Robotic Assembly of Splined Shaft-Hole Using Hybrid Compliance Control and Machine Learning based Logic Branching Algorithm

Arun Kumar Jaura

A Thesis

in

The Department

of

Mechanical Engineering

Presented in Partial Fulfilment of the Requirements
for the Degree of Doctor of Philosophy at
Concordia University
Montreal, Quebec, Canada

August 1996

ISBN 0-612-18407-2

ABSTRACT

Robotic Assembly of Splined Shaft-Hole Using
Hybrid Compliance Control and Machine Learning Based Logic Branching Algorithm

Arun Kumar Jaura, Ph.D.
Concordia University, 1996

Albeit the pace of growth of robotics has hastened considerably in the recent years, there are certain vital areas requiring significant improvement, especially manufacturing and assembly. The principal goal of this work is to explore the feasibility of splined shaft and hole assembly using machine learning to learn correct responses to contact forces encountered during assembly. A mixed flavor of learning by examples and learning by induction has been used to train the robot. A novel hybrid compliance control strategy has been used in the assembly process. The hybrid compliance control approach uses active implicit compliance control in conjunction with a passive compliance device. The active control provides force feedback enabling the robot follow an assembly path during task execution and the passive device supplements corrections for instantaneous time dependent assembly path uncertainties that arise from inherent errors in the system and manufacturing irregularities The design simplicity of the passive compliance device developed and used in the system is noteworthy, as it is based on a modular concept resulting in its versatile adaptability. Other secondary objectives are to map the contact forces to relative part locations, using weights and logic branching search tree, and to estimate the practical limits of this mapping relationship. To start the assembly process, random and gaussian techniques have been employed, both of these produce a similar

performance. The machine learning algorithm was developed ingeniously, in a way to avoid use of vision systems for assembly. The work is directed towards making a niche in the robotic assembly domain, by making it adaptable to parts of any geometry. The proposed algorithm and the hardware developed, i.e., the force sensor, and passive compliance device, were tested on a SCARA robot for splined shaft-hole and simple shaft-hole assemblies and encouraging results were obtained.

ACKNOWLEDGEMENTS

At the outset I wish to express my sincere thanks to my supervisors, Professors M.O.M.Osman and N. Krouglicof for their guidance in this research. The long hours they spent discussing various aspects of this work and the fruitful brain storming sessions helped shape this work in its present form. They are to a considerable extent responsible for any merit this research may have; it is scarcely necessary to add that its faults are my responsibilities alone. I am especially grateful to Professor Krouglicof for his invaluable assistance in the establishment of certain details of this research work, and finally reading the manuscript and checking many of the details.

I wish to acknowledge the assistance of Mr. Denis Richard of École de technologie supérieure (ÉTS) in evolving the communication routines and robot calibration for experiments. Thanks to Mr. Éric Corbeil and Mr. Sylvain Lampron of ÉTS for their help in machining and interfacing the hardware. Thanks to Mr. Richard Mainardi of Concordia University for his help with some AutoCad drawings. Many colleagues and friends, too numerous to mention by name, are also acknowledged for their support. Special thanks go to Professor T.S.Sankar by virtue of whose efforts I could undertake this research endeavour in Canada. My gratitude to the Graduate School for awarding me the elite 'Concordia University Fellowship'. I also wish to thank the Dean of Engineering and Computer Science, Dr. D. J. Taddeo and his assistant Dr. G. Turski for providing additional financial assistance and cooperation in administrative issues. Thanks

go to the Mechanical Engineering secretarial staff for their day to day administrative support and cooperation.

Finally, no words can convey my indebtedness to my parents and family in India for their constant moral support during this research work. My wife especially, deserves the greatest and very personal gratitude for being by my side as an inspiration and extending to me her moral support that was an invaluable aid and stimulus through out this research period. Also, special thanks to my wife for taking complete responsibility of all activities on the home front as well as taking care of our dear toddlers.

To

my wife, Meenakshi

daughter, Roshni

and son, Akhil

Contents

LIST OF FIGURES

LIST OF TABLES

LIST OF CHARTS

LIST OF SYMBOLS

Symbol.	*Description*
ΔX	Incremental change in X
ΔY	Incremental change in Y
ΔZ	Incremental change in Z
d_o	Offset between the sensor elements on mounting platform
L	Length of straining element
h	Distance between both mounting platforms
R	Radius of platform
F_X	Force in X direction
ΔF_X	Incremental change in F_X
F_Y	Force in Y direction
ΔF_Y	Incremental change in F_Y
F_Z	Force in Z direction
ΔF_Z	Incremental change in F_Z
M_X	Moment in X direction
ΔM_X	Incremental change in M_X
M_Y	Moment in Y direction
ΔM_Y	Incremental change in M_Y
M_Z	Moment in Z direction
ΔM_Z	Incremental change in M_Z

H	Distance between gripper CG and passive device plate
ϕ	Cocking misalignment
R_C	Cocking stiffness
K_L	Lateral stiffness
θ	Angle of approach of shaft into hole
c	Clearance
ε_0	Distance between shaft and hole axis
W	Sum of chamfer widths on peg and hole
dA	Incremental move in translation
T	Transformation matrix for translation and rotation
E	Young`s Modulus
A	Surface area
ε	Strain
ΔL	Change in strain gauge length
ΔR	Change in resistance
Rg	Resistance of strain gauge
S_A	Sensitivity of gauge
S_L	Linear Sensitivity
S_T	Transverse Sensitivity
G	Gauge factor
ν	Poisson Ratio
I	Current

V	Voltage
K	Spring stiffness
GI	Grading Index
GI_S	Grading Index (System performance)
GI_L	Grading Index (Learning Rate)
ϕ	Chamfer angle
β	friction angle ($\tan^{-1} \mu$)
C	Clearance ratio
γ_C	angle between line perpendicular to hole wall and line from compliance center of hole to contact point during chamfer crossing
γ	angle between line perpendicular to hole wall and line from compliance center of hole to contact point during one-point and two-point contact
D	diameter of hole
d	diameter of shaft
Δ	initial lateral error
a	distance from end of shaft to compliance center of shaft
a_0	optimized distance from end of shaft to compliance center of shaft
C_h	distance perpendicular to hole wall to hole`s compliance center
C_v	distance parallel to hole wall from corner of chamfer to compliance center of hole
r_c	distance between contact point and compliance center of hole during chamfer crossing
r	distance between contact point and compliance center of hole during one and two point contact

ς	$(\Delta - CD/2)\tan\phi - \Delta z$
l	driving insertion variable during one and two point contact
K_θ	rotational stiffness of peg support
$K_{\theta 1}$	rotational stiffness of left hand hole wall
$K_{\theta 2}$	rotational stiffness of right hole wall
K_x	lateral stiffness of shaft support
K_{x1}	lateral stiffness of left hole wall
K_{x2}	lateral stiffness of right hole wall
$\delta\theta$	rotational displacement of shaft
$\delta\theta_1$	rotational displacement of left hole wall
$\delta\theta_2$	rotational displacement of right hole wall
δx	lateral displacement of shaft`s compliance center
δx_1	lateral displacement of left hole wall
δx_2	lateral displacement of right hole wall
δz	vertical displacement of shaft`s compliance center
V	distance parallel to right hole wall from contact point to compliance center of hole
F_n	normal contact force during chamfer crossing
F_{n1}	normal contact force on left hole wall during one-point contact and two point contact
F_{n2}	normal contact force on right wall during two-point contact

Preface

The structure of the thesis is based in a way to highlight the development of hardware and to bring out the software subtleties of the machine learning algorithm, in addition to various experiments and simulations that have been carried out to support this research.

Chapter 1 provides a general outline of the robotics world and various difficulties encountered by today's researchers. This chapter superficially introduces assembly in a shop-floor environment.

Chapter 2 highlights the state-of-the art technology review. Efforts have been made to make this review exhaustive. The chapter is divided into four sections. One section describes the research on assembly strategies since late 1970, and the effort of various researchers to bring the technology to the point where it stands today. Another section describes various robotic wrist sensors that are currently available. A section in this chapter highlights the compliance control techniques and devices, both active and passive, that have been developed. Finally, the chapter has a section on various techniques used to teach robots to perform different tasks.

Chapter 3 is devoted to the proposed research work. It begins with the goals, followed by the motivation to carry out this work, briefly outlining the origin of the idea and the scope of this work. It highlights robot usage in industry and future projections as well.

Chapter 4 describes the development of the force-torque sensor and the passive compliance device. These two important components are mounted at the end of the robot arm in conjunction. The force coupling matrix, principles of measurement, design, detailed drawings and calibration have been supplemented by Appendices A through E.

Chapter 5 has been devoted to the assembly task formulation. This chapter describes various reference frames, assumptions made in the task process, various errors anticipated and major assembly bottlenecks. The termination conditions, assembly paths and parameters to be monitored have also been described in this chapter.

Chapter 6 starts with an overview of artificial intelligence and continues with the reasons for using machine learning in general and this research in particular. It describes the initial learning model and its versions as used by other researchers. The chapter discusses the algorithm that has been used in this research. Actual learning and selecting the good move are also discussed. Codes have been written in C and C++, and are found in Appendix G.

Chapter 7 discusses various equipment used in the experiments. The experiments that were carried out are also discussed and simulation results using the algorithm are presented. Test

set-up details have been included in Appendix F and codes are in Appendix G. Inferences drawn from the experiments are also presented.

Chapter 8 presents conclusions of this work. It presents the author`s research contributions and observations drawn as a result of experiments.

Chapter 9 concludes this research venture, highlights future avenues of this work and other spin-offs of this work.

Appendices A through H supplement the information presented in various chapters and bring out a vivid picture of overall aspects that may not be under the scope of this work.

Overall, the work has been directed to make a niche in the existing research that is being carried out in assembly processes, especially in training robots to perform assembly tasks. This research has hopefully moved in a positive direction contributing to the global efforts of researchers, both, in the industry as well as in universities all over the world, working on this problem.

Chapter 1

Introduction

In recent years, the pace of growth of robotics has considerably broadened the field of robot application. Due to their improved accuracy, speed and stability, robots are no longer limited to simple pick and place operations, welding and painting. Robots have made niches in space, telerobotics and several other areas. Despite this evolution, there are some areas, potentially interesting for robot applications, where present day robots and robot systems need significant improvements. Operations like automatic assembly in space / manufacturing, grinding, deburring require sensory feedback from the process in order to be successful.

In many automated processes and space applications, the robot is required to interact with its environment in a more controlled manner, while that environment is imprecisely known.

In mechanical assembly or micro gravity operations, for instance, the manipulator must mate parts whose position, orientation, size and shape are somewhat uncertain [Onda.et.al., 1986]. In this operation, the objective is combining a number of individual parts into one completed device. The parts must be quickly assembled while avoiding damage to either the individual parts or the device. The very nature of the assembly process requires the robot (through the part) to contact the environment (the partially complete device). To successfully complete this constrained manoeuvre, the manipulator must develop compliant motion where the interaction force / torque along the constrained direction is accommodated rather than resisted [Xu.et.al., 1990]. Fig. 1.1 shows various sensors and a compliance device located on a robot.

In fact, many science and engineering applications require hardware and software systems that acquire, process and integrate information gathered by various knowledge sources [Waibel.et.al., 1991]. A typical example is that of a robot deployed in space or on the manufacturing floor using several sensors to perform a variety of inspection and manipulation tasks. For most of these systems, information made available by the knowledge sources is incomplete, inconsistent or imprecise. This information is generally modelled within the framework of probability, evidence or possibility theories. A crucial element in achieving autonomy and efficiency for these systems is the availability of a mechanism that can model, fuse and interpret this information for knowledge assimilation and decision making. The fused data reflects not only information given by each knowledge source, but also information that cannot be inferred by either source acting alone.

In addition to these, the force sensors and Passive Compliance Devices (PCD) used on the robot are limited by their range of force sensing, manoeuvrability and flexibility. Available force sensors are not universally applicable, they are large in size and heavy. PCDs

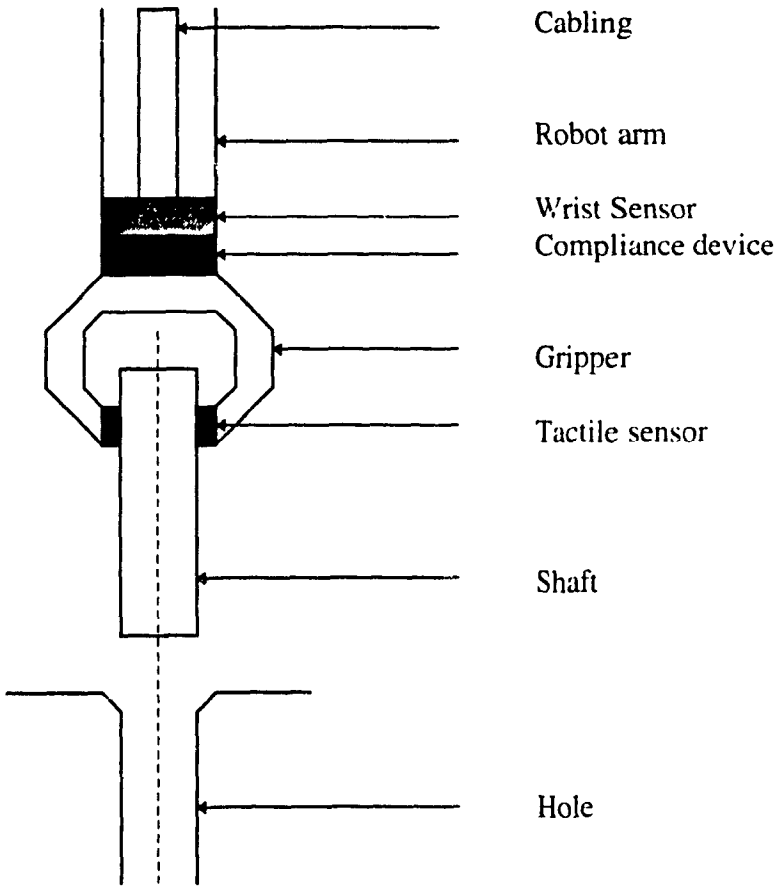


Fig. 1.1 Sensor integration - Compliance Device and Sensors

available in the market have limited range in misalignment and cocking and are expensive, as such their use is again limited. Fig. 1.2 shows a PCD placed in between the end effector and robot arm, carrying out an assembly operation.

Due to these drawbacks, industrial robots are currently used for repetitive tasks that generally do not require contact trajectories or force control. In spite of the fact that robots are flexible machines and are designed for different applications, their range of application remains limited. Apart from proper modelling, fusion and interpretation of knowledge,

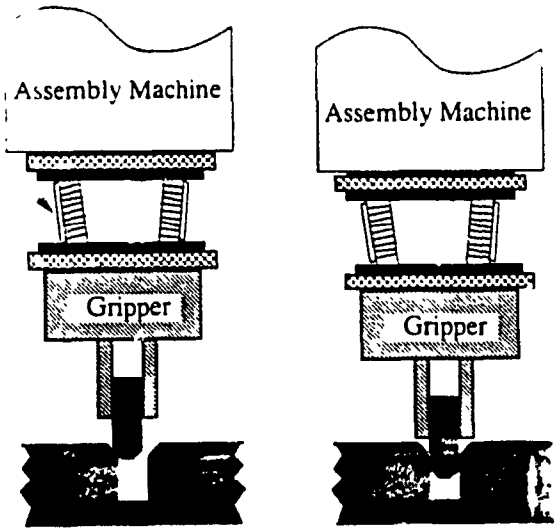


Fig. 1.2 Passive Device aligns part into hole [ATI Inc., 1985]

other factors that limit robot applications are [Reboulet.et.al.,1985]:

* Operator skill and time required for programming of even simple tasks (whether be on-line or off-line)
* Availability of only simple, low-level programming languages which are based solely on positional data (as opposed to high-level, task oriented languages)

* Failure of conventional robot programming techniques to account for the kinematics and dynamic limitations of the robot (joint over travel, singularities, linear speed and acceleration)

* Limitations within contemporary robot controllers with respect to the acquisition and integration of process variables (i.e., external sensors)

* Set-up and misalignment problems for repetitive, precision tasks.

A comprehensive solution to these shortcomings can be achieved through the integration of `intelligent sensors' with a high level, task oriented man-machine interface. Further. to overcome these shortcomings, training robots to perform tasks can broaden the horizon of robot application.

Experienced workers perform tasks with skills, while today's robots are lacking such skills. The robots require task-specific, detailed instructions substantially different from what is usually designated as skill and dexterity. Skill is the ability to use knowledge in performing and executing learned physical tasks. Rather than giving the robot instructions for performing a specific task, it should be taught skills that will enable it to exhibit dexterity and intelligence [Wolffenbuttel 1990].

In order to teach the robot these tasks, a method has been developed that elucidates the manner in which a human expert associates task characteristics with task strategies. This research is restricted to assembly tasks, focusing on splined shafts assembly into holes. The

task characteristics have been defined by termination conditions and a Common Assembly Path (CAP). The robot has been taught to carry out assembly using machine learning.

Broadly speaking, the assembly process is characterized by its in-born variability which is inherited from variation in the component parts, assembly operations and the equipment used. It consists of four main activities [Dore.et.al., 1991] :

* the gathering and organizing of piece part components;
* the mating of these components to form an assembled product;
* the fastening of these components together, and
* the transfer of the products to appropriate storage areas.

Mechanically-assisted, single or multi-station manual operation is by far the most popular method of assembly. It supplements human dexterity with simple, dedicated toolings to provide a system which is flexible and versatile. However, continuous rise in labor costs and demand for high-quality products have necessitated the development of special-purpose assembly machines consisting of indexing or free-transfer mechanisms, dedicated workheads and mechanized material handling equipment. These machines are built for a unique product. They are usually cam or pneumatically driven, accurate, consistent but expensive. Although they operate at high speeds, their overall performance depends greatly on the quality of incoming parts.

Assembly is still predominantly a manual process despite the availability of high speed dedicated assembly systems, and more recently, flexible assembly systems that incorporate robots [Hopkins.et.al., 1991]. As mentioned earlier, one major factor that inhibits the widespread adoption of automated assembly is the need for the assembly system to resolve uncertainty during the assembly process. In particular, uncertainty in the location of parts is a most demanding problem since it can affect all stages of the process.

This uncertainty arises from several sources. The parts themselves are subject to dimensional variation due to manufacturing tolerances. The feeders and fixtures are manufactured to a finite accuracy and clearances have to be allowed to accommodate the variations in the parts. Finally, there is a limit to the accuracy with which a manipulator, robotic or otherwise, can position parts. These factors result in lateral and anguiar misalignments between the parts as they are being assembled. Such misalignments can generate high contact forces that can damage the parts and, in extreme situations, prevent successful execution of the assembly task.

