181

Huang, T.; Wang, J.; Gosselin, M. & Whitehouse, D. (1999). Determination of closed form solution to the 2-d orientation workspace of Gough-Stewart parallel manipulators, *IEEE Transactions on Robotics and Automation*, 15, 6, 1121-1125

Hunt, K.H. (1978). Kinematic geometry of mechanism, *Claredon Press, Oxford*

Liu, K.; Lebret, J.A.; Lowe & Lewis, F.L. (1992). Control of a Stewart platform based robotic milling cell. *Proc. of ASME WinterAnnual meeting, Symp. On Manufacturing anc control issues in a telerobotics assembly workcell.* 1, 8-13

Lazard D. (1992). Stewart platform and Grobner basis. *In ARK.* 136-142

McCallion, H. and Pham, D.T. (1979). The analysis of a six degrees of freedom work station for mechanized assembly.*Proc. of ASME Winter Annual Meeting, Symp. On Manufacturing and control issues in a telerobotics assembly workcell*. 1, 8-13

Merlet, J.P. (1990). An algorithm for the forward kinematics of general 6 dof parallel manipulators. Research Report 1331. INRIA

Merlet, J.P. (1997). *Les Robots paralleles*. Ed. Hermes, ISBN 2-86601-599-1

Saltaren, R.; Aracil, R. & Reinoso, O. (2004). Analysis of a climbing parallel robot for construction applications. *Computer-aided civil and infrastructure engineering,* 19, 436-445

Saltaren, R., Aracil, R. & Reinoso, O. (2005). Climbing Parallel Robot: A computational and Experimental Study of its Performance around Structural Nodes. *IEEE Transactions on Robotics*, 21, 6, 1056-1066

Stewart, D. (1965). A platform with 6 degrees of freedom. *Proc. of the Institution of Mechanical Engineers,* 180, Part 1,15, 371-386

Tsai, L.W. (1999). Robot Analysis: the mechanics of serial and parallel manipulators. *Wiley Interscience.* John Wiley and Sons

Robotics for Improving Quality, Safety and Productivity in Intensive Agriculture: Challenges and Opportunities

Gustavo Belforte[1], Paolo Gay[2], Davide Ricauda Aimonino[2]
[1]*Politecnico di Torino,* [2]*Università degli Studi di Torino*
Italy

1. Introduction

Despite the large diffusion of robotic and automated solutions that took place during the last decades in most production processes, the agricultural sector only marginally benefited from automated solutions (such as the control of climatic parameters in greenhouses). Robotic applications have been confined so far almost only to research studies with few particular and specific applications made available for farmers.
This lack of advanced robotic solutions in agriculture and particularly in intensive farming cannot be motivated claiming a lack of interest for the latest results of technology and research in this sector. The use of latest results of genetics for the production of hybrid flower plants is one of the many examples that contradicts this kind of argument and shows how much the agricultural sector is keen to exploit the opportunities offered by modern technology and research.
However, from the automation point of view, agriculture remains mainly labour intensive not only in those countries where manpower is relatively cheap, but also for those enterprises which enthusiastically embrace latest technologies in an effort to improve their competitiveness and to ensure top quality products.
The motivation for the little development that robotic solutions deserved so far in the agricultural sector lies elsewhere. On one side it is related to some particularities of the specific sector, like the fact that farming is usually performed in an unstructured environment that is therefore less friendly for robotic solutions than a well structured industrial environment (Kassler, 2001). On the other side a consistent share of the research effort conducted so far, mainly tried to use standard industrial robotic solutions adapting them to the intensive farming sector instead of developing brand new solutions that exploit the specific features of the agricultural sector. Finally the focus of most research was so far on single specific activities or tasks and less frequently on the whole production process or on a consistent share of it. In our opinion the approach should be revised looking for new specific robotic solutions that take advantage of peculiarities and needs encountered in the agricultural sector that are different from those of the industry. In this context, beside overcoming specific constraints encountered in the farming context, the important challenge is to find solutions and develop technology to automatize consistent shares of the agricultural process. From a business point of view it is important to stress that, for the

robotic industry, the agricultural sector represents a new important market which is ripe for adopting robotic solutions that can be very effective in increasing the quality of products, in improving the safety for the workers of the sector, in reducing pollution and environmental impact, and in ensuring higher productivity.
In this paper the state of the art of robotic research and of available robotic solutions in intensive agriculture is presented first. The particular features that valuable robotic solutions should deserve for agricultural applications are then outlined discussing also which are, in the authors' opinion, the most promising research directions for the next years. Such research should address different problems to develop competences and build new knowledge needed for developing valuable practical solutions suited for the specific agricultural sector. Finally some research results attained by the authors along some of the lines previously described are illustrated.