In an effort to accommodate these uncertainties, this research envisages the use of compliance techniques supplemented by implicit force control, to incorporate greater flexibility that can take care of uncertain misalignments that arise. Also, employing machine learning to train robots to perform assembly tasks attempts to ensure that the robot has the best moves in its database, as it repeats tasks during the learning process, especially in view of the production line realities and other aspects discussed above.

Chapter 2

Current Technology Review

This chapter includes a review of assembly strategies, available force-torque sensors, Compliance control and devices, and training techniques for robots. An attempt to broadly review the literature and to bring out advantages and shortcomings of the existing technology has been made. As mentioned earlier, there are difficulties encountered in robot control, the inherent inaccuracy and problems with force sensing, still need effective solutions. In addition, sensors and compliance devices and techniques are not universally versatile and some are still laboratory models. An overview of the major work, consolidated from the literature is presented in the following sections. Analysis of the literature and various other aspects lead to this research process.

2.1 Assembly Strategies

The age of robotics was ushered in with much excitement as engineers and scientists predicted that soon a factory could be entirely run by these flexible machines. Companies rushed to incorporate robots into their assembly lines as a means of saving on their direct labor costs in their manufacturing processes while producing consistently high quality products. Unfortunately, almost as fast as the interest in robotics surged into a frenzy it ebbed away, leaving many people disillusioned with their newly acquired robots [Susan.et.al., 1994].

However, the machine shops have reduced both direct labor cost and manufacturing time by 75 percent with the use of numerically controlled and computer numerically controlled machines [Gordon, 1987]. Researchers have estimated that assembly accounts for about 35 percent of the production cost for discreetly engineered products. Nevins and Whitney of the Charles Stark Draper laboratories have studied the science of assembly and have classified the 3 modes of assembly.

Manual assembly is appropriate for products with low production volumes. Low fixed costs are also associated with this mode so there is no economy of scale. The manual assembler has the characteristics of being very flexible and easy to train. He has excellent sensory capabilities, but may tend to lack reproducibility and get bored.

Assembly via fixed automation is appropriate for products with high volume constraints.

Fixed automation typically has high fixed costs and high efficiencies. These systems are not very flexible and tend to fail due to part jams since there is usually little sensory capability. Programmable automatic assembly has medium fixed costs and is appropriate for medium production volumes. It has medium efficiency and is capable of responding to sensory inputs and learning new tasks.

Nevins and Whitney have also studied the amount which is invested in assembly in a number of different industries. Motor vehicle and radio and television industries have about 30 percent of direct labor attributed to assembly. Boothroyd postulates that assembly accounts for about 50 percent of the total manufacturing cost for a product.

2.1.1 Assembly Task Analysis

The operations necessary to perform the assembly of some products were studied. The investigation was carried out to :

* investigate which mechanical assembly operations are prevalent in certain product types

* determine which operations can and cannot be accomplished by an unaided six degrees of freedom manipulator

* investigate the difficulty of the different operations

None of the parts were machine assembled, nor was their design optimized for ease of assembly. Ten most prevalent mechanical assembly operations were identified:

1. Unstable assembly: Any operation where a part will not maintain its proper position under the force of gravity. A plate without fasteners covering a long, thin compression spring for example.

2. Required Orientation of another part prior to assembly : Stabilizing (fixing the position) of an already assembled part prior to insertion of a new part.

3. Spring Insertion / Compression: Operations which require insertion of parts which must be mechanically stressed prior to their installation.

4. Plastic Heading : Heading of rivets and other fastening techniques requiring plastic deformation of material.

5. Unstable Inversion : Requires that a part or assembly of parts be re-oriented prior to assembly such that without constraint, they would become unstable and fall apart.

6. Retaining Clip Insertion : includes assembly of internal and external snap rings and E clips.

7. Non-screw Twisting : Includes all helical insertions which are not performed with standard screws.

8. Press Fit : Similar to unidirectional insertion (similar to item 10 below) except there is an interference fit rather than a clearance.

9. Screw Insertion : Driving of standard shaped screws only. Specially designed parts which are screwed into an assembly are not included in this classification.

10. Unidirectional Insertion : Any unidirectional insertion with a clearance fit. There is no restriction on the geometric form of the parts so long as the parts ar rigid and the insertion direction is a straight line.

As can be inferred from the discussion above, one of the most widely studied tasks in robotics is the 2D peg-in-hole task. Detailed analyses have been carried out to determine strategies that guarantee successful insertion once the peg is partly in the hole. When the initial uncertainty in position is large enough, a strategy must also be devised to ensure

that the peg can find the hole. Overall the variety of strategies can be grouped as follows [Lozano-Perez.et.al., 1984].

1. Chamfers - Chamfers on the hole entrance and/or the peg tip increase the range of relative positions where the peg can fall into the hole at least partway. This technique is especially effective if the peg support has lateral compliance [Drake, 1977; Whitney, 1982].

2. Tilting the peg - Tilting the peg slightly also increases the range of relative positions where initial entry into the hole is guaranteed [Inoue 1974]. In fact, the geometric effect of tilting the peg is almost identical to that of providing a chamfer.

3. Search - The simplest strategy is a search in which the peg slides along the top surface until it falls into the hole. In general, the search has to pick an initial direction of motion and possibly, back up if the hole is not found.

4. Biased search - A slight modification to the search strategy is to introduce a bias into the initial position of the peg [Inoue, 1974]. This strategy reduces the chances of initial entry into the hole, but guarantees that the peg will be to one side of the hole.

The approach of Tomas.et.al, is based on a different view of assembly: that is the geometric constraints should 'guide' the parts to their destination without necessarily having to know exactly where the parts are relative to each other.

One of the earliest explorations in the area of automatic synthesis of fine-motion strategies from strategy skeleton was by Taylor [1976]. Taylor developed a technique for propagating the effect of errors and uncertainties through a model of a task. These error estimates were used to make decisions for filling in the strategy skeletons. For peg-in-hole insertion, for example, the decision whether to tap the peg against the surface next to the hole was based on whether the error estimate for position normal to the surface exceeded a threshold.

Lozano-Perez [1976] also proposed a method for selecting the motion parameters in strategy skeletons. Each motion in a skeleton was specified symbolically by the relationship among parts that it was designed to achieve. The expected length of guarded moves and their force-terminating conditions were then computed from the ranges of displacements that achieved this relationship (taking into account uncertainty in position).

Recently, Brooks [1982] extended Taylor's approach by making more complete use of symbolic constraints in the error computations. The resulting constraints can be used in

the 'forward' direction to estimate errors for particular operations. But, importantly, they also may be used in the 'backward' direction to constrain the values for plan parameters, such as initial positions of objects, to those that enable the plan to succeed. When no good choice of parameters exists, the system chooses appropriate sensing operations (such as visual location of parts) that reduce the uncertainty enough to guarantee success.

Another line of research was focused on building up programs automatically from attempts by the robot to carry out the operations. Dufay and Latombe [1983] describe how partial local strategies ('rules') for a task can be assembled into a complete program by processing the execution traces of many attempts to carry out the task. The method, however, requires knowing the actual relationship between parts achieved by each motion, for example, which surfaces are in contact. This information can be obtained, in many cases, from careful analysis of the forces and positions, but, in general, the information is ambiguous in the presence of measurement and control errors. Moreover, the rules used by the system are specific to the tasks and must be provided by the users.

A related approach to deriving a strategy from 'experiments' is based on the theory of stochastic automata [Simons.et.al., 1982]. The goal is to have the robot learn the appropriate control response to measured force vectors during task execution. The method requires a task-dependent evaluation function so as to judge progress toward its goal.

These previous approaches to fine-motion synthesis are based on the assumption that there is a basic repertoire of operations, such as peg-in-hole insertion and block-in-corner placement, whose geometric structure is known a-priori. In this way, the task of a synthesis program is to make some pre-defined set of choices among alternative actions, select the order of operations In fact, small changes in the geometry of parts can have significant impact on fine motion strategies.

Different operations in peg-in-hole require substantially different programs to ensure reliable execution. Similarly, differences in expected position errors call for different strategies for the same task.

Since the early 1960s researchers have analyzed the peg-in-hole assembly task in detail, Laktionev and Andreev in1966, Andreev and Laktionev in 1969, Gusev in 1969, Drake in 1977, Ohwovoriole, Roth and Hill in 1980, Ohwovoriole and Roth in 1981 and Whitney in 1982. In most of the analyses, the assumption is that the peg is initially partly in the hole, possibly at a chamfer. Two important types of insertion failure have been identified : *jamming* and *wedging*. Jamming is due to misproportioned applied forces; wedging is due to geometric conditions that arise when parts are slightly deformed. These analyses have led to the formulation of conditions for successful insertion forces applied to positions of the peg and hole. A number of heuristic strategies for peg-in-hole insertion have also been formulated, based on more fragmentary analysis. These heuristic strategies

have been used successfully to an extent [Inoue, 1974; Goto, Takeyasu and Inoyama, 1980].

Mason's [1982] detailed analysis of pushing and grasping operations in the presence of friction also leads to conditions for successful task completion. These conditions provide the basis for synthesis of operations that succeed in the presence of uncertainty (without requiring sensing).

Simunovic [1979] formulated the information approach to fine motion, based on the principle that assembly is purely a relative positioning task. From this premise, he argues that the role of an assembly program is to determine the relative positions of parts during an assembly and to issue position commands to correct errors. He developed an estimation technique to infer, from a series of noisy position measurements and knowledge of the geometry of the parts, the actual relative positions of the parts. One problem with this approach is that it requires a very large amount of on-line computation, although this could be solved using special-purpose electronics. A more fundamental problem is that the approach assumes only position control and a robot capable of making fine incremental motions. This need not be the case for assembly; by exploiting compliant behavior, the robot can perform high-accuracy tasks even with low-accuracy position control, for example, the task of following a surface by maintaining a downward force. Another problem is that this technique requires knowing which surfaces are in contact.

This limits the method to situations with relatively small errors; in more general cases, the identity of the contact surfaces is not known.

Assembly of parts with irregular surfaces using active force sensing has been carried out by Lee.et.al [1994], work on a precise robotic chamferless peg-hole insertion operation without force feedback and remote center compliance has been done by Qiao.et.al [1994]. Mimura.et.al [1994] have done parameter identification of contact conditions by active force sensing and Leysen.et.al [1992] have evolved an augmented task level programming system for force controlled assembly operations. Most of these research ventures resulted in a variety of algorithms suited for specific tasks and have produced encouraging results.

2.2 Robotic Wrist Sensors

Research on wrist sensors has been carried out since the late 1970s and various approaches have been attempted. Most sensor designs have not been successful in industry due to their large size, cost, complex controllers, specific applications and some are still laboratory models.

Van Brussel and Simons [1979] have designed and constructed a five DOF wrist (no axial rotation) that contains both sensing and direct position control. Each wrist axis is driven by a DC motor via a soft servo loop. Servo gain and torque saturation levels are programmable and thus axes have an automatically adjustable 'equivalent spring' stiffness. This emphasis

on wrist control eliminates the need for complex arm control algorithms and it avoids ambiguous position situations arising from arm joint manipulation. The ability to adjust the stiffness for a given task is a further advantage. Obviously the cost and size of this wrist unit are large.

Cutosky and Wright [1982] have introduced a design that is a development of the Instrumented RCC concept. The new feature is that a range of compliant control is available in one unit. The design presented adds mechanical complexity, but in some situations aids assembly by reducing the need for complex software. The 'compliance-range' is achieved by introducing stiff elastomeric spheres between the critical wrist components. The philosophy of the design is to mimic the muscle-control of the human forearm when engaged in tasks that require a range of accuracy and strength. In this case, the design is complex and the size of the wrist is large - 254 mm in diameter - thus its versatility in various applications may be difficult.

Hollis.et.al [1993] describe a high performance 6 DOF magnetically levitated fine motion wrist with programmable compliance. The stiction effects have been eliminated by 'floating' the wrist's moving part using active magnetic levitation. The authors claim that with its frictionless magnetic suspension, extremely high acceleration, precise positioning capability and programmable compliance, the wrist should find applications in automatic assembly and testing micro electronics. But the authors mention that a robot mountable

version of the prototype wrist, with a number of design improvements is still under development. The actuators are large, and controller design is complex. A prototype wrist sensor, 'Maglev', has been developed for tele-operation by a firm in Montreal on the basis of the design by Salcudean. S, which has been performing well. Cost of the prototype is quite high, and the estimated production cost is more than $ 10,000 CND.

Xu.et.al [1992] have developed a compliant wrist combining passive compliance and a displacement sensor. The wrist provides the necessary flexibility to accommodate transition as the robot makes contact with the workpiece, to correct positioning error and to avoid high impact forces in automatic assembly. Sensory information from the device makes it possible to actively control the contact forces or to compensate for the positioning error during motion and contact. It is known that passive compliance could degrade the positioning capability of robots, in addition, the use of rubber could introduce hysteresis problems, especially when the wrist is used in fast repetitive operations.

Lindsay.et.al [1993] present a design and implementation of an instrumented compliant wrist device that serves both as a passive compliance and an active sensing mechanism. The compliance helps reduce the impact effects of robot / environment interaction and improves force control performance. However, positioning accuracy of the end-effector degrades with increased compliance. Instrumentation of the compliant wrist enables it to serve as a compliant force / torque sensor which can be used to achieve both responsive force control

and accurate position control. Wrist devices use rubber elements for compliance and damping and a serial linkage, with potentiometer at each joint, is used for sensing deflections produced in the wrist. This design has similar drawbacks as that of Xu.

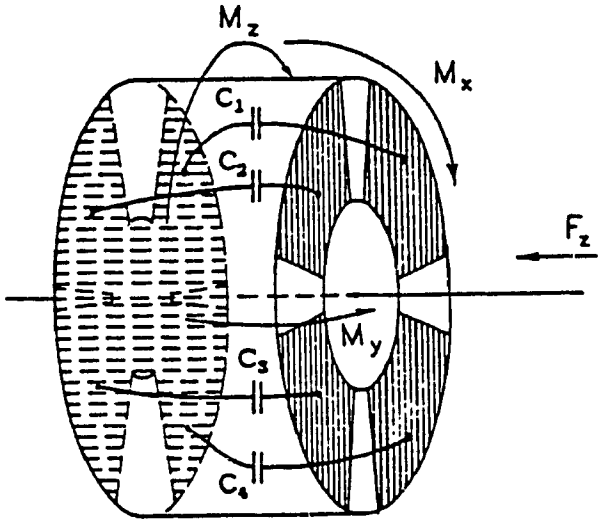


Fig. 2.1 Electrode pattern of capacitive sensor [Wolffenbuttel.et.al., 1990]

Wolffenbuttel.et.al [1990] designed a sensor to measure the bending moment in the X and Y directions, force in the Z direction and the torsion moment about Z axis. The sensor consists of 2 opposite electrode patterns, Fig. 2.1 with an elastomeric material in between and electrical contacts only to the arm-side electrode pattern. Applying a force changes capacitance due to deformation of the compliant intermediate. The four sensitive capacitance patterns between the two electrodes give different values for forces - moments

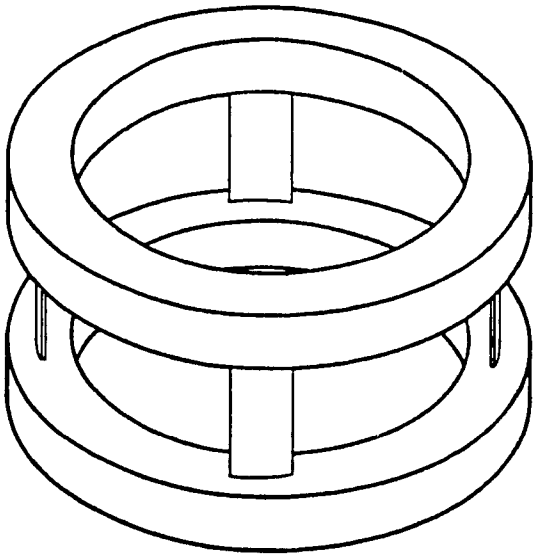


Fig. 2.2 Cage type sensor [Watson.et.al]

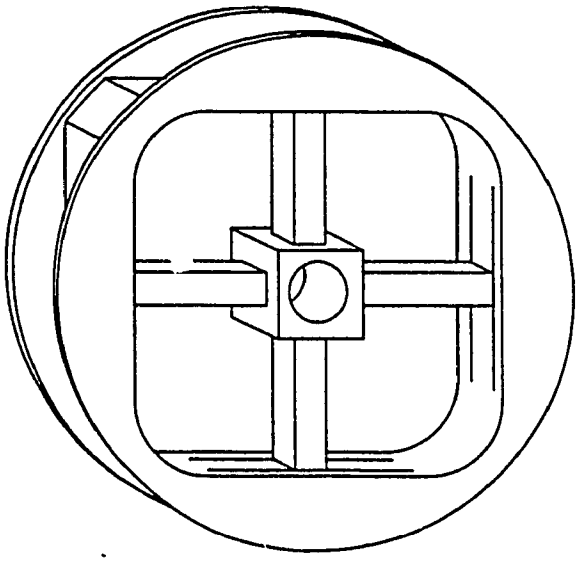


Fig. 2.3 Schematic of sensor developed for JPL [Bejczy.et.al]

in different directions. The drawback of this design is that a change in di-electric medium changes the behaviour of the sensor.

2.3 Compliance Control and Devices

Before looking at the compliance devices it is necessary to introduce some concepts of assembly and task planning. A brief outline of Gross motion and Fine motion planning will be discussed.

Tracing the evolution of robotics illustrates the fact that the pace of robotic applications did not match the demands of industry. There were several factors responsible for this gap, although robots were capable of performing a broad range of tasks. In order to study this cause, the field of assembly and task planning arose to address the issue of how a detailed robotic operation plan could be automatically synthesized given a high level description of a product to be assembled [Susan et al., 1994]. *Assembly Planning* (AP) is generally considered to be the process of determining a set of instructions for mechanically assembling a product from a set of components. Each instruction usually specifies that a component be added onto the assembly partially in a particular way. AP does not demand that the assembly be carried out by a robot; however, if it is, then additional interpretation is usually needed to map these instructions onto robot operations. This translation is called *Task Planning* (TP) and it may also be required in robot activities other than assembly.

The reason for studying AP is twofold: a large number of the potential applications for robots involve assembly-like tasks, and the assembly domain provides researchers with a rich set of problems to consider which are suitably complex but not unsolvable. Much of what is learned in studying the specific field of assembly planning can be generalized into non-assembly oriented tasks. On the other hand, the study of task planning is not specialized into a single application such as assembly and thus complements the work performed in assembly planning.