2. State of the art of robotic studies in agriculture

In order to understand which are the research opportunities at hand in the sector of robotic automation for agricultural processes, it is convenient to review the main research studies carried on in the last years. As remarked before agriculture is still labour intensive also in those countries in which the cost of manpower is very high and robotic automation solutions are widely used in other industrial sectors. Some applications are indeed available as illustrated in Kondo and Ting (1998), but most solutions which have been developed so far mainly focus on specific problems trying to automatize single operations. The available studies can be grouped, for an easier understanding, in two main classes depending on whether they deal with applications in field or in greenhouses. The reason for this grouping is related to the fact that, although many operations must be conducted in both environments, nonetheless there are some important differences in the two settings that are related to the available infrastructures and to the expected productivity (and economic return) per unit surface that sets constraints to acceptable investments. All this ends up in remarkable differences between solutions that can be considered satisfactory in the two different conditions.
The most important lines of study, for what concerns automated solutions for application in greenhouses, are related to specific cultural operations, to harvesting of different crops, and to guidance problems. In Tillet (1993) an excellent overview on robotic applications in horticulture is presented. Transplanting and seeding are some of the cultural operations whose automation has been specifically studied in the works of Ryu et al. (2001) and of Tai et al. (1994). In the wide chapter of automated harvesting Kondo and Monta (1999) introduce a strawberry harvesting robot, while Reed et al. (2001) address the automatic harvesting of cultivated mushrooms. Cho et al. (2002) consider a lettuce harvesting robot equipped with a computer vision system, while an automated cucumber picking by a robot was studied by Van Henten et al. (2002) and (2003). Finally Mandow et al. (1996) and Sandini et al. (1990) discuss guidance topics for robots in greenhouses.
For what concerns applications in the field, automatic guidance has attracted consistent research efforts. Automatic guidance systems for tractors and for other specific machines such as harvesters, transplanters, etc. have been implemented based on different technological solutions. Some systems make use of artificial vision as in the works of Hague and Tillet (1996), Hague et al. (2000), Marchant et al. (1997), Åstrand and Baerveldt (2002), and Tillet (1991). A particular section is represented by studies which use stereoscopic vision

as in the works of Kise et al. (2005) and Rovida Màs et al. (2004). The second technological solution which was studied in more recent years is based on the use of global navigation satellite systems (GNSS) presently represented by the American global positioning system (GPS) (see e.g. Thuilot et al., 2001, and Bak & Jakobsen, 2004). The joint use of GPS and artificial vision has been proposed by Benson et al. (2003) and Chen et al. (2003), while solutions based on lasers (Jimenez et al., 1999), (Château et al., 2000) and ultrasonic sensing (Harper & Mc Kerrow, 2001) have been also investigated.
Other automated solutions studied for specific cultural operations are represented by the work of (Tillet & Hague, 1999) for the automatic hoe control; the works of (Tillet et al., 2002; Åstrand & Baerveldt, 2002) for the inter-row raking and tilling for weed control in sugar beet; the works of Tian et al. (1999), Paice et al. (1995), Tillet et al. (1998) for precision spraying devoted to weed control and crop treatment; the work of Blasco et al. (2002) for weed control implemented with electrical discharges.
Final fields of study for automated applications in the field are represented by robotic solutions for harvesting, as illustrated by the works of Monta et al. (1995), Sarig (1993), Sanders (2005) and Peterson et al. (1999), and mapping yield and fruit quality (Quiao et al., 2005).
It is worth noting that most studies in literature deal with one specific agricultural operation for which automated solutions are studied and presented. This ends up in most cases with the design and testing of a dedicated machine that is often even quite sophisticated, but can perform only one operation. The case of multipurpose robots that can perform different tasks is usually an exception although some studies in this category (Monta et al. 1995, Van Henten et al. 2002, 2003, and 2006) are also available.

3. Desired features for agricultural robots

In order to analyze which are the desirable features for robots to be used in agriculture and therefore which are the most promising lines for research as well as for engineering and product development it is informative to reconsider some of the relevant features of currently available robots and to understand how some of them have impaired the diffusion of robotic solutions in agriculture.
One important aspect is that robots are in general quite expensive and sophisticated. They require manpower with specific skills that are usually not available in agricultural workers and need specific infrastructures (power supply, systems for their movement throughout the crop, etc.). Particularly relevant is the fact that currently available robots developed for factory applications usually have precisions in the range of tenths or hundredths of millimeter. Such precision is about two to three order of magnitude greater than the precisions required in most agricultural operations where errors in the range of some millimeters are usually satisfactory. Remark that the higher is the required precision the higher is the weight of the robot, since its structure must be rigid to avoid deformations, and this ends up with higher power consumption and higher costs.
From these simple considerations it follows a first very important guideline for research and product development related to robotic solutions in agriculture: it is important to design specific robots with mild precision (in the range of millimeters) and therefore much less expensive than typical industrial robots.
Although the cost of robots for agricultural operations can be consistently reduced according to the previous guideline, however it cannot become negligible. To be appealing robotic solutions must offer a significant improvement in productivity with a reduction of costs that justifies their

use with respect to standard solutions involving human labour. This obvious consideration leads to other two new guidelines. One concerns the type of cultures for which robotic solutions should be studied first. In fact, when automatizing intensive and highly remunerative cultures it is more easy to ensure an economic return that compensates higher investments. Remark that all cultures in greenhouses are of this kind and note also that greenhouses offer infrastructures (such as power supply, artificial illumination, plinths to which rails can be hooked, etc.) and represent an environment that is usually partially structured or can be partially structured (pots/plants are regularly displaced on benches or on the earth, obstacles are in known positions, etc.).
The second guideline for research and product development concerning robotic solutions in agriculture is then to focus on applications in greenhouses.
The third guideline steaming from the need to ensure significant improvement over the standard use of human manpower indicates that valuable robotic solutions should be versatile and cover different tasks usually performed by human operators. This in order to reduce the amount of labour needed. If ideally all the tasks to be performed throughout the growth of one crop (planting, irrigating, fertilizing, spraying, weed control, etc.) could be managed by a robot, then human operators could just perform supervising tasks. In connection with this guideline it is important to note that the option of several dedicated robots each one specialized in one specific task does not seem a convenient strategy. In fact the different tasks are usually performed in different periods for only a few times per year, so that the robots would work one at a time while the other would remain idle. Moreover sharing the machine with other farmers in order to reduce costs would not be a feasible solution in most cases because neighbourhood properties usually follow the same calendar and need the machine at the same time. The third guideline is then to look for versatile multitask robots that ideally can manage all the operations needed in the production cycle of different crops.
The forth and last guideline concerns the development of new applications and tasks that can be performed only by a robot and are aimed to improve the crop quality, to improve safety and to reduce pollution and costs. A nice example of such a task is offered by precision spraying for cyclamens. This crop frequently needs treatment. Since the pests to be targeted live under the leaves, the spraying should occur under the leaf canopy in order to be really useful. The cyclamens should then be sprayed one by one, properly positioning the spraying nozzle. This indeed is practically impossible with human operators when the number of plants to be processed is huge as it is usually in dedicated enterprises. The task, however, can be done by robots that do not lose concentration during the operations and can work indefinitely day and night. Another field in this line is represented by the operations involving vision and image processing. Beside surveillance for pest and illnesses detection, image processing can be used to control the growth of each single plant/flower/fruit as well as to identify the level of ripeness in order to plan optimal harvesting time.