Gross Motion Planning (GMP) also referred to as path planning, is necessary to plan a path for bringing the end-effector to an assembly component so that it may be grasped, for bringing the assembly component to the first so the two may be assembled. Uncertainties are generally ignored in GMP because they are considered to be small relative to the clearances allowed between the objects in the work cell.

Fine Motion Planning (FMP) also known as Compliant Motion, is carried out when the clearances allowed between parts in an assembly are small relative to the uncertainties. When this is the case, it is necessary to develop a fine motion plan for assembling the parts. FMP relies on motion with sensory feedback, in particular force and torque feedback, to overcome the uncertainties as the assembly operation proceeds.

In effect, the nature of interaction between a robot and its environment can be categorized in two classes. The first one concerns the non-contact (eg., unconstrained) motion in a free work space, without any relevant environmental influence exerted on the robot. In non-contact tasks, the robot's own dynamics have a critical influence upon its performance. In a contact task, there are phases where the robot's end effector must come into contact with objects in its environment, produce certain forces upon them, and move along their surfaces. Inherently, each manipulation task requires contact with the object being manipulated. Motion through unstructured, insufficiently known work space also is a contact type task. Contact tasks are characterized by dynamic interaction between the robot and the environment, which often cannot be predicted accurately [Miomir.et.al., 1994].

For successful completion of contact tasks, either the interaction forces have to be monitored and controlled, or control concepts ensuring compliant interaction with the environment must be applied. Compliance can be considered as a measure of the ability of a manipulator to react based on forces resulting from the interaction with the environment. The increased demand for advanced robotic applications has brought about an enormous interest in the development of different concepts and schemes for the control of compliant motion.

In summary, the type of contact tasks may vary substantially for specific requirements, but in all cases, the robot has to perform three kinds of motion :

gross motion	- related to robot movement in free space
compliant or fine motion	- related to robot movement constrained by an environment
interface or approach motion	- representing all transitions between gross and compliant motion

The methods of control for robots in the constrained motion tasks can be classified according to different criteria. Based on the compliance, there are two basic groups.

1. Passive Compliance - where by the robot's real position is approaching the desired position only by influence of the contact forces themselves.

2. Active Compliance - where by the compliance is provided by using force feedback in order to achieve either control of the interaction force or a task-specific compliance of the robot's end effector.

Passive compliance can be further classified into two groups, based on their source.

1. Non-adaptable methods

- methods based on the inherent compliance of the robot mechanical structure.

- methods that use specially constructed passive deformable devices attached to the robot end effector.

2. Adaptable methods

- methods based on devices with tunable compliance

- methods based on compliance achieved by the adjustment of the joint servo gains.

Active control methods may be classified further into the following two groups.

1. Hybrid position / force control, whereby both position and force are controlled in a non-conflicting way in two orthogonal sub-spaces defined in a task specific frame.

1a). explicit or force-based methods where force signals are used to generate the torque inputs for the actuators in the robot's joints.

1b). implicit or position based algorithms, whereby the force control error is first converted to an appropriate robot's motion adjustment in the force-

controlled directions and then that position is used as input in the position controller.

2. Impedance control, which is in essence based only on the position control and uses different relationships between the acting forces and manipulator's position.

2.3.1 Passive Compliance

With this technique, instead of rigidly locating the parts, a degree of compliance is introduced into the assembly system which deforms under the influence of the assembly forces, thus reducing the misalignment [Hopkins.et.al., 1991].

Early devices developed to assist the alignment of components used 'floating heads' or vibrated one of the components relative to the other. Other methods of vibrating the part held by the robot have involved superimposing a sinusoid on the joint controller input signals, or using a vibratory gripper. Other devices have employed air jets and vacuum techniques, but the first commercially available passive device for reducing misalignment was the Remote Centre Compliance (RCC) [Watson, 1978].

The RCC is a mechanical structure which deflects under the contact forces encountered during assembly. It is designed in such a way that lateral forces applied at the remote

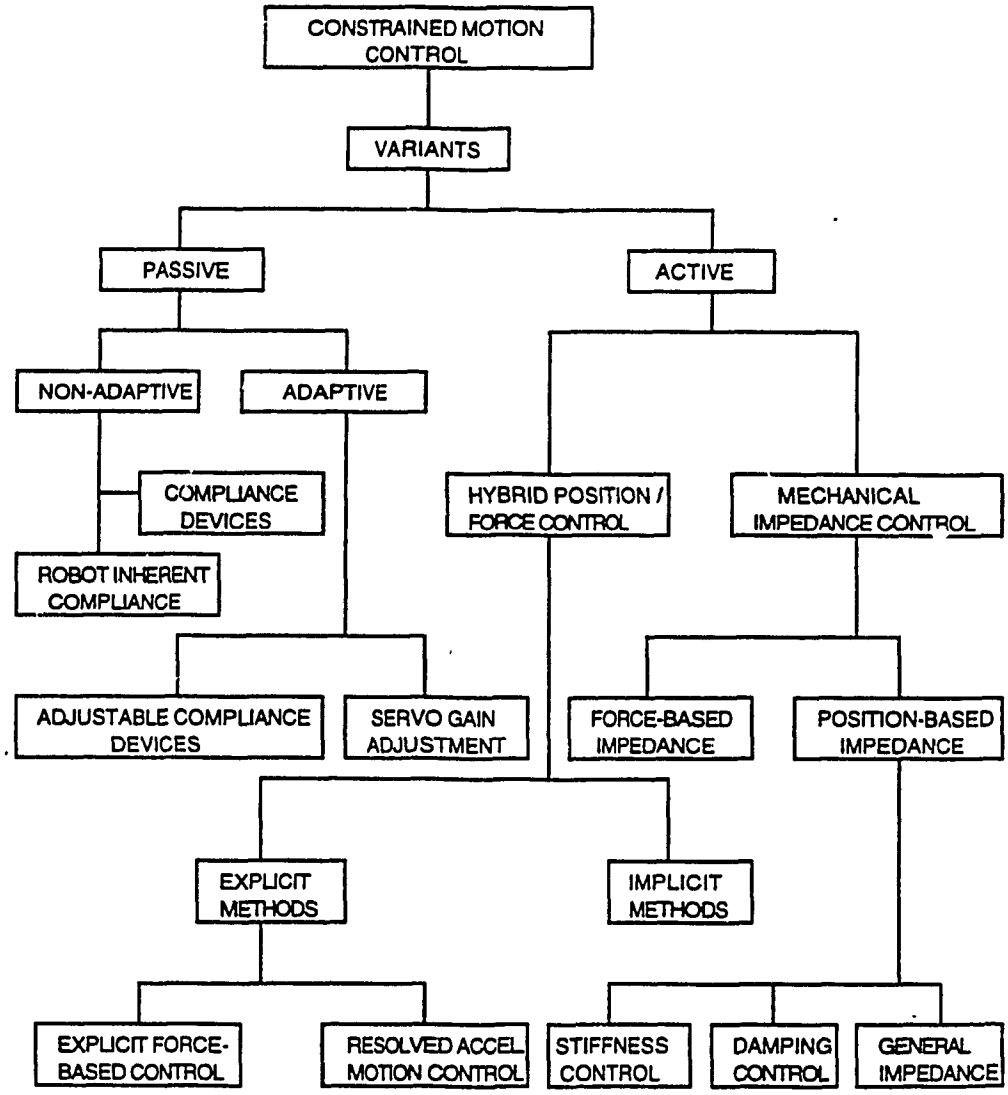


Chart I Different Methods for Constrained Motion Control [Miomir, 1994]

centre produce only lateral deflections, while applied moments about the remote centre produce only angular deflections. Thus, if the RCC is positioned behind the gripper of the manipulator so that the tip of the component is located at the remote centre, the RCC deforms in such a way that the problems of wedging and jamming are minimized. Although this device has been successfully employed in a variety of applications, it does have its limitations. The parts must be chamfered and the RCC has to be designed to suit the geometry of the parts.

Drake [1977] developed a device that could mimic an active force feedback system. He also showed that forces during assembly are a function of initial positional errors, coefficient of friction between pieces and the value of the compliance matrix at the assembly interface. This technique is based on designing a passive compliance structure or mechanism as in Fig. 2.4.

Researchers have developed passive grippers [Pham.et.al.,1991]. These grippers automatically conform to the shape of the workpiece by means of gripping elements which are elastic or have passive (i.e., non-actuated) degrees of freedom. Several passive grippers have been developed, including those based on the fluid-solid phase-changing 'bag of beans', the matrix of retractable pins, the bourdon tube, and the differential mechanism. Generally, with passive gripping, it is difficult to ensure precise positioning of the gripped workpiece with the robot's coordinate system. Hence, the technique is

mainly used for simple pick-and-place applications where positional accuracy is not required.

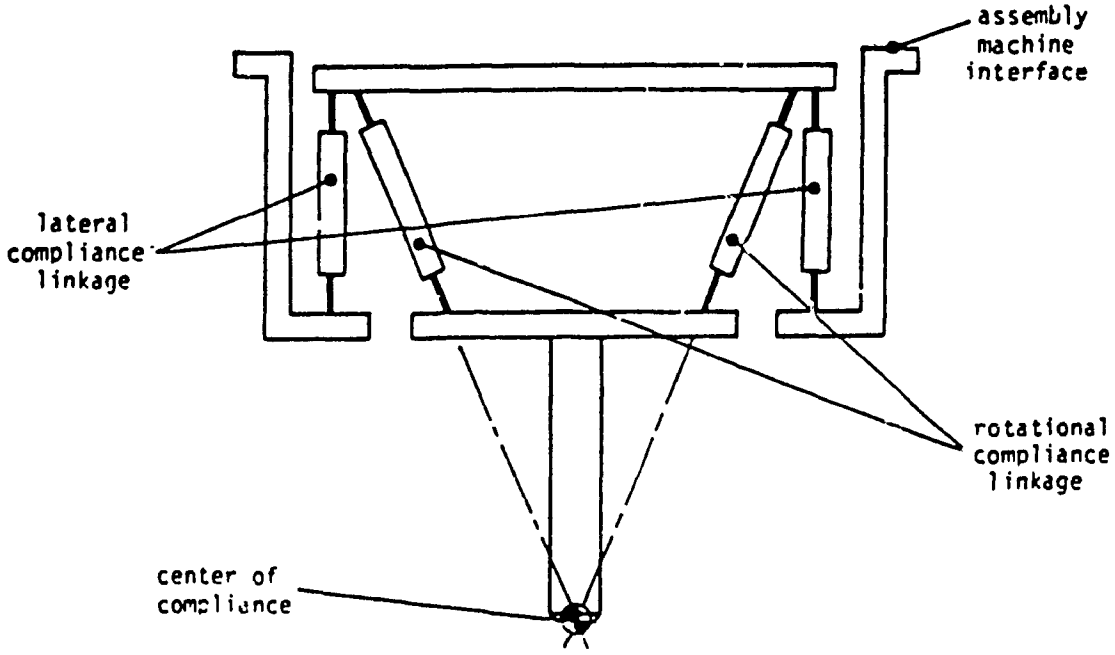


Fig. 2.4 Schematic of Passive Compliance Device [Drake, '77]

2.3.2 Active Compliance

To overcome the limitations of passive devices, active devices have been developed in which the contact forces are measured by transducers and the signals produced are used to actively control actuators which compensate for the misalignment.

One method of producing a force sensor is to measure the deflection of a passive device. In the simplest form, micro-switches have been used to detect when the deformation of a

passive device has exceeded a pre-set thresold [Williams et al., 1985]. Alternatively the passive device can be equipped with continuous displacement sensors; for example, optical sensors have been used to measure the deflection of the RCC [De Fazio et al., 1984].

A variety of six component force sensors have been developed using strain gages to measure the deflection of a mechanical structure [Van Brussel et al., 1985], although other types of transducers have also been used. Other researchers have reduced the complexity of the sensor by measuring only three force components and where necessary, compensating for loss of information by intelligent signal processing [Spalding, 1982; Bland et al., 1986]. Often the sensing is performed at the wrist of the manipulator, immediately behind the gripper; alternative positions have also been investigated, for example, in the fingers of the gripper [Thorton and Smith, 1987; Lestelle, 1985], below the assembly fixture [Kasai et al., 1981] or at the robot joints [Wu, 1985].

In addition to sensing, active accommodation must incorporate some means of actuation to provide the compensatory motion needed to reduce the misalignment. The simplest means of achieving this is to use the sensor signals to control the robot [Hirzinger 1986]. This approach has its drawbacks. First, it is difficult to achieve the fine control needed due to high inertia of the manipulator arm. Secondly, component misalignment is a problem not only in robotic systems but in classical hard automation where there are no

servo-controlled axes which can be adaptively controlled. Thus, force sensing alone is not the complete solution and systems need to be developed which combine both sensing and micro manipulation.

2.4 Robot Training Techniques

Techniques for robot learning and transferring manipulative skills have been reported in the literature. Some of these are specific to the area of machine learning, others are in the area of transferring human skills to robots. Machine learning has made its debut less than a decade ago in the field of assembly strategies, therefore a large amount of work still remains to be done in this area. Work related to machine learning, available in the literature is still in the kick-off stage.

2.4.1 Using Machine Learning

Fuzzy pattern matching has been used by Kei.et.al [1992] for teaching robots. This work deals with precision insertion for chamferless parts with uncertain positional information by heuristic search. The heuristic search is based on fuzzy pattern matching between fuzzy sets of search areas and additional sets of the hole center positions. Fuzzy pattern matching evaluates the value of the heuristic function. The search of the hole is started in the search area having the highest value of the heuristic function. RCC (remote center compliance) and a force sensor are used in the search. RCC finely modifies the position of the peg and the force sensor detects whether the insertion has succeeded.

Andrew.et.al [1992] discuss the use of pattern-recognizing stochastic learning automata. A class of learning tasks is described in this research that combines aspects of learning automation tasks and supervised learning pattern-classification tasks. These tasks are called the associative reinforcement learning tasks. An algorithm is presented, called Associative Reward-Penalty algorithm, for which a form of optimal performance is proved. This algorithm simultaneously generalizes a class of stochastic learning automata and a class of supervised learning pattern-classification methods related to the Robbins-Monro stochastic approximation procedure. The relevance of this hybrid algorithm is discussed with respect to the collective behavior of learning automata and the behavior of networks of pattern-classifying adaptive elements.

Admittance control has been employed in the learning process by Gullapalli.et.al [1992]. A peg-in-hole insertion task is used as an example to illustrate the utility of direct associative reinforcement learning methods for learning control under real-world conditions of uncertainty and noise. An associative reinforcement learning system needs to learn appropriate actions in various situations through search guided by evaluative performance feedback. Such a learning system was used, implemented as a connectionist network, to learn active compliant control for peg-in-hole insertion. These researchers indicate that direct reinforcement learning can be used to learn a reactive control strategy that works well even in the presence of a high degree of noise and uncertainty.

Vaaler [1991] in his research uses machine learning to train robots for assembly tasks. The model derives a relationship between the force information and the robot position, and the best moves are stored as the task is performed over and over again. The robot learns from this mapping and performs at a faster rate with progressing time. Logic branching has been employed in this research and it has proved to be quite successful in the training process.

Ahn.et.al [1992], split the assembly task into searching tasks and insertion tasks. In the search task, the learning is iterative and force-moment information is related to different actions. In the insertion task, learning is achieved by mapping the actions along with the forces monitored. The researchers have carried out a series of experiments and have demonstrated that their strategy works well.

Techniques pertaining to transferring human skills to robots have been studied as well. MIT is the pioneer in this area of research, as noted from the literature.

2.4.2 Transferring Human Skills

Asada and Izumi [1987] developed a methodology for the automatic generation of robot programs for hybrid position / force control. This uses a direct teaching and automatic program generation method which eliminates manual programming and task interpretation / translation. The operator teaches a given task by 'teaching-by-showing', in which the operator contacts the robot end effector to the environment and accommodates

the control force. During the operator's motion, the force applied by the operator as well as the position of the end-effector are measured. The acquired motion data are then processed and interpreted so that necessary information to generate robot programs is obtained, Fig. 2.5. The choice of control modes as well as reference inputs to the robot controller are derived from motion data, this result is translated into a robot program.

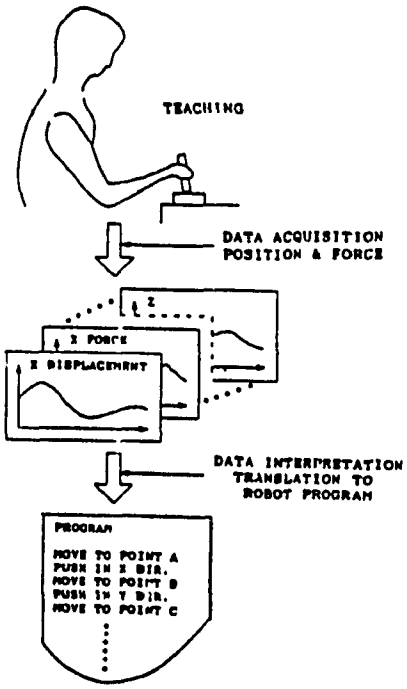


Fig. 2.5 Automatic program from motion data [Asada-Izumi, 1987]

Eric Aboaf.et.al [1989] report a preliminary task level robot learning approach, an approach to learning from practice. They programmed a robot to juggle a single ball in 3 dimensions by batting it upwards with a large paddle. The robot uses a real-time binary

vision system to track the ball and measure its performance. Task-level learning consists of building a model of performance errors at the task level during practice and using that model to refine task level commands. The authors claim that application of task-level learning dramatically increased the number of consecutive hits the robot could execute before the ball was hit out of range of the paddle.

Asada and Asari [1988] developed another direct teaching method of tool manipulation skills via impedance modification of human motions. First a skilled worker shows how to perform a task and his motions are measured, Fig. 2.6. Specifically, the force exerted by the worker and the displacement of the tool manipulated by the worker are monitored and stored in the computer. The data are then analyzed in order to find the control law of the human expert. The functional relationship between the force and the displacement is derived from the data by using a curve fitting technique. The identified relationship is then used as a reference model for controlling a manipulator arm to replicate the expert motion. The authors have performed this technique on grinding operations and presented the results.

Yang and Asada [1989] developed and applied another method for teaching human expertise to robots. Human skills and knowledge of performing a given task are transferred to robots through the acquisition and interpretation of human linguistic information along with demonstration data, Fig. 2.7. Based on linguistic information, a

non linear control structure is obtained which describe the expertise. The relationship between each linguistic label and the corresponding sensor signals is determined by matching demonstration data and linguistic data in order to interpret linguistic information in the sensor space. For each partitioned space, a piecewise linear control law for modifying control actions is obtained from the demonstration data. The authors have built a high level controller for robotic deburring in which a robot can change its trajectory and accommodate its tool holding compliance by recognizing changes in the process, like burr size and hardness.

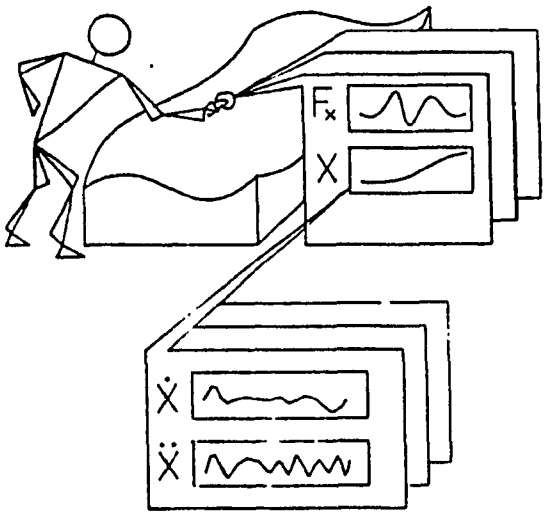


Fig. 2.6 Measurement of hand motion [Asada-Asari, 1988]

The authors have developed techniques for selecting significant features extracted from sensor signals and reducing the dimensions of the sensor space.

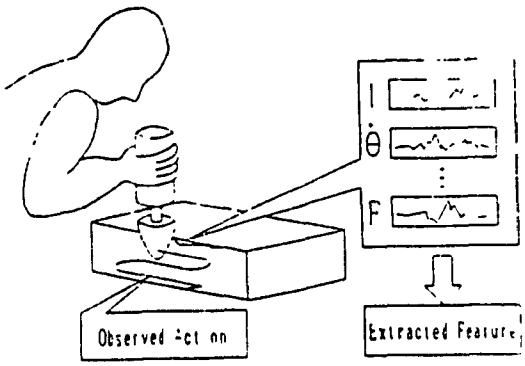


Fig. 2.7 Acquisition of teaching data from expert [Yang, 1989]

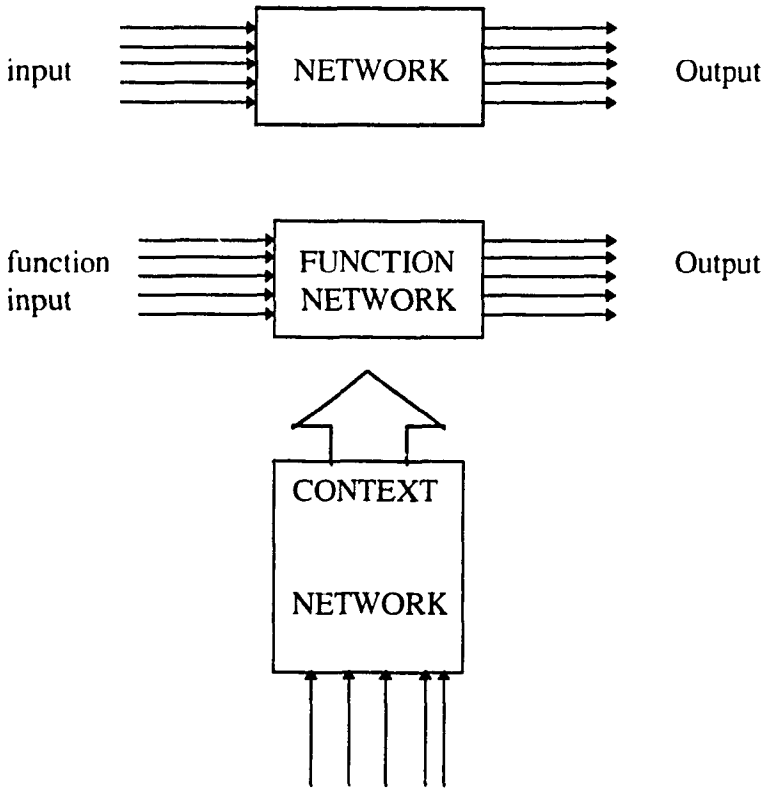


Fig. 2.8 Standard network versus Context sensitive network [Yeung.et.al,89]

Yeung and Gekey [1989] propose and use a context sensitive learning network for robot arm control in the learning of complex non-linear mappings. The authors claim that this class of learning networks have better scale-up properties. In this method they propose to partition the set of input variables into two sets, Fig. 2.8, one set (convex input) acts as the input to a context network, the output of which is used to set up the weights in a function network. The function network maps the second set of input variables (function input) to the output. Depending on the context input provided by the current input pattern, the function network represents different functions at different times. The function network could be thought of as being 'programmed' by the context network, and hence the two networks belong to different levels of abstraction.

2.5 Design for Assembly (DFA)

Design for assembly [Geoffrey Broothroyd, 1991], is an attempt to design the parts so that they can be assembled in the presence of system position errors as well as errors in the parts themselves. Researchers have developed various methodologies for faster and easier assembly. DFA is important because the design of a part can have a strong impact on the success of a particular assembly strategy. Chamfers are a very successful DFA technique and are the subject of considerable research [Whitney, 1982; Caine, 1985].

2.6 Selective Assembly

A fairly sophisticated automatic inspection system employing a robot to dimensionally and functionally inspect precision parts such as automobile crankshafts or injectors was developed [Camera, 1981]. Instrument grippers have been used for some gross dimensional inspection. After being inspected, the parts are placed in tolerance groups for later insertion into other parts belonging to appropriately matched groups.

To summarize, an overview has been attempted to present various wrist sensors, compliance techniques, both, passive and active and the robot training techniques. Hardware available in the industry and academia, has been presented and for the robot training techniques, most of them are in the laboratory stage of development but have been presented for an insight to the reader.

Chapter 3

Proposed Work

In recent years robots have been deployed in several domains. Due to their better accuracy, speed and stability, robots are no longer limited to simple pick and place operations, welding and painting. This trend is demonstrated in Fig. 3.1, which shows projected US robot sales by the year 1996 [Assembly, 1994].

In many automated processes, the robot is required to interact with its environment in a more controlled manner, while that environment is imprecisely known. In mechanical assembly, for instance, the manipulator must mate parts whose position, orientation, shape and size are somewhat uncertain [Whitney, 1982]. In this operation, the objective is combining a number of individual parts into one completed device. The parts must be quickly assembled avoiding damage to either the individual parts or the device.

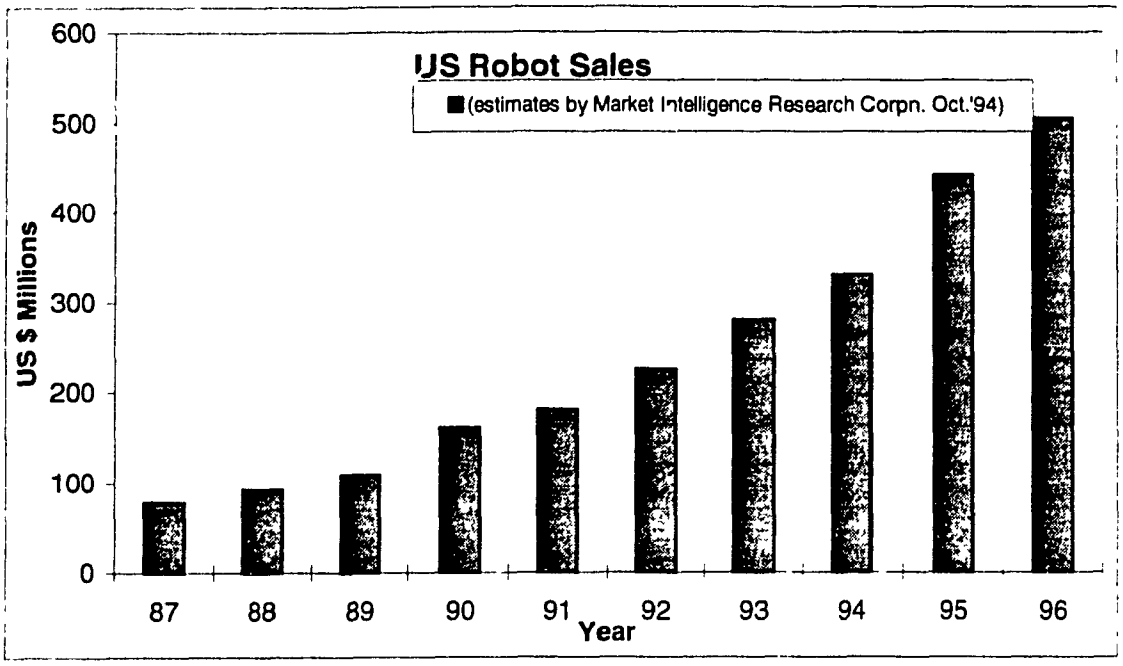


Fig. 3.1 US Robot Sales [Assembly, 1994]

Robotic assembly cells capable of assembling parts with high accuracies are far too expensive to compete with human assembly workers. Possible solutions to this problem include: improving the cost to performance ratio of the robot, redesigning the parts so that they can be assembled more easily, or installing sensors on the robot and fixtures to gain access to information about the assembly that can be used to correct for non-ideal behaviour of the robot.

3.1 Goals

Principal goal of this work was to explore the feasibilty of splined shaft and hole assembly using machine learning to learn the correct responses to contact forces encountered during assembly. There were other secondary goals of this work, as listed below.

i - Design, develop and manufacture a Passive Compliance Device, that has greater versatility and increased misalignment range in the vertical and angular axes.

ii - Manufacture and calibrate a Force-Torque sensor for the robot, based on a prototype that was already developed in the Laboratorie d'Automatique et de Mecatronique.

iii - Develop a cost effective solution for compliance control, as such use Hybrid Compliance Control for the assembly task, with random initialisation of algorithm.

iv - Determine the practical limits of accuracy of the mapping from contact forces to relative part positions

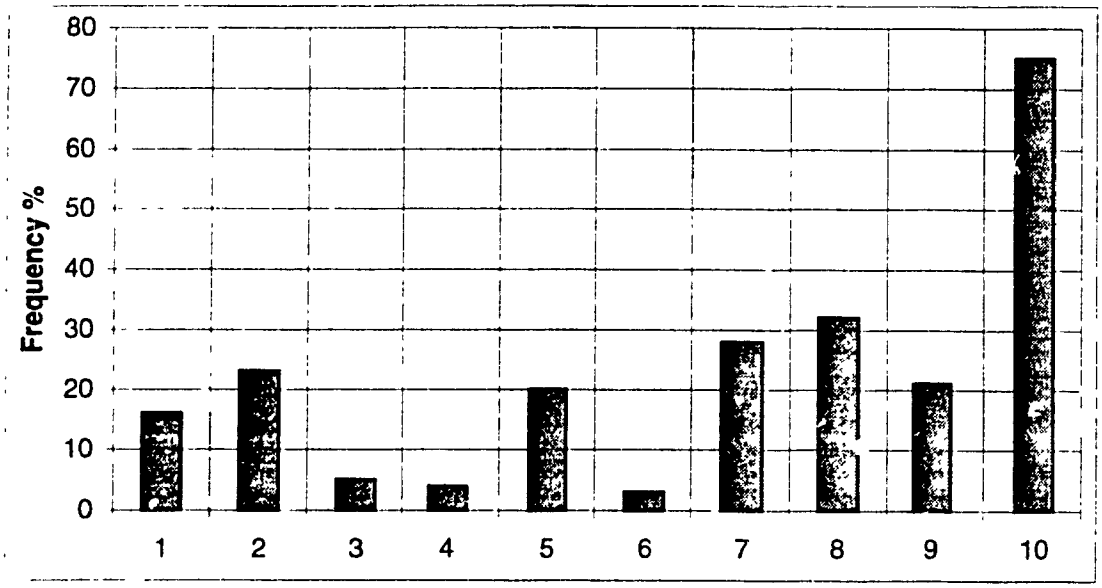
v - Use a combined strategy of Learning by Examples and by Induction in the machine learning algorithm, and finally

vi - Reduce costs in implementing these features in a 'shop floor' environment.

3.2 Motivation of Work

For several companies, assembly constitutes more than 50 percent of the cost of manufacturing a product. For complex machines manufactured in small quantities, percentage costs are much higher. Reducing the costs of assembly, for small and medium sized companies, is one of the motivations of this research work. An increasing need for work in an unstructured and hazardous environment, especially nuclear plants, offshore platforms, under sea pipe lines, and space stations, has resulted in an emphasis on research in this domain.

Researchers [Gordon, 1987] , have established that in an assembly task, 33 percent of the operations are peg in hole, 27 percent are screw insertions, 12 percent are push and turn, less than 10 percent are multiple peg-hole insertions, force fits and another 10 percent constitutes supporting parts, and removing locating pins. Fig. 3.2 highlights the frequency of



1 Unstable Assembly
2 Orient another part
3. Clip Insertion
4. Compress Spring
5. Plastically deform
6. Unstable inversion
7.Non-Standard screwing
8. Press fit
9. Standard Screwing
10. Unidirectional screwing

Fig. 3.2 Frequency of operations [Gordon, 1987]

operations for several automotive and consumer electro-mechanical processes [Gordon 1987]. A study showed that at General Motors, 90 percent of parts in an average automobile weigh less than 2 kilograms and at John Deere, 80 percent of parts in their farm equipment are less than 4 kilograms. This is an indication that robotic assembly can be implemented in a much more elaborate manner than is presently the case in industry.

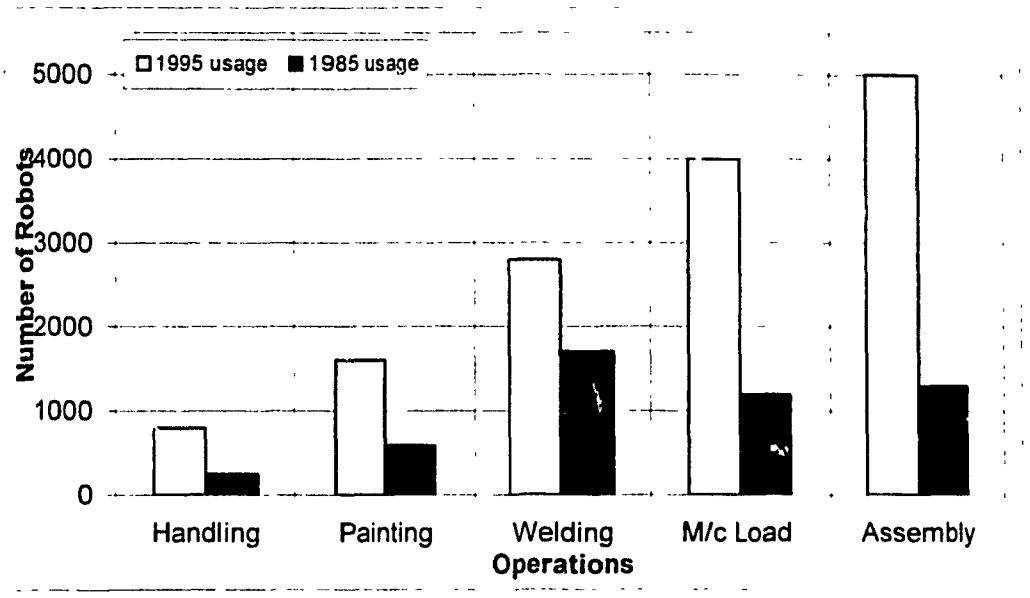


Fig. 3.3 General Motors Robot usage [GM Review, 1990]

Fig. 3.3 shows a comparison of robot usage, as of 1985 and 1995, for different tasks at General Motors [GM review, 1990].

Despite the fact that 33 percent of assembly operations are peg in hole assembly, there are not many companies venturing into it nor are there many manufacturers in the business of making Assembly robots as seen in Fig. 3.4 and 3.5, with Japan leading as a manufacturer of these robots. There is a gap between the implementation of assembly robots in industry and the research being carried out, Fig. 3.6 [Assembly Automation 1990, 91, 93]. As seen, research in the automotive industry is the highest, since it involves high turn over and has the maximum repetitive assembly tasks.