4. Structure of robot systems for greenhouses

Different robotic solutions for greenhouses are presented and analyzed in this section. A first important choice is related to the robot structure. Among the different possible solutions (anthropomorphic, scara, Cartesian, etc.) the most convenient one seems to be a Cartesian configuration. Actually the working space in which the robot must move to operate on plants in greenhouses is orthotopic since plants are positioned on rectangular benches (or rectangular portion of soil) so that their position (or the position of their parts) can be easily identified in a three coordinate Cartesian reference. Knowing the position of the plants or retrieving it from video images, convenient trajectories can be easily defined in a Cartesian reference and are immediately translated (without inverse kinematics problems) into commands to the robot if its structure is also Cartesian. It must be noted that the width of benches is in many cases in the range of 3000 mm so that a working space in the range of 3000×1500×1000 mm can be representative of several practical situations.

Drawbacks of a non cartesian structure were experimented by the authors in studies and experiments executed using a robot with pantographic structure (Belforte et al. 2006). It was practically noticed that its non-Cartesian structure consistently increased the required computation for trajectory evaluation and motion control, because the commercially available industrial axis controllers that were used are designed for driving motions along linear axes, therefore they can be easily integrated into cartesian robots while they require more or less complex customization for non-cartesian structures. Last but not least mechanical components and technical solutions available on the market allow fast assembling of relatively low cost Cartesian robots with a wide range of dimensions while their accuracy remains higher than the one required in agricultural operations. Such flexibility in the robot size is of primary importance since greenhouse dimensions are definitely not standardized and new robots should allow adaptation to existing greenhouses if they have to penetrate this new market.

All the above considerations strongly recommend cartesian robot structures. In the following different possible solution of this kind are outlined, however another distinctive characteristic of the robot has to be discussed first: the capability of displacement within the greenhouse. Actually the choice is between fixed robots that operate in a fixed point on mobile benches and mobile robots that operate on the whole surface of the greenhouse.

The fixed robot solution has the advantage that the robotic cell can be highly sophisticated with different equipment that do not need to be moved around and can then be quite heavy. The supply of any required substance/material to the cell for its working needs must be ensured in one single point of the greenhouse and is therefore easier as well as the removal of products/debris. However the fixed point robot requires moving benches that are a quite sophisticated and expensive infrastructure that is available only in relatively few cases.

Moving robots on the other hand can operate on fixed benches or directly on the ground, but require a moving support that adds complexity to the machine and requires some infrastructure in the greenhouse (rails or paths along which the movement can occur). Such kind of robot cannot be very heavy and requires a more careful design for what concerns the feeding/removal of substances/parts required or generated during the robot operations. In general these requirements should not constitute a severe constraint and a moving robot solution seems to be slightly more flexible and definitely less expensive. Here different possible solutions of cartesian robots are introduced.

4.1 Fixed gantry robot with moving benches

In this solution the robot has a gate structure. Its advantage is to be symmetric (especially for loads and forces), while the disadvantage is the requirement of a clear space on both sides of the benches for the supports of the robot.

4.2 Fixed throat depth robot with moving benches

In this solution the robot stands on one side only of the benches. While this solution does not require clear space on both sides of the benches at the same time it is asymmetric and this could require to make it heavier to avoid deformation.

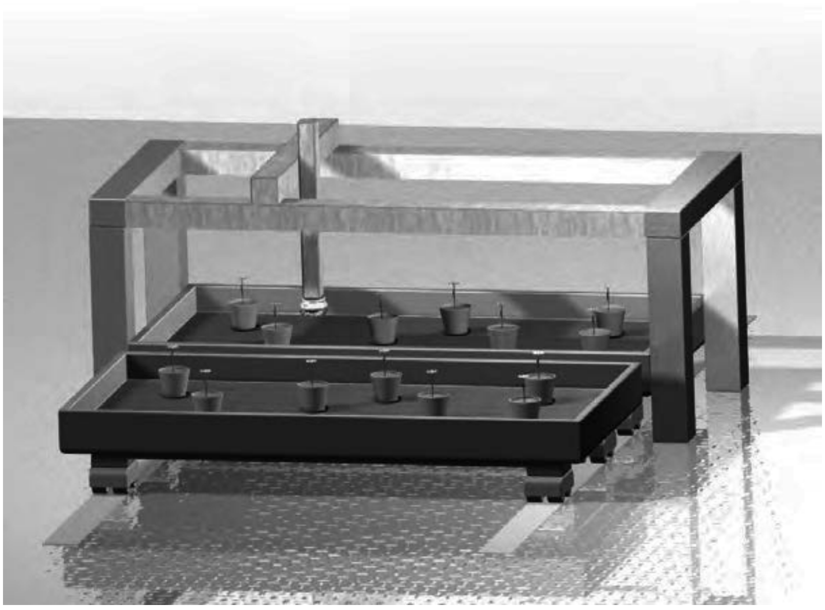


Fig. 1. Fixed gantry robot with moving benches.