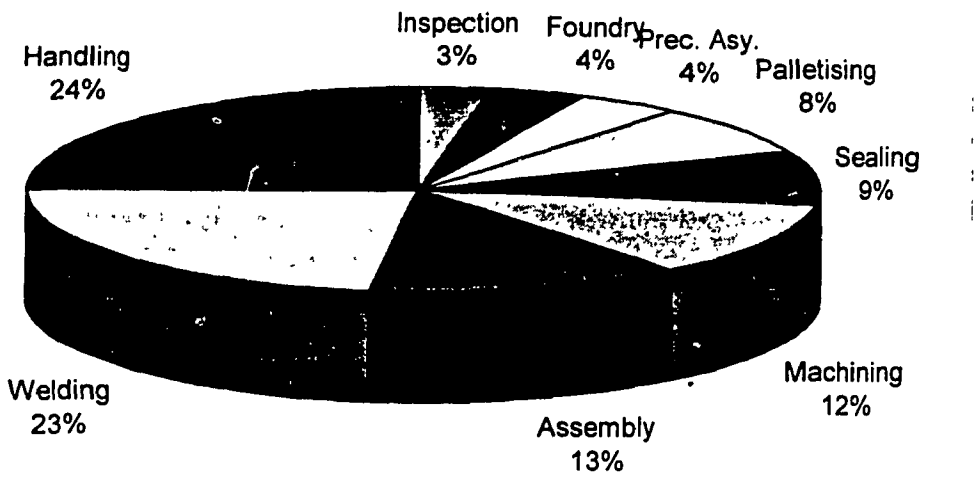


Fig. 3.4 Companies making state-of-the-art robots [Assembly Automation 1991, 1993]

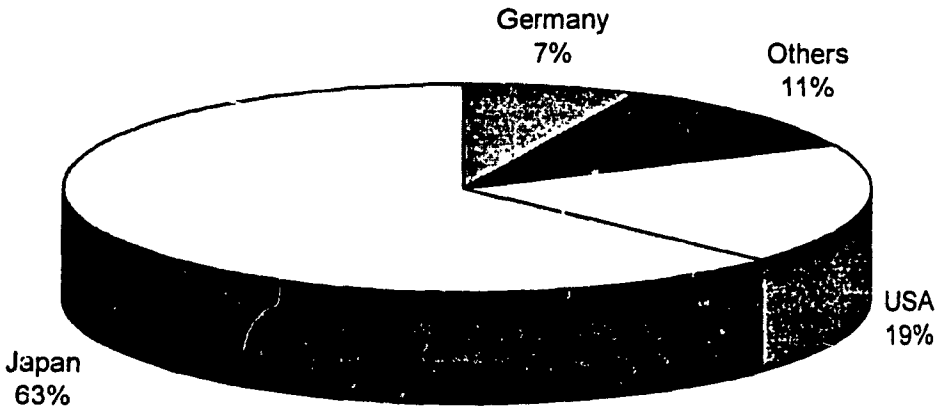


Fig. 3.5 Manufacturers of Assembly robots [Assembly, 1991]

It is also evident from the literature that there is a paucity of techniques to train robots for assembly tasks and even relatively simpler tasks like edge following and edge detection. In addition, there is a need for a compact universally suitable wrist sensor for sensing the force

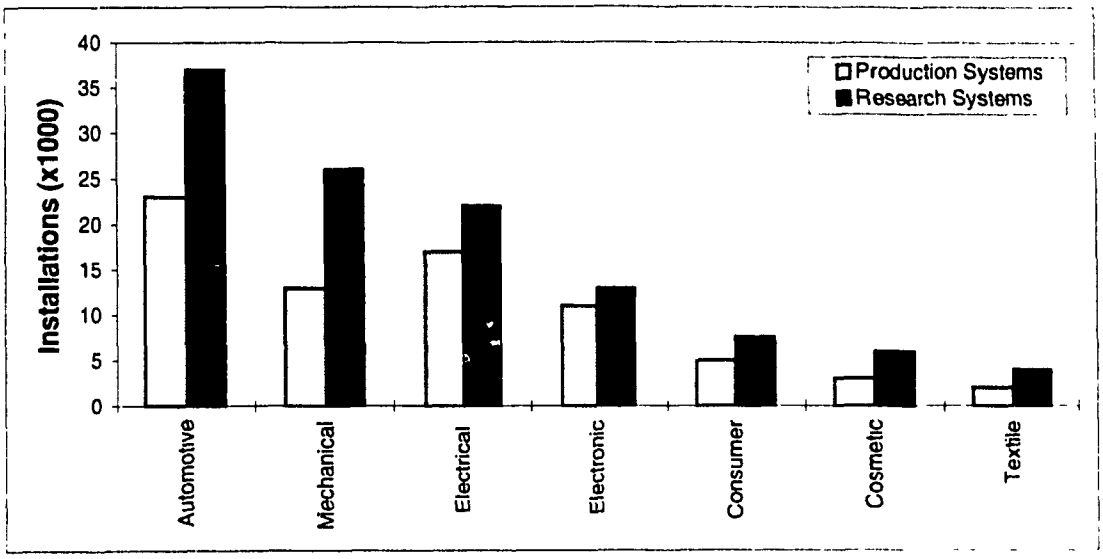


Fig. 3.6 Assembly Systems in Industrial sectors
[Assembly Automation, 1990]

that is unknown in magnitude, direction and position. Adjustable compliance is required in the area of assembly or peg-in-hole problems as well as edge detection and monitoring and this work aims to implement this feature. The existing algorithms that have been developed are based on position sensing strategies, and do not generally incorporate overload sensing features. Available sensors and passive compliance devices are large in size, are designed for dedicated applications and several of these are laboratory models.

Further, basic sensory feedback attempted so far in such operations have been visual feedback and force feedback. Visual feedback has certain shortcomings that limit its use in some high precision activities. It is difficult to use vision for final small corrections / corrective motions to assemble parts because the tool, fixtures and the parts being

assembled tend to block the view of the assembly interface. Visual feedback is generally limited to identifying parts, their locations and orientation under controlled lighting conditions. Force feedback too has a disadvantage, namely, if the direction of the applied force vector is outside certain bounds, the parts being assembled may get jammed, however, it is easier to implement in a manufacturing environment as it requires less data processing and helps reduce excess electronics.

In view of this, an application area more relevant to the automotive industry was identified and the supporting literature review provided sufficient motivation to embark on the present work, which can be extended to other industrial applications.

3.3 Scope of Work

The work was limited to the development of a machine learning algorithm for assembly of two rigid parts, (splined shaft and a hole), which essentialy is a problem of assembling multiple shafts into holes and monitoring the orientation of parts prior and during assembly. The orientation of one part was known and the other part's orientation was known within a particular range, say + / - 10 degrees.

Similar approaches have been tried by researchers [Vaaler, 1991 and Ahn, 1992], for a single shaft and hole assembly. There are three significant fundamental differences in the present work. The first is that this work was envisaged for multiple shafts assembly into

holes and used a simpler form of machine learning, for greater versatility, with termination and threshold conditions suiting this specific application. The other difference is that weighting technique is being used based on force / moment discretised steps along with logic branching to make the robot learn. Finally it is different in the compliance, in that it uses passive compliance along with implicit force feedback, in essence it uses 'Hybrid Compliance Control'. Here an algorithm was written that 'trains the robot' to execute the desirable moves based on sensor inputs. This algorithm is also able to handle unexpected data correctly, as compared to analytical solutions, since one of the most significant challenges facing assembly system designers is the presence of uncertainity [Caine, 1985].

The learning algorithm was set-up in such a manner that standard 'a-priori' decisions were made about the possibility of certain force combinations. Unsupervised learning was employed as it was felt that it produced a more robust system and a vivid picture of the effect of the underlying structure of this learning algorithm than other forms of machine learning.

Sufficient information was available from the force sensor, responses to most recent moves and results from previous assemblies. As such the approach did not require a solid model of the assembly process. The only information about the assembly that was required was a Common Assembly Path (CAP) and a termination condition. The CAP is the path that would be followed if the assembly was attempted without feedback.

The scope of work also covered the design and manufacture of a passive compliance device for mounting on the robot end effector. Different concepts were proposed and the design mentioned herein was selected and pursued for manufacture and integration.

For the force-torque sensor, a prototype version had been built in the Laboratorie d'Automatique et Mecatronique, which was scaled down and modified to reduce backlash. The scaled down model fitted well on the SCARA robot and gave good results. This sensor was calibrated for each element as well as after assembling all elements.

Finally, the forces and torques were monitored and mapped for every move of the robot and the real-world limits of the accuracy of mapping were determined. This was carried out with a view to study the implementation bottlenecks of this algorithm in the industry. The results achieved were encouraging.

The approach was proposed to be extended to the fixing of weather stripping on cars in a production line environment, where the robots are working continuously and have the highest possibility to optimise the best moves. The author was in touch with General Motors engineers and discussed the implementation for trial runs of this work on the production line.

Chapter 4

Hardware Development

Prior to proceeding to designing the hardware, a recaptulation of the force and vision control methods is required. This will enable the work to be on an objective footing after comparing the pros and cons of both methods. A brief introduction to part locating strategies preceeds the actual design of the hardware, i.e., the wrist sensor and the passive compliance device.

4.1 Part Locating Systems

Generally in any environment, parts can be located by vision systems or force control. Force control can be active or passive. Part locating systems typically use some type of a sensor to

determine the location of a part relative to the gripper, the mating part or base frame [Tomas Lozano Perez, 1983; Gordon, 1987].

Vision systems [Gordon, 1987] are generally based on cameras or position sensitive detectors. The most commonly employed method involves the use of cameras that can be mounted either on the manipulator or on a stationary system. Stationary sensor mounting has the following advantages.

* Best features to sense are the mating features of parts. If the sensor were mounted on the manipulator, the mating feature may be difficult to sense, since it will most likely face away from the upper part of the robot arm.

* With a manipulator mounted sensor there is limited flexibility in part orientation during sensing. Only the joints between the sensor mount and the end effector are available for orientation prior to sensing. Additional degrees of freedom are required for arbitrary part positioning relative to the sensor.

* If the measurement is made relative to the world frame rather than to the robot frame, the positioning of the part in the assembly is less dependent on the calibration between the sensor and the manipulator. This means that the manipulator may be

moved slightly or substituted with an entirely different manipulator without having to re-teach the assembly tasks.

Another method of locating parts is to make the parts contact each other in several places. The robot position during each of these contacts is then run through an algorithm that produces the desired relative position information [Simunovic, 1979]. Researchers have developed an algorithm that will generate the points where the parts should be made to contact in order to reduce the relative position uncertainity to a given value using the minimum number of contacts (and time).

4.2 Proposed Wrist Sensor

In light of the difficulties in using a vision sensing system, it was decided to use a force sensing mechanism. Also, incorporating a vision system in a robot cell involves high costs, in terms of equipment and information processing and interfacing. Some of the common force sensing techniques employed in various tactile and wrist sensors as reported in the literature are:

* Magnetic techniques
* Strain based measurement
* Mechanical methods
* Capacitive sensing

* Thermal techniques

* Piezo electric method

Not all of the above techniques are universally applicable, since some of these have limitations when working in vacuum, with non-metals, a low light environment, low heat surroundings, in hostile environment and continually changing force directions and orientations.

Based on the literature, strain based measurement and capacitance sensing techniques are on the forefront of development. Both these techniques employ displacement-type transducers.

* The strain based measurement gives the magnitude of force applied when the sensor material is deformed. . Although the transducer has a compliant sensor body, the deformation of the force-sensing part is negligible, due to the large modulus of elasticity

* Capacitive sensing has advantages in its application, in the form of easy maintenance, simple data processing, low power requirements, distributed sensing area, larger surface coverage, lesser cabling and simpler electronics. This technique has the disadvantage of being EMI sensitive, sensing depends on the compliant layer, environment conditions and is also prone to temperature variations.

Although capacitive sensing techniques are being researched, the major drawback in their use is that, if the environment is polluted or if there is mist or fog the functioning of the capacitive sensor is affected, as the di-electric medium changes under these conditions. As such, it was decided that designing a sensor based on strain measurement was the most viable solution.

4.2.1 Strain Measurement design

The 6 DOF sensor developed is shown schematically in Fig. 4.1, with straining rods for sensing force / torque in each direction. Each straining rod in Fig. 4.2, has strain gages mounted on its surface. Forces and torques acting on the end effector of the robot are transmitted through the metal disk to the strain rods. Since the rods are pinned on both ends, the moment about any pin is equal to zero, so the rods are always under pure compression or tension. These compressive and tensile forces on the rods produce individual strain in each rod which is measured using the strain gages. These elements are compliant enough to measure forces as low as 0.1N, in any direction of actuation. It was envisaged that when all 6 elements are assembled and fixed in place, the effective force sensing capability will be linear and still measure 0.1N. In order to reduce processing time and also not to make changes in the controller, the inputs and data processing was carried out by the computer and final commands given to the existing controller.

The forces and torques acting on the disk are resolved from the rod strains using a 6 by 6 transformation matrix which can be evaluated by a computer in real time, Equation 4.1. This transformation / coupling matrix is discussed in Appendix A.

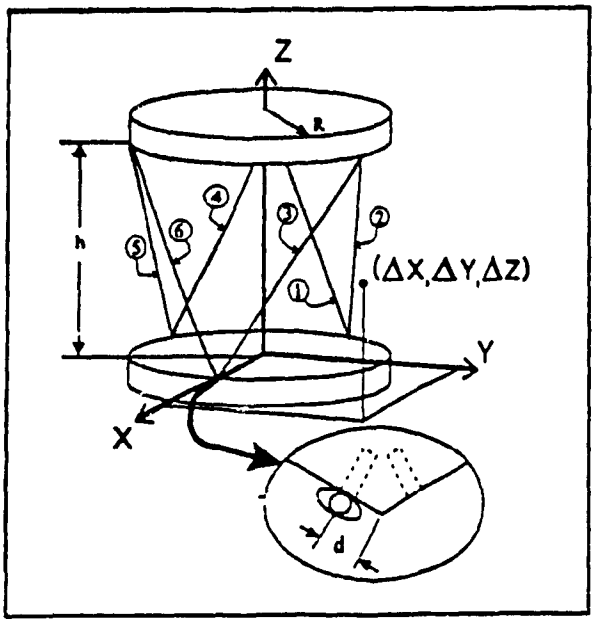


Fig. 4.1 The Six degrees of freedom force sensor

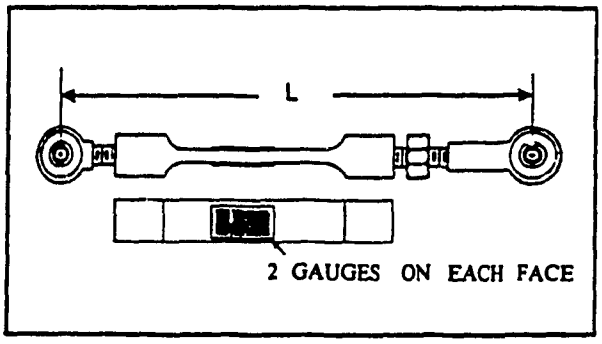


Fig. 4.2 The Straining Rod

$$\begin{bmatrix} F_X \\ F_Y \\ F_Z \\ M_X \\ M_Y \\ M_Z \end{bmatrix} = \begin{bmatrix} F_{X1} & F_{X2} & F_{X3} & F_{X4} & F_{X5} & F_{X6} \\ F_{Y1} & F_{Y2} & F_{Y3} & F_{Y4} & F_{Y5} & F_{Y6} \\ F_{Z1} & F_{Z2} & F_{Z3} & F_{Z4} & F_{Z5} & F_{Z6} \\ M_{X1} & M_{X2} & M_{X3} & M_{X4} & M_{X5} & M_{X6} \\ M_{Y1} & M_{Y2} & M_{Y3} & M_{Y4} & M_{Y5} & M_{Y6} \\ M_{Z1} & M_{Z2} & M_{Z3} & M_{Z4} & M_{Z5} & M_{Z6} \end{bmatrix} * \begin{bmatrix} F_1 \\ F_2 \\ F_3 \\ F_4 \\ F_5 \\ F_6 \end{bmatrix} \quad \text{.....} \quad (4.1)$$

where F_x, F_y, F_z and M_x, M_y, M_z represent the resultant forces and moments, respectively, relative to X, Y and Z axes. Terms F_{x1} ..... M_{z6} are parameters pertaining to geometry of the sensor.

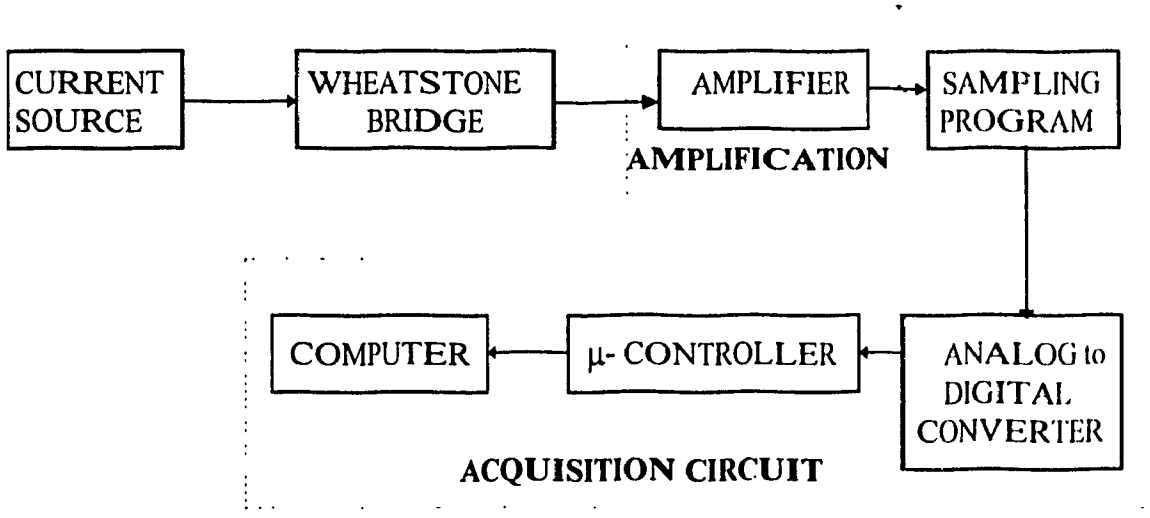


Fig. 4.3 Amplifier and Data Acquisition schematic

The design is based on the Stewart Platform mechanism. Work on a prototype sensor was carried out in the Laboratoire d'Automatique et Mecatronique [Croteau, 1995]. The present research is confined to developing a scaled down version of this prototype with certain modifications. The changes were with respect to mounting the straining rods, to ensure low backlash unlike the prototype. The dimensions and measuring range were also scaled down to suit the robot and the assembly application. The existing amplifier data acquisition system, shown schematically in Fig. 4.3, were used after suitable modifications. The first

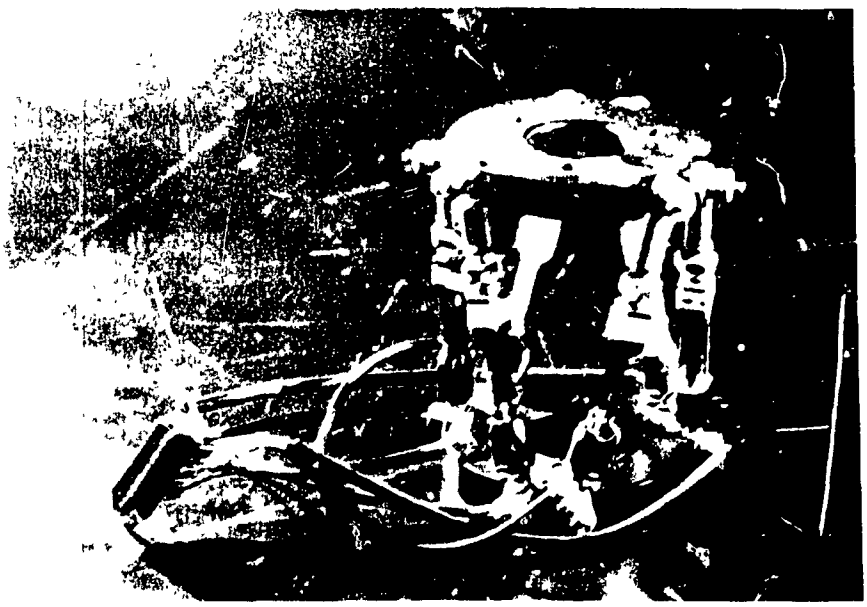


Fig. 4.4 First Prototype of Wrist Sensor [Croteau, 1995]

prototype developed is shown in Fig. 4.4 and the new wrist sensor developed for this research is in Fig. 4.5. The schematics of straining rod and the mounting platform are shown in Fig. 4.6 and 4.7.

In Appendix D, Table. I displays the dimensions of the new sensor and Table. II gives the specifications of the new sensor. Each element / straining rod of this sensor was calibrated over a range of 0 - 20 kgf and as seen from the calibration curves, the elements showed a linear behaviour, excepting small deviations at some points. These deviations can be attributed to the noise in the data acquisition system, noise in the system, and human measurement errors. The testing and calibration is discussed in Appendix F.

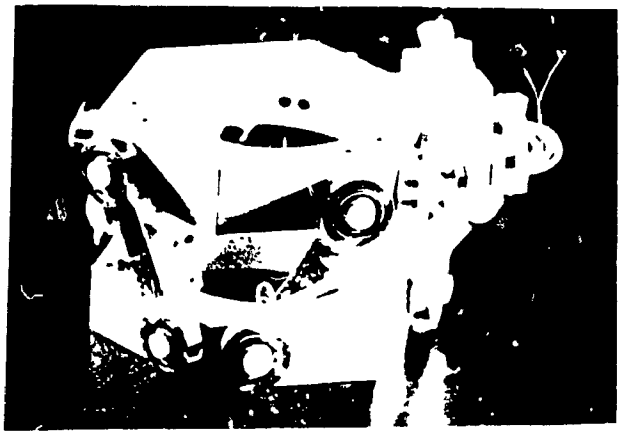


Fig. 4.5 New Wrist Sensor developed

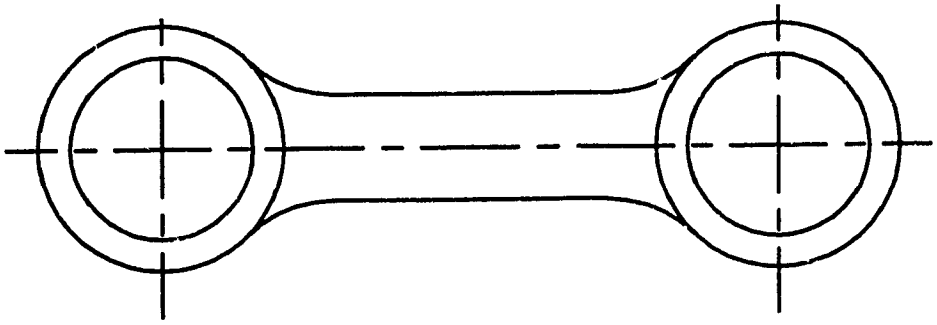


Fig. 4.6 Schematic of straining rod used in sensor

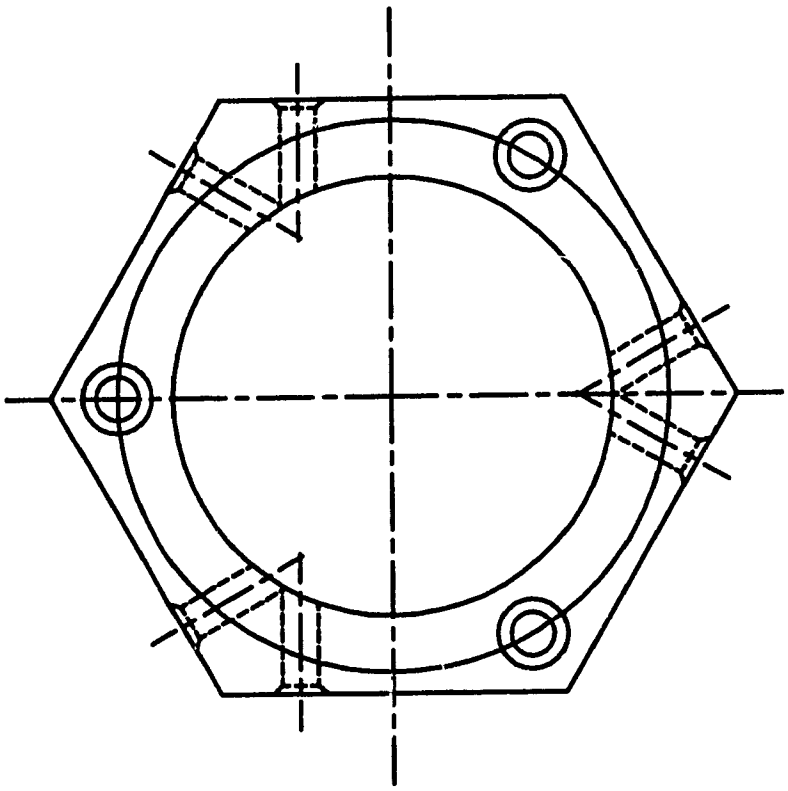


Fig. 4.7 Schematic of Mounting platform used in sensor

The new configuration of the sensor worked well displaying linear behaviour. As the foundation of this research was based on force sensing, the proper functioning of the new sensor was an encouraging factor.

4.3 Passive Compliance Device (PCD)

Compliant motion occurs when the position of the manipulator is constrained by the task. Compliant motion is an important part of a mechanical assembly system, since fitting parts

together generally requires motion between objects in contact. Ideally these operations should be performed quickly but without producing excessive forces at points of contact.

Before going into the pros and cons of employing a compliant device, it is necessary to have an idea of the magnitude of forces and torques and application areas that this device must work with. The compliance device takes care of torsional, compression forces, lateral forces, torsion and cocking, Fig. 4.8. Fig. 4.9 shows this device being used in a non-vertical application.

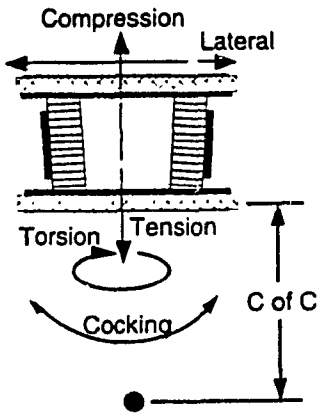


Fig. 4.8 Compliance Device orientation for various forces / torques [ATI Inc., 1985]

As mentioned earlier, two methods primarily for producing compliant motion are :

a - passive mechanical compliance built into the manipulator / end-effector

b - active compliance implemented in the software control loop,

'force control (FC) '

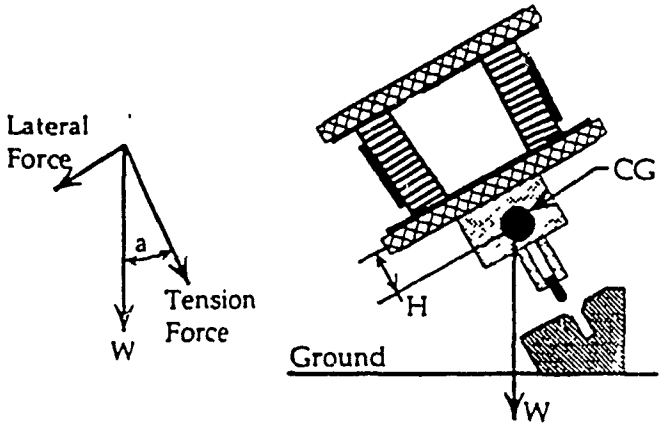


Fig. 4.9 Compliance Device in a Non-vertical application [ATI Inc..1985]

Lateral Force = W * Sin (a)

Tensile force = W * Cos (a)

Cocking Torque = Lateral force * H ..... (4.2)

The major disadvantage of active compliance is that it requires manipulator programming. Effective programming occurs only when the programmer has a thorough understanding of the programming language primitives. Hence, the ground rules should be well defined which requires a good comprehension of force control.

Using passive compliance instead of active force feedback has several major advantages like :

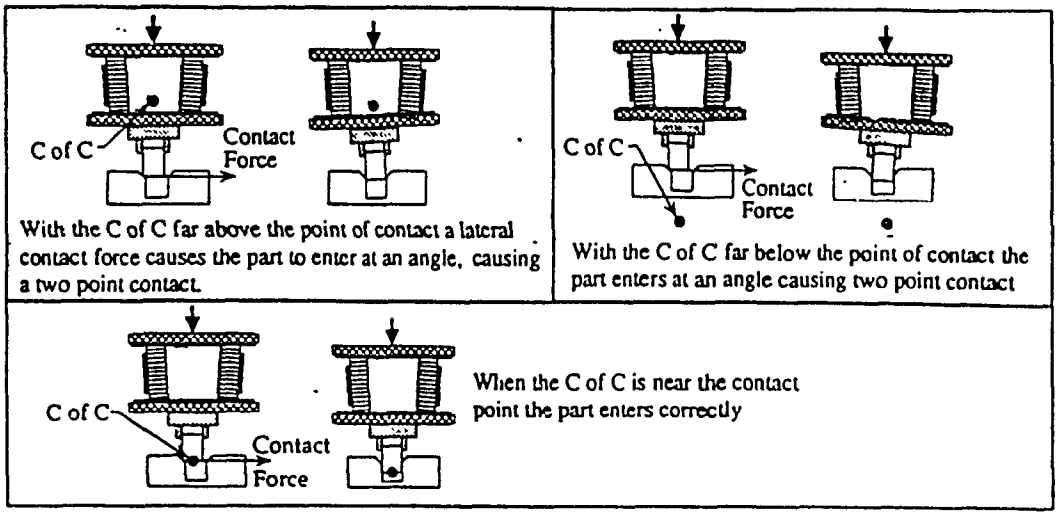


Fig. 4.10 Different locations of Centre of Compliance [ATI Inc.,1985]

* force sensor not necessary

* additional computation power not required

* some operations can be done with fewer than 6 DOFs

* assembly speed limited by input dynamics and speed of assembling machine

* do not need to re-program the robot for a new operation / job every time

* do not need an operator with knowledge of programming language primitives

* reduces costs and time

The Remote Centre Compliance (RCC) is a unique passive compliance device for aiding assembly insertion operation [Paul, 1987] which fits between a robot's arm and its end effector. It is entirely mechanical, deriving its properties from its geometry and the elasticity of its parts. Its major function is to act as a multi-axis `float', allowing positional and angular misalignments between parts to be accommodated. The RCC is designed to hold a work piece so that the piece can rotate about its tip, that is about the point where it engages a mating part. A crucial feature of the RCC is that the lateral error and angular error are absorbed independently. Its design permits lateral motion in response to laterally directed contact forces without any accompanying angular motion. During a typical assembly the lateral part does the work during engagement while the angular part takes over during insertion. The RCC is not designed to cope with the case where the error is so large that the chamfers do not meet.

A key feature to the compensator is the projected (remote) compliance centre [ATI Inc., 1985]. The Centre of Compliance (c of c) is the point in space at which a contact force will cause a translation with no rotation and a torque will cause a rotation with no translation. When the centre of compliance is near the contact point, the insertion part axis will align with the location axis during assembly. Fig. 4.10 shows different locations of the centre of compliance and their effect during assembly.

Excessive contact force is the main problem in many assembly applications [Stokic.et.al., 1986]. Excessive contact force causes galling, jamming and broken parts. During a typical assembly process [ATI Inc., 1985], there are 3 main contact forces : single-point, sliding and two-point, Fig. 4.11. The key to reducing single-point or sliding contact force is using a compliance device with a low lateral stiffness, K_L. Two- point contact force is reduced with a low cocking stiffness R_C.. Fig 4.12 shows details of contact force profile during an insertion.

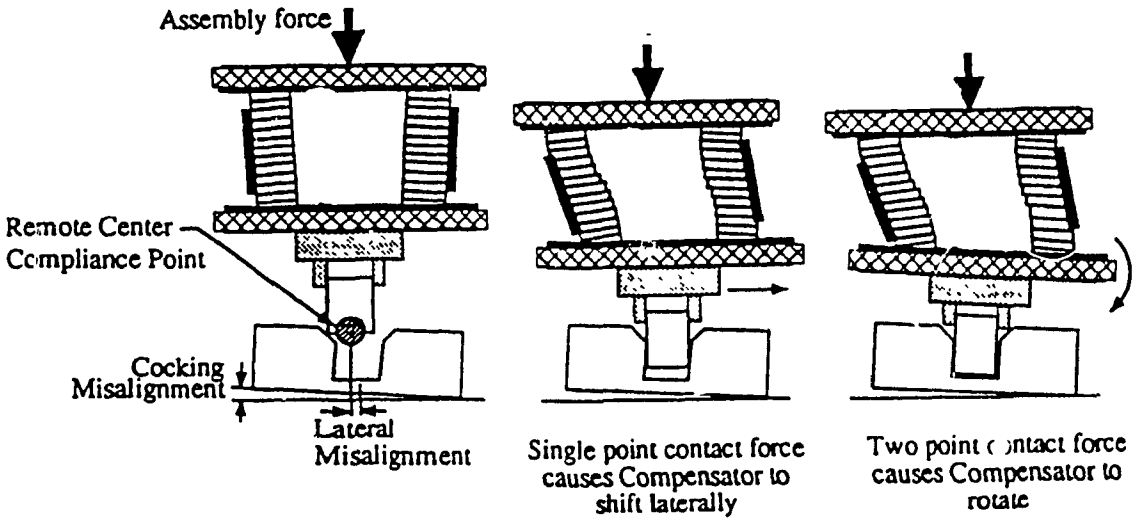


Fig. 4.11 Contact forces while assembly [ATI Inc.]

In order to arrive at the design of a new PCD, several preliminary ideas were explored and due to various reasons were not pursued for design and manufacture. In order to have a linear response and to have torsional requirements taken care of, design of the proposed

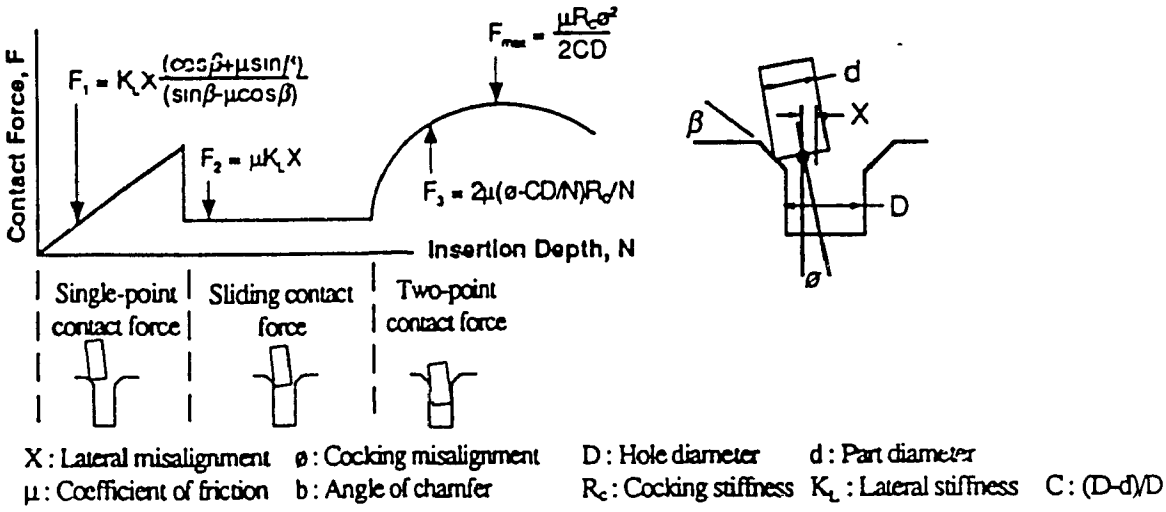


Fig. 4.12 Insertion force profile [ATI Inc.]

Passive Compliance device (PCD) shown in Fig. 4.13 was taken up. The design and testing have been discussed in Appendix C and E.

This PCD is comprised of five components. The Housing, which houses the 6 springs and the float. The Bottom Plate, this covers the base of the PCD and locates one end of the 3 springs. The Float is the disc that locates springs at the top and bottom sections of the PCD. This component plays an important role as it is subjected to compression forces and torsion. It also helps the PCD to have different DOFs. The Top Plate is located on top of the

housing, it locates one end of the three springs and also acts as a junction between the wrist sensor of the robot and the PCD. At the bottom most part is a Mounting Plate which is connected to the float by a bolt and takes care of the various forces and torques in conjunction with the float. This plate is also connected to the gripper on the lower surface and is the link between the PCD and the end-effector. Detailed drawings of these components are in Appendix C.

The PCD was tested in the laboratory and after mounting on the robot. As for the translational range and cocking values it was checked before mounting on the robot. Specifications of the PCD are presented in Appendix D, a comparison with commercially available PCDs in trade is also presented.

Advantages of this PCD : There are some very significant advantages of this design vis-à-vis the devices developed in the literature, which deserve to be mentioned.

1. The device is very simple to manufacture and is versatile in applications.
2. It has low lateral stiffness compared to the devices available in trade, thus reducing single point and sliding contact forces.
3. It has low cocking stiffness helping reduce the 2 point contact force.
4. It is modular and can be used for different types of fits. If a tight fit is required in a certain assembly task, the PCD can be used for that task, by

changing the set of springs, using more stiffness and resulting in lower longitudinal travel of the PCD.

5 The travel range in the Z axis can be changed depending on the compliance required in an application. This can be done by either increasing

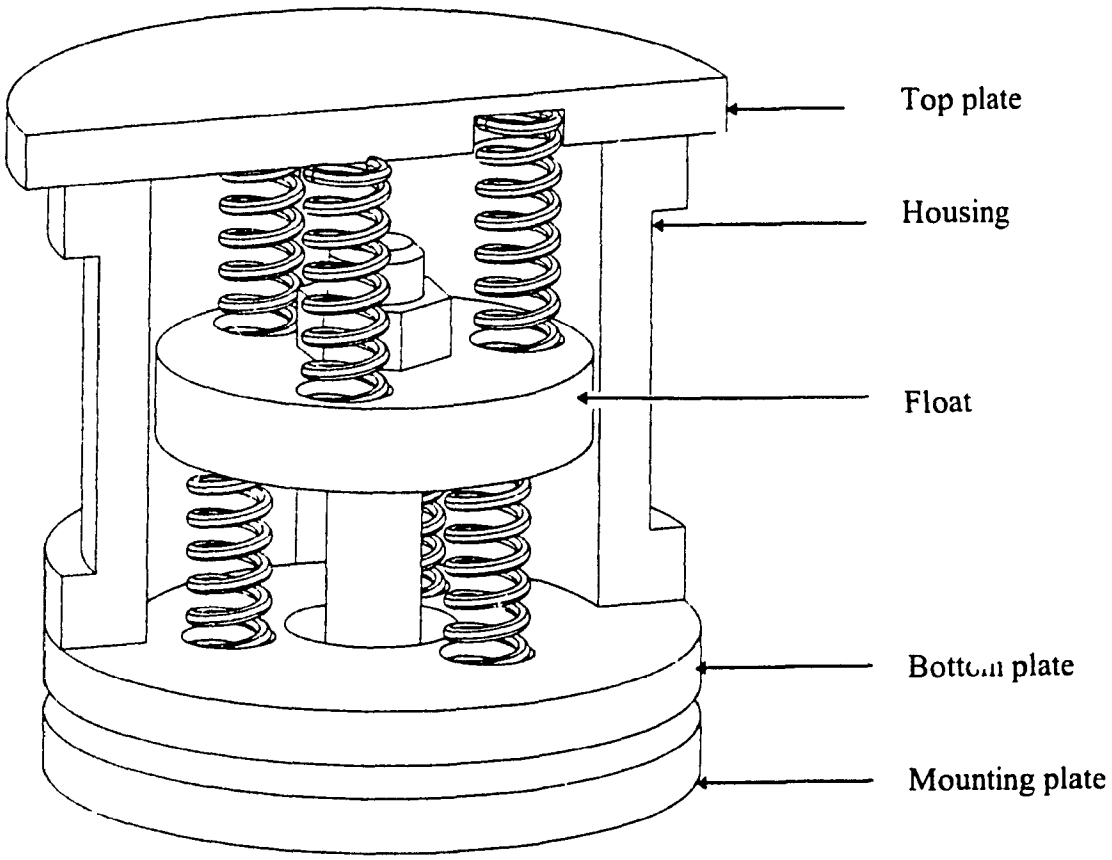


Fig. 4.13 Proposed Passive Compliance Device

or decreasing the number of washers to be used in the locating holes.