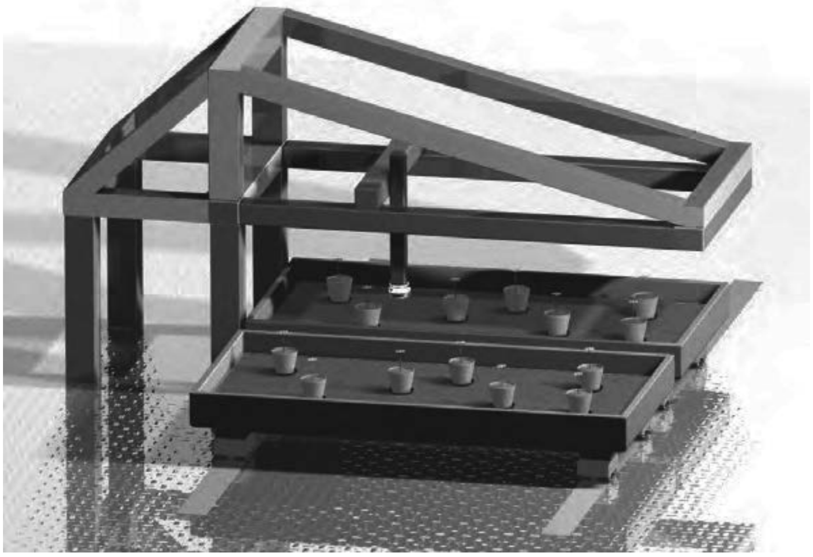


Fig. 2. Fixed throat depth robot.

4.3 Moving gantry robot

This structure can be adapted to work both on fixed benches and directly on the soil. The gate structure laying directly on the ground allows to construct heavier robots and could be used also outside greenhouses.

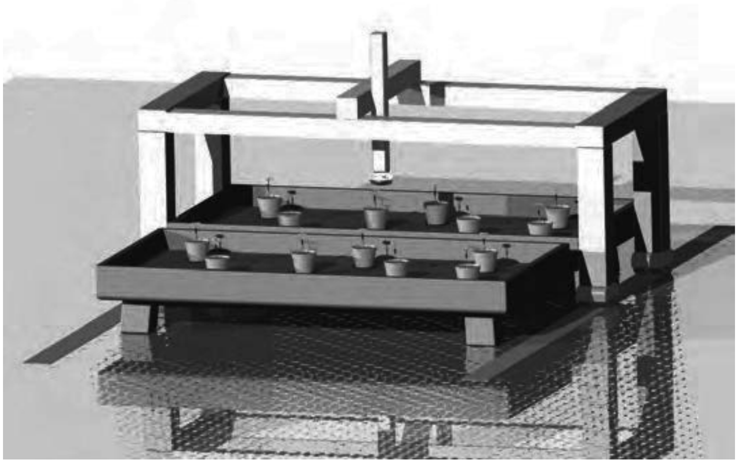


Fig. 3. Moving gantry robot.

4.4 Moving asymmetric gantry robot

Also this structure can be adapted to work both on fixed benches and directly on the soil. Since it requires an elevated rail, such solution could take advantage of existing columns in greenhouses. At the same time it is less suited for use in open fields.

4.5 Aerial robot

This structure is similar to the previous two ones. Its main difference is the fact that both rails are high and possibly attached to the columns or to the structures of the greenhouse. The main advantage of such structure is that it does not occupy any portion of the ground. Indeed when the rails on which the robot moves along are attached to the columns or structures of the greenhouse it should be verified that these last are strong enough to support also the robot. The problem is less important if the robot is designed together with the greenhouse, but if the robot is added in an existing greenhouse this point must be carefully considered.

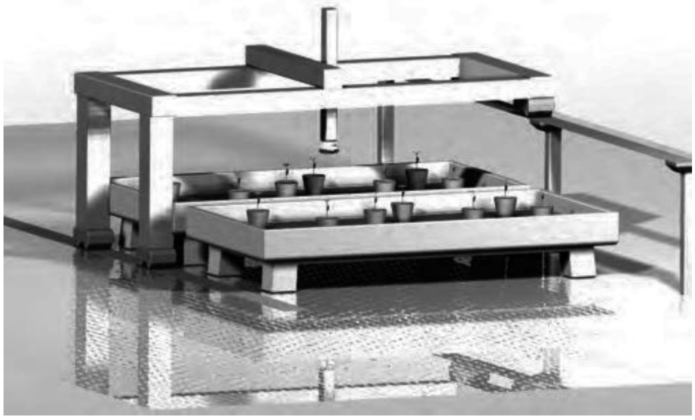


Fig. 4. Moving asymmetric gantry robot.

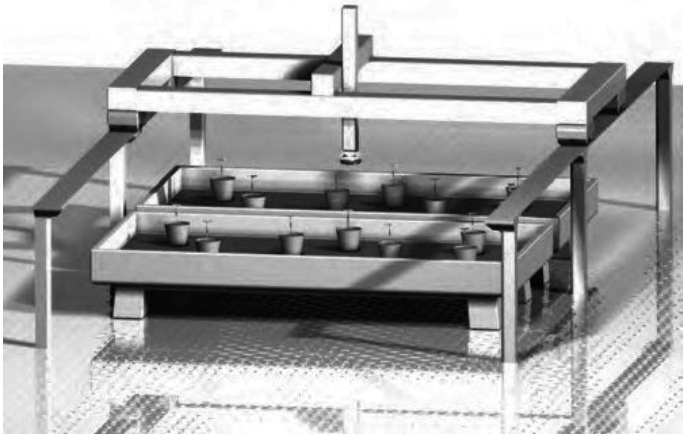


Fig. 5. Aerial moving robot.

4.6 Moving throat depth robot
This robot has the same advantages/disadvantages of the similar fixed structure, indeed with the difference that in this case the robot is mobile.

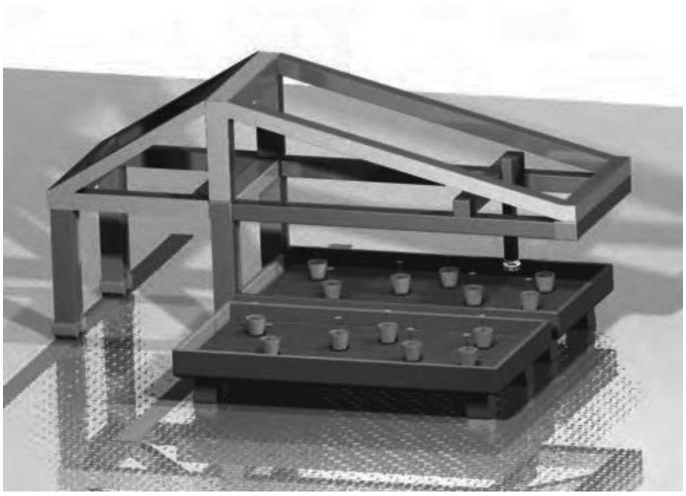


Fig. 6. Moving throat depth robot.

4.7 Vertical mounted moving robot
This structure seems very different from the previous ones, but the difference is more apparent than real. In this case the robot is placed vertically instead than horizontally. The movement in the greenhouse (or perhaps in the field) is facilitated by two rails one on the floor and the other high over the previous one. In this way the robot can better operate on plants with a consistent vertical extension that are organized in rows (like fruit plants and similar).

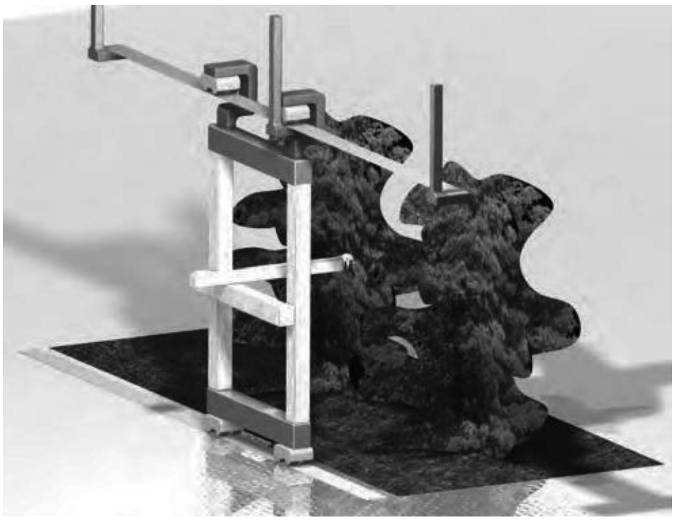


Fig. 7. Vertical mounted moving robot.

4.7 Hanging robot

This last structure could be convenient in those greenhouses that are equipped with one monorail running high over the soil to which different equipment can be attached. Indeed these solution could pose severe constraints on the weight of the robot.
The main advantage consists in keeping the soil clear from supports and in using of existing solutions integrated in some greenhouses.

5. Design of an experimental robot for applications in agriculture

Based on the above considerations and on previous experiences the research group, to which the authors belong, decided to construct a cartesian robot prototype to be used for research and study of automated solutions in agriculture.

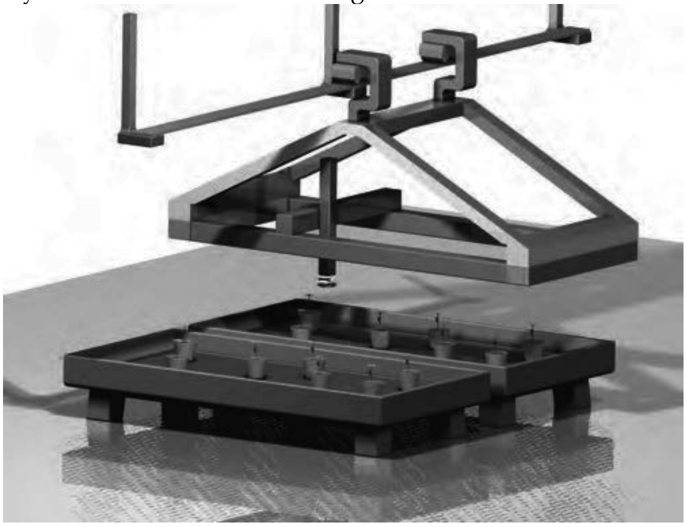


Fig. 8. Hanging robot.