6. The Stiffness can be varied in the Z direction by changing the pre-load on the springs, or by changing the number of washers.

7. The PCD is designed to be the weakest link in the whole system. If the torsional or longitudinal limits are exceeded during an assembly task, the springs have been installed so as to dislocate from their positions. This dislocation causes a sudden drop in the Z axis force, which is instantly sensed by the algorithm and the robot aborts the task, thus avoiding damage to the parts being assembled as well as to the robot.

The above advantages justify the development of the new passive device for this research. As seen, the device is not similar to the generic RCC devices. Depending on the application, the device can be modified to create / have a RCC by changing the spring mounting positions and orientation.

Chapter 5

Assembly Task Formulation

This chapter will discuss the mechanics of part-mating, the failures in part-mating, assigning of task frames, configuration details and parameters that were monitored during the process of assembly.

5.1 Mechanics of Assembly

The mechanics of part mating are governed by the geometry of the parts, the stiffness of the parts and tooling, the friction between parts as they move past each other during assembly, and the amount of lateral and angular error between the parts as mating begins. The interplay of these factors determines whether assembly will be successful and how large the forces exerted on the parts by the tooling and each other will be [Nevins.et.al 1989].

The success or failure of a peg-hole assembly depends on how the parts behave while passing through two potential danger zones. First, the lateral or angular errors before assembly could be so large that the parts fail to meet within the bounds of the chamfers (or part diameters if there are no chamfers). Second, there are two forms of failure, as discussed before, associated with two-point contact during the fine motion phase; these are called 'wedging' and 'jamming'. These two are explained in the following paragraphs and followed by a brief overview of forces.

Wedging is an event in which the contact forces between peg and hole can set up compressive forces inside the peg, effectively trapping it part way in the hole. To avoid wedging, the angular error between peg and hole at the onset of two point contact must be small.

Jamming is an event in which the peg cannot advance into the hole because the insertion force vector points too far off the axis of the hole. To avoid jamming, one must support the peg so that the reaction forces set up by the two contact points are able to turn the peg parallel to the hole's axis. These supports are also important in chamfer crossing and avoidance of wedging.

Wedging and Jamming

Wedging and jamming are conditions that arise from the interplay of forces between parts. The forces applied to the peg by the compliances are represented by F_x, F_z, and M at or about the tip of the peg [Nevins.et.al 1989]. The forces applied to the peg by its contact with the hole are represented by F_1 and F_2 and the friction forces normal to the contacted surfaces. The coefficient of friction is μ. (In case of one-point contact, there is only one contact force and its associated friction force). The analyses that follow assume that these forces are in approximate static equilibrium. This means in practice that there is always some contact, either one point or two point and that accelerations are negligible. The analyses also assume that the support for the peg can be having a compliance center.

Wedging can occur if two-point contact occurs when the peg is not very far into the hole. A wedged peg and hole are shown in Fig. 5.1 and the forces and moments on a peg in 2 - point contact are shown in Fig. 5.2. The contact forces F_1 and F_2 are pointing directly toward each other. The smallest value of θ for which this can occur is θ_w, given by

$$\theta_w = c / \mu \qquad \text{.....} \qquad (5.1)$$

Geometric conditions for stage 1, for successful entry of the peg into the hole and the avoidance of wedging can be stated, in terms of initial lateral and angular errors. To cross the chamfer and enter the hole,

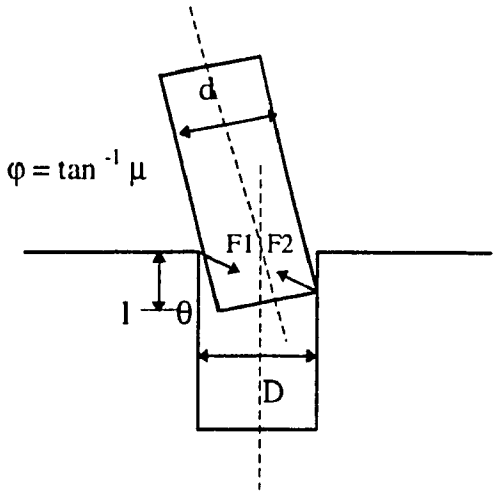


Fig. 5.1 Wedged peg and hole, when l = μd [Nevins.et.al 1989]

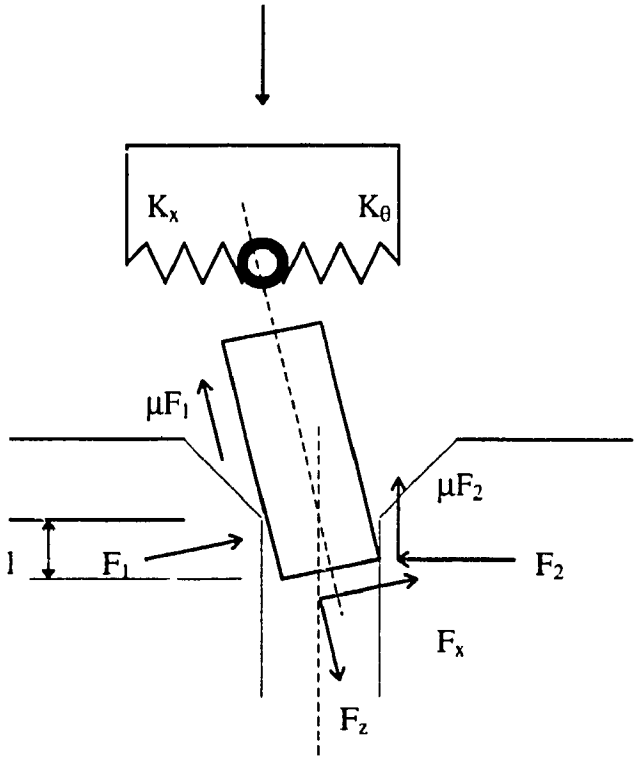


Fig. 5.2 Forces / Moments on a peg in 2-point contact [Nevins.et.al. 1989]

$$|\varepsilon_0| < W \qquad \text{..... (5.2)}$$

where W is the sum of chamfer widths on the peg and hole, ε_0 is distance between shaft and hole axes, and

$$\theta_0 + S\varepsilon_0 < \pm c/\mu \qquad \text{..... ..... (5.3)}$$

where,

$$S = L_g / (L_g^2 + K_\theta / K_x)$$

If the parts get wedged, there is generally no cure except to withdraw the peg and try again. Wedging should be avoided in the first place. Avoiding wedging is related to success in initial entry, and both are governed by control of initial lateral and angular errors. The amount of permitted lateral error depends on the amount of angular error.

If $L_g = 0$, the interaction between lateral and angular errors disappears. This makes planning of an assembly the easiest and makes the error window the largest. Jamming can occur because the wrong combination of applied forces is acting on the peg.

Fig. 5.3 shows a schematic assembly of a round peg and hole in two dimensions, although assembly is 3 D in general. The figures define four typical phases of an assembly: approach, chamfer crossing, one-point contact, and two-point contact. Not every assembly contains all of these phases but most do [Nevins.et.al 1989].

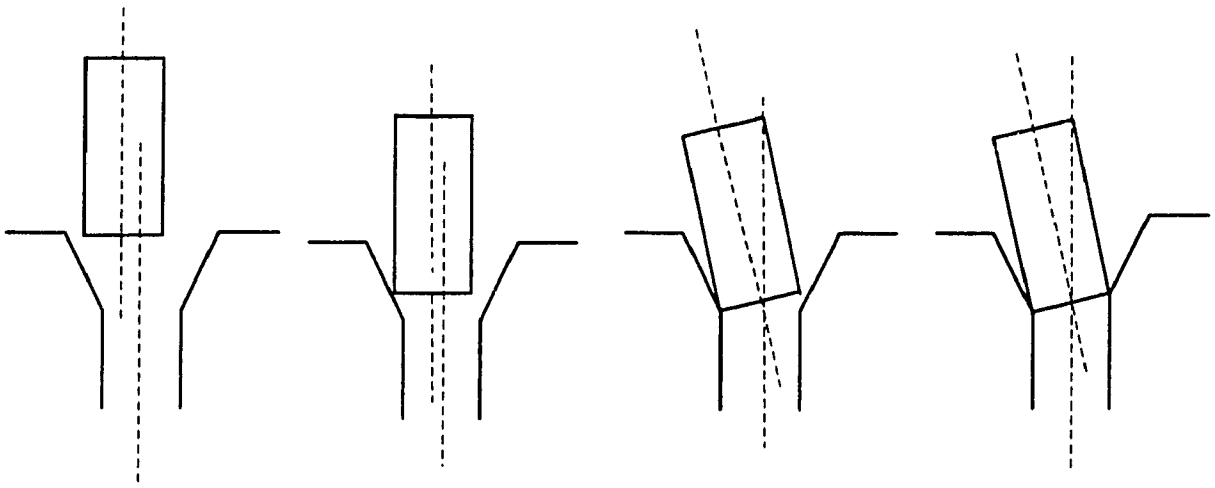


Fig. 5.3 Typical phases of an Assembly of peg-hole [Nevins.etal., 1989]

As the shaft advances farther into the hole, it finally strikes the opposite side, establishing a second contact point. During the 2 point contact phase, the parts try to rotate with respect to each other to remove angular errors. The part is turned angularly by the torque created by the forces acting at the two contact points. In some cases, two-point contact may be followed by line-contact, in which the parts are exactly parallel and in contact along one wall of the hole.

These moves constitute the fine motions of a typical simple assembly. Other assemblies, such as push-twist, snap actions, and thread mating, include other fine motions.

5.2 *Assembly Equations*

This section introduces and derives assembly equations, for the peg and hole assembly. Various researchers have carried out extensive work in this area considering assembly with and without chamfers. The equations [Hennessey, 1982], are being derived here for the sake of completeness and in order to bring out the dynamics of the assembly process. Researchers have considered two different cases of compliance, translational and rotational, and have derived the equations separately as well as with both compliances combined. This can be applied to the spline shaft-hole assembly, as it is in the case of multiple shaft-hole assembly.

Lateral Compliance

This section introduces the shaft-hole subjected to lateral compliance in the hole. Fig. 5.4 illustrates the initial configuration of the peg and hole with the compliance center of the peg indicated. During assembly, the hole walls will initially deform outward, enlarging the hole. This deformation (although small) will be treated as a uniform lateral translation of the hole walls parallel to their initial position. Both sides (left and right) will deform away from the center line of the hole so that the distance between the center axis of the hole and the sides of the hole will always be non-negative (i.e., $\delta x_1, \delta x_2 \geq 0$). The quasi-static phases of successful assembly to be analyzed are (I) chamfer crossing, followed by (ii) one-point contact, (iii) two-point contact, (iv) resumption of one-point contact and the final phase, (v) line contact. Line contact occurs when the peg is in a vertical position and

is in contact with the left side of the hole. The compliance center's position along the peg's axis can greatly affect the assembly characteristics.

Chamfer Crossing

Chamfer crossing is shown in Fig. 5.5, with all external forces and moments present, in the free body diagram. Other geometrical parameters and the insertion variables are also indicated.

The positions of the shaft and hole during chamfer crossing are completely determined by

(a) balancing the external forces and moments on the shaft

(b) invoking geometric constraints on the shaft and the hole.

The force and moment balance involves a horizontal and vertical force balance along with a moment balance at the peg's compliance center. The peg's support and the hole's wall stiffness may be readily identified. From Fig. 5.5, the following equations [Hennessey, 1985], can be derived.

Equilibrium Requirements:

$$F_x = F_n (\sin\phi - \mu \cos\phi)$$

$$F_Z = F_n (\cos\phi + \mu \sin\phi)$$

$$M = F_n \{ a [\sin(\phi + \delta\theta) - \mu \cos(\phi + \delta\theta)] - [\cos(\phi + \delta\theta) + \mu \sin(\phi + \delta\theta)] * d/2 \} \quad \ldots\ldots \quad (5.4)$$

Force-deformation relations:

$$F_x = K_x \delta x$$

$$M_x = K_\theta \delta_\theta$$

$$K_{x1} \delta x_1 = F_n (\sin\phi - \mu \cos\phi) \qquad \text{..... (5.5)}$$

Geometric Compatibility requirements:

$$\frac{\Delta z}{\tan\phi} = \delta x + \delta x_1 + a \sin\delta\theta + d \sin^2(\delta\theta / 2)$$

$$a + \Delta z = \delta z + a \cos\delta\theta + (d/2) \sin\delta\theta \qquad \text{..... (5.6)}$$

Variable Δz is defined as the insertion distance. Chamfer crossing begins when $\Delta z = 0$ and ends when $\Delta z = (\Delta - CD/2) \tan\phi > 0$. Here C is the clearance ratio, defined as

$$C = \frac{D - d}{D} \qquad \text{..... (5.7)}$$

To avoid jamming during assembly, the compliance center must be located at least at a distance of $\frac{d}{2\tan(\phi + \delta\theta - \beta)}$ from the end of the shaft. The friction angle (β) is given as

$$\beta = \tan^{-1}\mu \qquad \text{..... ..... (5.8)}$$

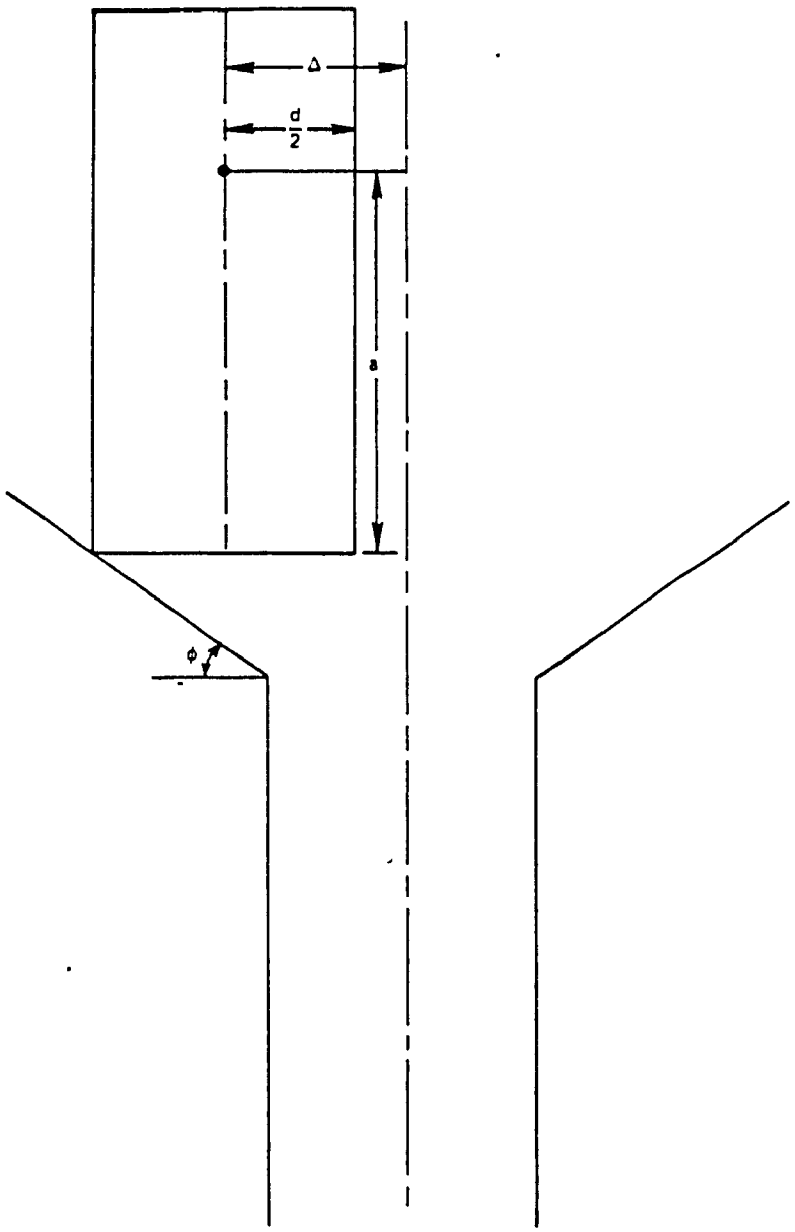


Fig. 5.4 Initial configuration of shaft and hole [Hennessey, 1982]

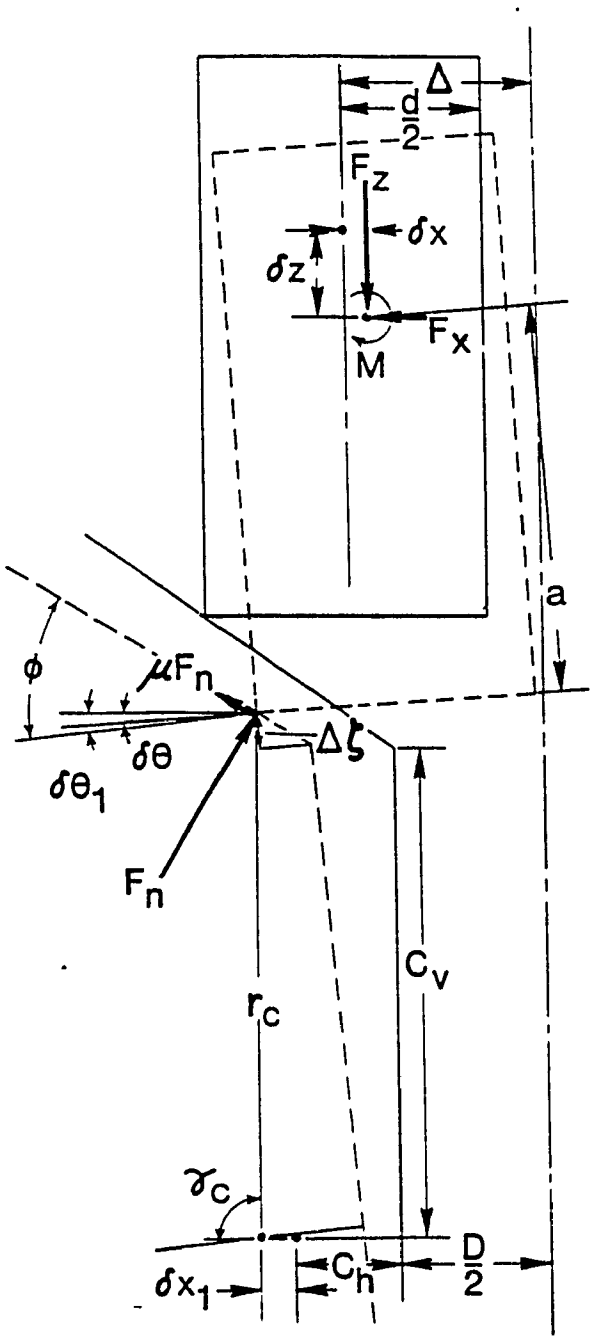


Fig. 5.5 Chamfer crossing with lateral and rotational compliance[Hennessey '82]

which can be derived by requiring the normal force (F_n) and the angle of the shaft with respect to the vertical ($\delta\theta$) to non-negative values in the moment balance equation 5.4 above. Since $\delta\theta$ is not known a-priori, an estimate of the maximum value of $\delta\theta$ will provide an estimate of the minimum acceptable value of a. Also, to avoid wedging the chamfer angle (ϕ) must be greater than ($\beta - \delta\theta$).