Beside providing a research tool, the construction of the robot was expected to represent a first tentative in the direction of relatively low cost robots specifically minded for agricultural applications.
For this reason the robot was developed assembling together to the largest possible extent standard components available on the market.
While for commercial robot the operability throughout the greenhouse is an essential feature that can be ensured with any of the solutions discussed in Section 3, for a research tool this feature is not important. In fact the focus in research is on developing tools, programs and procedures to enable the robot to perform different basic operations within its operational volume. Once this is achieved the integration of the robot with moving benches or its coupling with a dislocation system will enable to operate on the whole crop throughout the greenhouse. For these reasons the prototype we constructed works in fixed position.
The robot structure is represented in fig. 9. The dimensions of the robot are 3,60 m (x-axis), 2,50 m (y-axis) and 0,80 m (z-axis), according to the guidelines discussed above. The electrical motors ensuring the movement to the robot axes are standard 48V DC motors with encoder and speed transducers. Four independent PWM axis drivers, plugged to velocity sensors, supply and control the motors.
The robot control logic has been implemented on a National Instruments 1000b PXI controller unit equipped with a PXI-8170 Pentium III CPU board, a PXI-7344 Motion Controller and a PXI-6255 I/O board.
The PWM amplifiers are driven by the digital axes controller on the PXI-7344 board. The position measurements of the four axes are fed-back to the digital PID controllers on the 7344 that generates velocity paths. Input/output signals are managed by a PXI-6255 board. Maximal velocities along the axes are of 1 m/s and enable the robot to move fast in order to ensure a satisfactory productivity.
The robot has been equipped with a forth axis that allows complete rotation around the z axis. This axis is powered by a standard 48V DC motor with encoder and speed transducer. The gearing ration has been designed in order to allow mechanical operation to the soil, such as weed control.
The robot has been equipped with a vision subsystem constituted by one colour and one NIR CCD camera (both are KC-40H1P Rapitron Day and Night CCD cameras) fixed to the y-axis. This solution allows to maintain a constant distance from the floor of the bench, i.e. to operate with fixed focus. The camera view angles allow to investigate only a portion of the bench at a time, but this is not a limitation since the cameras are used to plan trajectories in the neighbourhood of the single plants and not to control wide displacements, that are controlled using encoder feedback.

6. Elementary agricultural operations and tasks performed by the robot

As discussed before it is important to enable the robot to do as many operations as possible. In previous studies (Belforte et al. 2006) performed with a pantographic robot structure some elementary operations were already studied. They were: a) the precise spraying of cyclamines and b) the precise fertilization. The interest of these operations lies in the fact that they cannot be manually performed on extended cultures because human operators lose their concentration in this kind of repetitive activity and are far to slow to ensure acceptable costs.

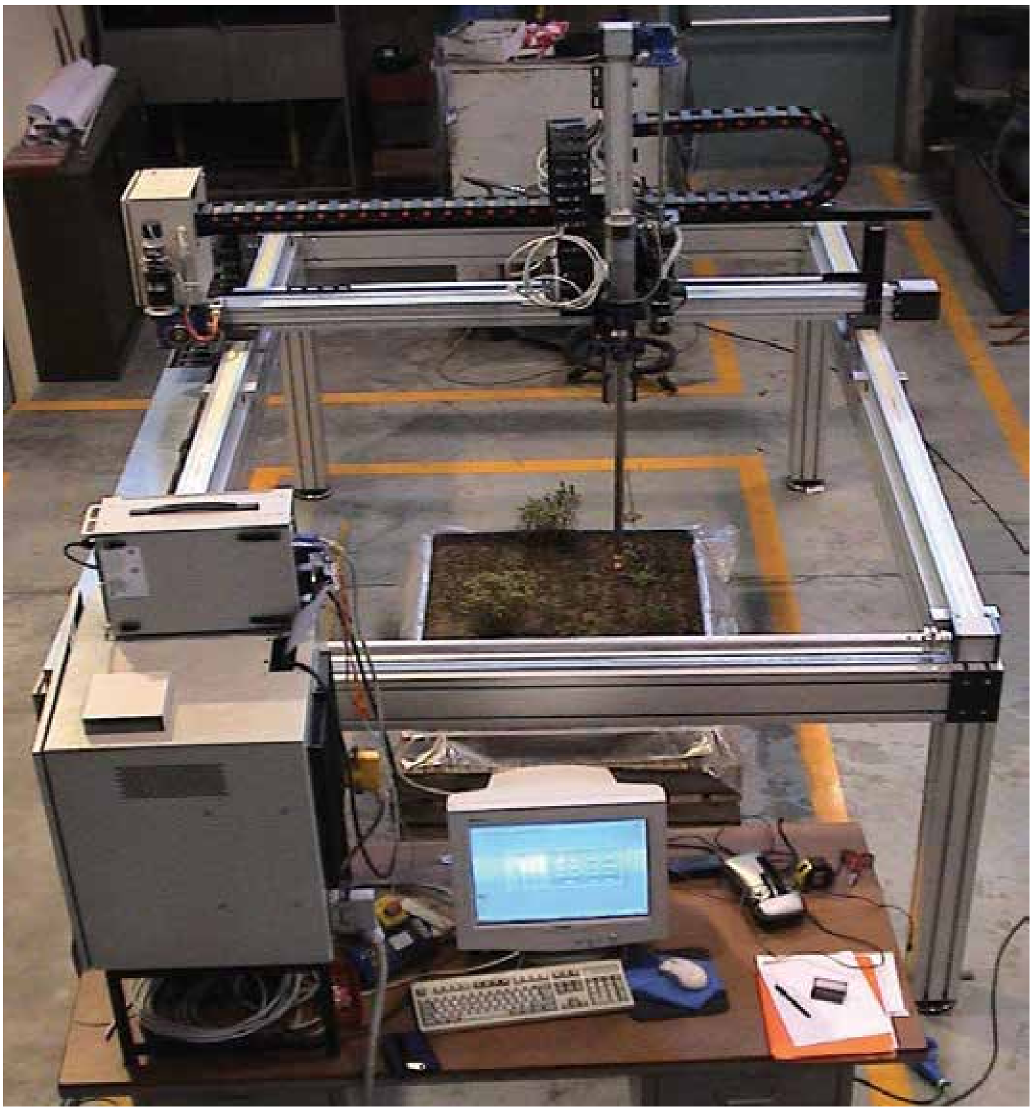


Fig. 9. Cartesian fixed gantry robot prototype.

With this new robot a third operation has been added: c) mechanical weed control. This is obtained with a "fork-like tool" secured to the end effector of the robot, that rotates along the *z* axis, moved by the forth axis motor. The tool can then weed the soil around the plants mechanically removing not desired vegetation. The interest of this operation lies in the fact that it could substitute, at least in part, chemical treatments and could therefore contribute to pollution reduction and to a safer working environment.

7. Future activities

In Section 3 it has been evidenced that robots should be able to perform as many elementary operations as possible in order to facilitate their industrial use in intensive farming. This because a robot that can automatically manage the whole growing cycle of several crops is commercially appealing. To contribute to the definition of this kind of robot the future lines of study and research that we will develop mainly consist in the derivation of tools and techniques enabling other significant elementary operations. In doing this particular attention will be devoted to the integration of the elementary operations with the artificial vision system integrated in the robot. Actually the vision system seems to be a very important part of an efficient robot for two reasons. The one is that several elementary operation need guidance and control. In the greenhouse where the environment is not completely structured and changes can be caused by chance or by the action of human operators, a visual control for the localization of plants and obstacles seems an almost obliged choice. The second reason is that the artificial vision system integrated in a robot can itself perform evaluations, comparisons, and monitoring of growth and of other characteristics that cannot be performed by human operators and could become highly useful for optimal management of crops.

8. Acknowledgements

The authors would like to thank National Instruments Italy s.r.l., Agricontrol s.n.c. and CeRSAA for their support to this research, and R.Deboli, S. Tribocco, P. Pezzuto, G. Perin and G. Benvegnù for their cooperation in constructing the robot.

9. References

Åstrand, B.; Baerveldt, A., J (2002). An agricultural mobile robot with vision-based perception for mechanical weed control. *Autonomus Robots*, 13, 1, (July 2002) 21-35, ISSN 0929-5593

Bak, T.; Jakobsen, H. (2004). Agricultural robotic platform with four wheel steering for weed detection. *Biosystems Engeneering*, 87, 2, (February 2004) 125-136, ISSN 1537-5110

Belforte, G.; Deboli, R.; Gay, P.; Piccarolo, P.; Ricauda Aimonino, D. (2006). Robot Design and Testing for Greenhouse Applications. *Biosystem Engineering, 95, 3, (November 2006) 309-321, ISSN 1537-5110*

Benson, E. R.; Reid, J. F.; Zhang, Q. (2003). Machine vision-based guidance system for an agricultural small-grain harvester. *Transactions of the ASAE*, 46, 4, (July/August 2003) 1255-1264, ISSN 0001-2351

Blasco, J.; Alexios, N.; Roger, J. M.; Rabatel, G.; Moltò, E. (2002). Robotic weed control using machine vision. *Biosystems Engeneering*, 83, 2, (October 2002) 149-157, ISSN 1537-5110

Chateau, T. ; Debain, C. ; Collange, F. ; Trassoudaine, L.; Alizon, J. (2000). Automatic guidance of agricultural vehicles using a laser sensor. *Computers and Electronics in Agriculture*, 28, 3, (September 2000) 243-257, ISSN 0168-1699

Chen, B.; Tojo, S.; Watanabe, K. (2003). Machine vision based guidance system for automatic rice transplanters, *Transactions of the ASAE*, 46, 1, (January/February 2003) 91-97, ISSN 0001-2351

Cho, S. I.; Chang, S. J.; Kim, Y. Y. (2002). Development of a three-degrees-of-freedom robot for harvesting lettuce using machine vision and fuzzy logic control. *Biosystems Engineering*, 82, 2, (June 2002), 143-149, ISSN 1537-5110

Hague, T.; Tillet, N. D. (1996). Navigation and control of an autonomous horticultural robot. *Mechatronics*, 6, 2, (March, 1996)165-180, ISSN 0957-4158

Hague, T.; Marchant, J. A.; Tillet, N. D. (2000). Ground based sensing system for autonomous agricultural vehicles. *Computers and Electronics in Agriculture*, 25, 1-2, (January 2000) 11-28, ISSN 0168-1699

Harper, N.; Mc Kerrow, P. (2001). Recognising plants with ultrasonic sensing for mobile robot navigation. *Robotics and Autonomous System*, 34, 2-3, (February 2001) 71-82, ISSN 0921-8890

Jimenez, A. R.; Ceres, R.; Pons, J. L. (1999). A machine vision system using a laser radar applied to robotic fruit harvesting. *Proceedings of the IEEE Workshop on Computer Vision Beyond the Visible Spectrum: Methods and Applications*, pp. 110 – 119, ISBN 0-7695-0050-1, Ft. Collins CO USA, June 1999, IEEE Inc, Piscataway NJ USA

Kassler, M. (2001). Agricultural automation in the new millennium, *Computers and Electronics in Agriculture*, 30, (February 2001) 237-240, ISSN 0168-1699

Kise, M.; Zhang, Q.; Rovira Màs, F. (2005). Stereovision-based crop row detection method for tractor-automated guidance. *Biosystems Engineering*, 90, 4, (April 2005) 357-367, ISSN 1537-5110

Kondo N; Monta M (1999). Strawberry harvesting robots. *Proceeding of ASAE/CSAE-CGR Annual International Meeting*, Paper No. 99-3071, July 1999, Toronto Canada, ASAE, St. Joseph MI USA

Kondo, N.; Ting, K. C. (1998). *Robotics for Bioproduction Systems*, ASAE, ISBN 0-929355-94-6, St Joseph MI USA

Marchant, J. A.; Hague, T.; Tillet, N. D. (1997). Row-following accuracy of an autonomous vision-guided agricultural vehicle. *Computers and Electronics in Agriculture*, 16, 2, (January 1997) 165-175, ISSN 0168-1699

Monta, M.; Kondo, N.; Shibano, Y. (1995). Agricultural robot in grape production system, *Proceedings of the IEEE International Conference on Robotics and Automation*, pp. 2504 – 2509 ISBN 0-7803-1965-6, Nagoya Japan, May 1995, IEEE Inc, Piscataway NJ USA

Paice, M. E. R.; Miller, P. C. H.; Bodle, J. D. (1995). An experimental sprayer for the selective application of herbicides. *Journal of Agricultural Engineering Research*, 60, 2, (February 1995)107-116, ISSN 0021-8634

Peterson, D. L.; Bennedsen, B. S.; Anger, W. C.; Wolford, S. D. (1999). A systems approach to robotic bulk harvesting of apples. *Transactions of the ASAE*, 42, 4, (July/August 1999) 871-876, ISSN 0021-8634

Qiao, J.; Sasao, A.; Shibusawa, S.; Kondo, N.; Morimoto, E. (2005). Mapping yield and quality using the mobile fruit grading robot. *Biosystems Engineering*, 90, 2, (February, 2005) 135-142, ISSN 1537-5110

Reed, J. N.; Miles, S. J.; Butler, J.; Baldwin, M.; Noble, R. (2001). Automatic mushroom harvester development, *Journal of Agricultural Engineering Research*, 78, 1, (January 2001) 15-23, ISSN 0021-8634

Rovia-Màs, F.; Zhang, Q.; Kise, M. (2004). Stereo vision navigation of agricultural tractor in structure fields, *Proceeding of AgEng 2004 International Conference*, Paper No. 163, ISBN 90-76019-258, Leuven Belgium, September 2004, Tecnologisch Instituut, Leuven

Ryu, K. H.; Kim, G.; Han, J. S. (2001). Development of a robotic transplanter for bedding plants, *Journal of Agricultural Engineering Research*, 78, 2, (February 2001) 141-146, ISSN 0021-8634

Sanders, K. F. (2005). Orange Harvesting Systems Review. *Biosystems Engineering*, 90, 2, (February 2005) 115–125, ISSN 1537-5110

Sarig, Y. (1993). Robotic of fruit harvesting: a state-of-the-art review, *Journal of Agricultural Engineering Research*, 54, 4, (November 93) 265-280, ISSN 0021-8634

Tai, Y. W.; Ling, P. P.; Ting, K. C. (1994). Machine vision assisted robotic seedling transplanting. *Transactions of the ASAE*, 37, 2, (March/April 1994) 661-667, ISSN 0021-8634

Thuilot, B.; Cariou, C.; Cordesses, L.; Martinet, P. (2001). Automatic guidance of a farm tractor along curved paths, using a unique CP-DGPS. *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, 674 – 679, October 2001, IEEE Inc, Piscataway NJ USA

Tian, L.; Reid, J. F.; Hummel, J. W. (1999). Development of a precision sprayer for site-specific weed management. *Transactions of the ASAE*, 42, 4, (July/August 1999) 893-900, ISSN 0021-8634

Tillet, N. D. (1991). Automatic guidance sensor for agricultural field machine. *Journal of Agricultural Engineering Research*, 50, 3, (November 1991) 167-187, ISSN 0021-8634

Tillet, N. D. (1993). Robotic manipulators in horticulture: a review, *Journal of Agricultural Engineering Research*, 55, 2, (June 1993) 89-105, ISSN 0021-8634

Tillet, N. D.; Hague, T. (1999). Computer-vision-based hoe guidance for cereals - an initial trial. *Journal of Agricultural Engineering Research*, 74, 3, (November 1999) 225-236, ISSN 0021-8634

Tillet, N. D.; Hague, T.; Marchant, J. A. (1998). A robotic system for plant-scale husbandry. *Journal of Agricultural Engineering Research*, 69, 3, (March, 1998) 169-178, ISSN 0021-8634

Tillet, N. D.; Hague, T.; Miles, S. J. (2002). Inter-row vision guidance for mechanical weed control in sugar beet. *Computers and Electronics in Agriculture*, 33, (March 2002) 163-177, ISSN 0168-1699

Van Henten, E. J.; Hemming, J.; Van Tuijl, B. A. J.; Kornet, J. G.; Meuleman, J.; Bontsema, J.; Van Os, E.A. (2002). An Autonomous Robot for Harvesting Cucumbers in Greenhouses. *Autonomous Robots*, 13, 3, (November, 2002) 241–258, ISSN 0929-5593

Van Henten, E. J.; Hemming, J.; Van Tuijl, B. A. J.; Kornet, J. G.; Bontsema, J.; Van Os, E. A. (2003). Field test of an autonomous cucumber picking robot. *Biosystem Engeneering*, 86, 3, (November, 2003) 305-313, ISSN 1537-5110

Van Henten, E. J.; Van Tuijl, B. A. J.; Hoogakker, G. J.; Van Der Weerd, M. J.; Hemming, J.; Kornet, J. G.; Bontsema, J. (2006). An Autonomous Robot for De-leafing Cucumber Plants grown in a High-wire Cultivation System, *Biosystems Engineering*, 94, 3, (July 2006) 317–323, ISSN 1537-5110

Edited by Low Kin Huat

This book covers a wide range of topics relating to advanced industrial robotics, sensors and automation technologies. Although being highly technical and complex in nature, the papers presented in this book represent some of the latest cutting edge technologies and advancements in industrial robotics technology.

ISBN 978-953-51-5808-0

IntechOpen