One-point Contact

Chamfer crossing is not immediately followed by one-point contact. Instead a transient phase occurs while the normal force (F_n) changes direction so as to align itself perpendicularly to the surface of the side of the peg. This phase, although quite brief, is responsible for producing a discontinuity in all of the insertion variables between chamfer crossing and one-point contact. In effect it is quite small, less than 5 %.

Proceeding on similar lines, various conditions will be arrived at. The one-point contact phase is shown in Fig. 5.6, with various forces and moments.

Equilibrium Requirements:

$$F_x = F_{n1}(\cos\delta\theta - \mu\sin\delta\theta)$$

$$F_z = F_{n1}(\sin\delta\theta + \mu\cos\delta\theta)$$

$$M = F_{n1}\{(a - l) - \mu d/2\} \quad \text{.....} \quad (5.9)$$

Force-deformation relations:

$$F_x = K_x \delta x$$

$$M_x = M_\theta \delta_\theta$$

$$K_{xl} \delta x_l = F_{nl} (\cos \delta\theta - \mu \sin \delta\theta) \quad \text{..... (5.10)}$$

Geometric Compatibility requirements:

$$\Delta z - \frac{CD}{2} = \delta x + \delta x_l + (a - l) \sin\delta\theta + d \sin^2(\delta\theta / 2)$$

$$a + (\Delta - CD/2) \tan \phi = \delta z + (a - l) \cos \delta\theta + (d/2) \sin \delta\theta \quad \text{..... (5.11)}$$

Where 'l' is the insertion distance. One-point contact begins when the 'l = 0'. Also, it is required that the angle of shaft with respect to the vertical ($\delta\theta$) be non-negative. From Equations 5.9 and 5.10 it follows that the distance from the tip of the shaft to its compliance center (a) must at least be $\mu d/2$, as the equation is only valid for small values of 'l'.

One-point contact ends and two-point contact begins when the lower right corner of the peg comes in contact with the right side of the hole. To determine the values of 'l' and other insertion variables for which two-point contact begins, the following additional geometric constraint is included with the one point contact equations when solving for these insertion variables.

$$l \sin \delta\theta + d \cos \delta\theta = D + \delta x_l \quad \text{..... (5.12)}$$

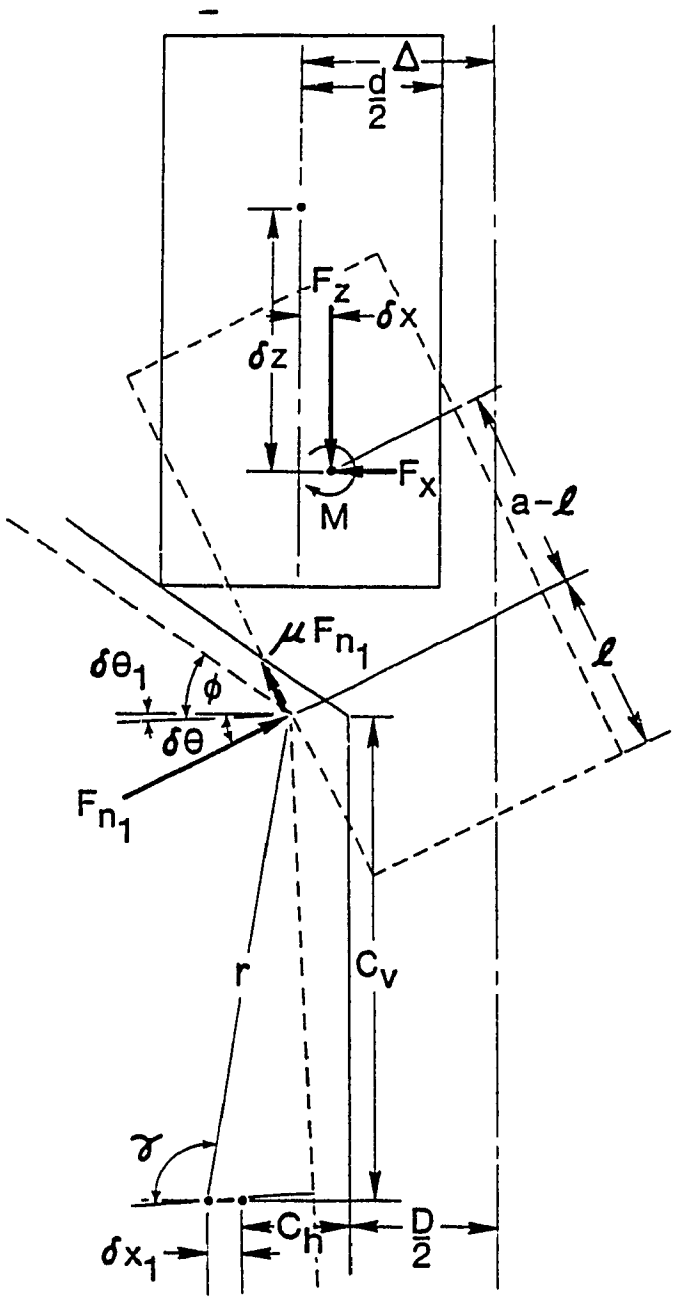


Fig. 5.6 1-point contact with lateral and rotational compliance [Hennessey '82]

This equation indicates that the shaft's lower right corner has just touched the right wall (no normal force as yet).

Two-point Contact

Fig. 5.7 illustrates the free body diagram for the 2-point contact, the equations can be defined along similar lines as above.

Equilibrium Requirements:

$$F_x = F_{n1}(\cos\delta\theta - \mu\sin\delta\theta) - F_{n2}$$

$$F_Z = F_{n1}(\sin\delta\theta + \mu\cos\delta\theta) + \mu F_{n2}$$

$$M = F_{n1}[(a-l) - \mu d/2] - F_{n2}[a(\cos\delta\theta - \mu\sin\delta\theta) - (\sin\delta\theta + \mu\cos\delta\theta)*d/2] \quad \text{..... (5.13)}$$

Force-deformation relations:

$$F_x = K_x\delta x$$

$$M_x = M_\theta\delta_\theta$$

$$K_{x1}\delta x_1 = F_{n1}(\cos\delta\theta - \mu\sin\delta\theta)$$

$$K_{x2}\delta x_2 = F_{n2} \quad \text{..... ..... (5.14)}$$

Geometric Compatibility requirements:

$$\Delta z - \frac{CD}{2} = \delta x + \delta x_1 + (a-1)\sin\delta\theta + d\sin^2(\delta\theta/2)$$

$$a + (\Delta - CD/2)\tan\phi = \delta z + (a-1)\cos\delta\theta + (d/2)\sin\delta\theta$$

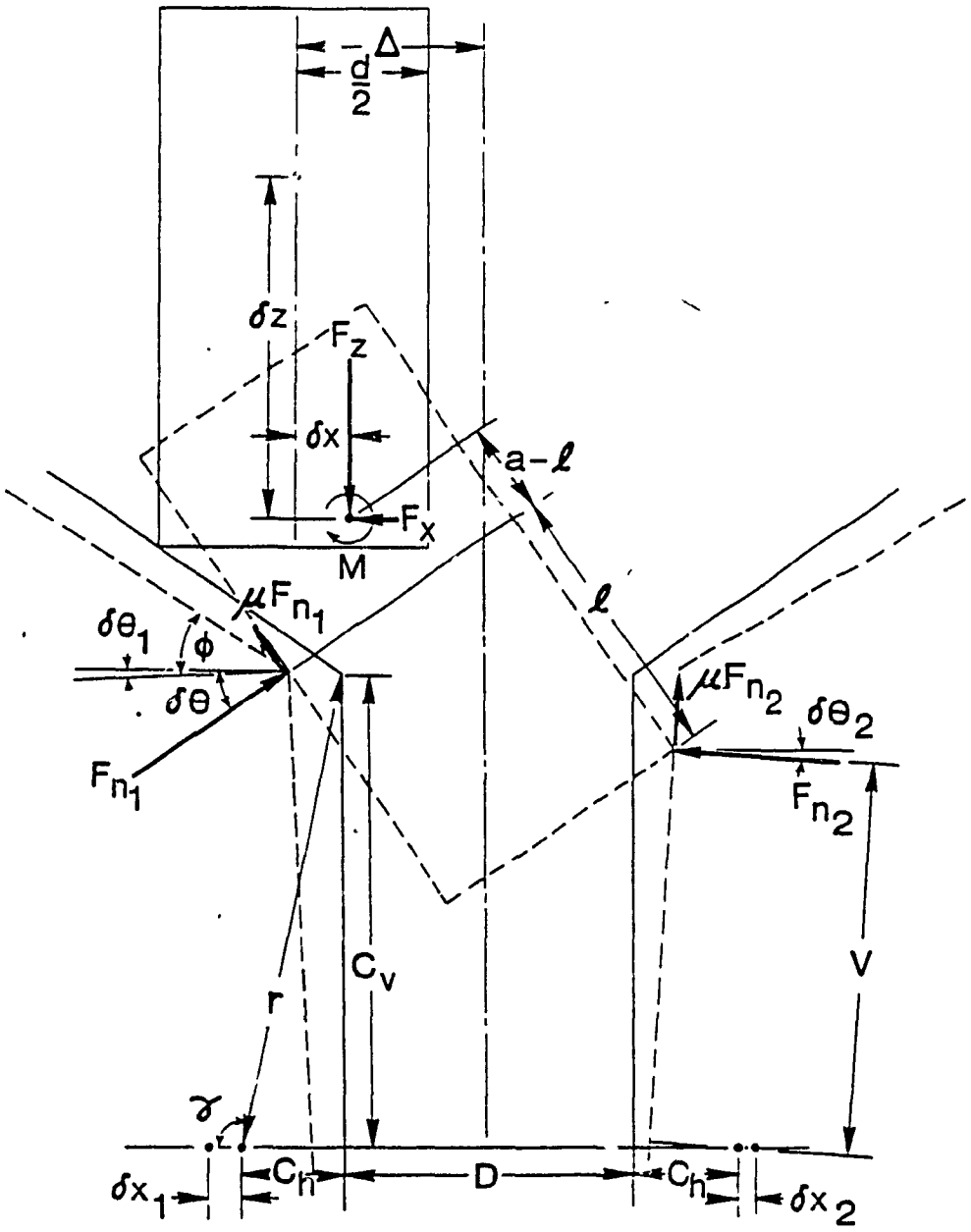


Fig. 5.7 2-point contact with lateral and Rotational compliance [Hennessey '82]

$$l \sin \delta\theta + d \cos \delta\theta = D + \delta x_1 + \delta x_2 \qquad \text{..... (5.15)}$$

The solutions to these have been derived by Hennessey using exact methods as well as linearized solutions and are not herein, however, the results of these solutions are presented in the following sections.

Chamfer crossing with lateral compliance

$$\delta x = \frac{K_\theta \Delta z}{\tan\phi[K_x a^2 (1 - \frac{d}{2a\tan(\phi-\beta)}) + K_\theta + \frac{K_x K_\theta}{K_{x1}}]}$$

$$\delta\theta = \frac{K_x a \Delta z}{\tan\phi[K_x a^2 + \frac{K_\theta}{(1 - \frac{d}{2a\tan(\phi-\beta)})} + \frac{K_x K_\theta / K_{x1}}{(1 - \frac{d}{2a\tan(\phi-\beta)})}]} \qquad \text{..... (5.16)}$$

One-point Contact with lateral compliance

$$\delta x = \frac{K_\theta(\Delta - \frac{CD}{2})}{K_x(a-1)[(a-1) - \frac{\mu d}{2}] + K_\theta + K_x K_\theta / K_{x1}}$$

$$\delta\theta = \frac{K_\theta(\Delta - \frac{CD}{2})[(a-1) - \frac{\mu d}{2}]}{K_x(a-1)[(a-1) - \frac{\mu d}{2}] + K_\theta + K_x K_\theta / K_{x1}} \qquad \text{..... (5.17)}$$

Two-point contact with lateral compliance

$$\delta x = \frac{K_\theta K_{x1}(\Delta - \frac{CD}{2}) + K_\theta K_{x2}(\Delta + \frac{CD}{2}) + K_{x1}K_{x2}l[l(\Delta + \frac{CD}{2}) - aCD]}{K_x K_\theta + K_x K_{x1}(a-l)[(a-l) - \frac{\mu d}{2}] + K_x K_{x2} a(a - \frac{\mu d}{2}) + K_\theta K_{x1} + K_\theta K_{x2} + K_{x1}K_{x2}l^2}$$

$$\delta\theta = \frac{K_x K_{x1}(\Delta - \frac{CD}{2})[(a-l) - \frac{\mu d}{2}] + K_x K_{x2}(a - \frac{\mu d}{2})(\Delta + \frac{CD}{2}) + K_{x1}K_{x2}lCD}{K_x K_\theta + K_x K_{x1}(a-l)[(a-l) - \frac{\mu d}{2}] + K_x K_{x2} a(a - \frac{\mu d}{2}) + K_\theta K_{x1} + K_\theta K_{x2} + K_{x1}K_{x2}l^2}$$

..... (5.18)

Rotational Compliance

If the hole has rotational compliance too, then the equations are derived similarly, taking into consideration the angular offset. The details of analysis are available in Hennessey. The final values of δx and $\delta\theta$ are obtained as below.

Chamfer crossing with rotational compliance

$$\delta x = \frac{K_q a Dz}{\tan f[K_x a^2(1-\frac{d}{2a\tan(f-b)})+ K_q + \frac{\frac{K_x K_q}{K_{q1}}(C_v+Dz)[(C_v+Dz)\tan(f-b)+(\frac{Dz}{\tan f}-C_h)]}{\tan(f-b)}]} \quad \ldots\ldots\ldots\ldots \quad (5.19)$$

$$\delta\theta = \frac{K_x a Dz}{\tan f[K_x a^2 + \frac{K_q}{(1-\frac{d}{2a\tan(f-b)})}+ \frac{K_x K_q}{K_{q1}}\frac{(C_v+Dz)[(C_v+Dz)\tan(f-b)+(\frac{Dz}{\tan f}-C_h)]/\tan(f-b)}{(1-\frac{d}{2a\tan(f-b)})}]} \quad \ldots\ldots\ldots\ldots \quad (5.20)$$

<u>One-point contact with rotational compliance</u>

$$\delta x = \frac{K_\theta(\Delta - \frac{CD}{2})}{K_x(a-1)[(a-1)-\frac{\mu d}{2}]+ \tilde{K}_3 + \frac{K_x K_\theta}{K_{\theta 1}} C_v(C_v - \mu C_h)} \quad \ldots\ldots\ldots\ldots \quad (5.21)$$

$$\delta\theta = \frac{K_x(\Delta - \frac{Cd}{2})[(a-l) - \frac{\mu d}{2}]}{K_x(a-l)[(a-l) - \frac{\mu d}{2}] + K_\theta + \frac{K_x K_\theta}{K_{\theta l}} C_y (C_y - \mu C_h)} \quad \text{.....} \quad (5.22)$$

Lateral and Rotational Compliance

Considering both, lateral and rotational compliances in the hole walls together, is equivalent to having both, the lateral and rotational compliances individually. As the rotational compliance hole problem is much more difficult than the lateral compliance hole problem, the basis of deriving the equations will be the rotational compliance. The assembly equations, equilibrium requirements and force deformations are the same as before, but the geometric compatibility requirements are different.

Geometric compatibility requirements

Chamfer crossing

$$\frac{\Delta z}{\tan\phi} = \delta x + \delta x_l + a\sin\delta\theta + r_c[\cos(\gamma_c - \delta\theta_l) - \cos\gamma_c] + d\sin^2\frac{\delta\theta}{2}$$

$$a + \Delta z = \delta z + a\cos\delta\theta + \frac{d}{2}\sin\delta\theta + r_c[\sin(\gamma_c - \delta\theta_l) - \sin\gamma_c]$$

..... (5.23)

One-point Contact

$$\Delta - \frac{CD}{2} = \delta x + \delta x_1 + (a-1)\sin\delta\theta + r[\cos(\gamma - \delta\theta_1) - \cos\gamma] + d\sin^2\frac{\delta\theta}{2}$$

$$a + (\Delta - \frac{CD}{2})\tan\phi = \delta z + (a-1)\cos\delta\theta + \frac{d}{2}\sin\delta\theta + r[\sin(\gamma - \delta\theta_1) - \sin\gamma]$$

..... (5.24)

Two-point contact

$$\Delta - \frac{CD}{2} = \delta x + \delta x_1 + (a-1)\sin\delta\theta + r[\cos(\gamma - \delta\theta_1) - \cos\gamma] + d\sin^2\frac{\delta\theta}{2}$$

$$a + (\Delta - \frac{CD}{2})\tan\phi = \delta z + (a-1)\cos\delta\theta + \frac{d}{2}\sin\delta\theta + r[\sin(\gamma - \delta\theta_1) - \sin\gamma]$$

$$l\sin\delta\theta + d\cos\delta\theta = D + \delta x_1 + \delta x_2 + r[\cos(\gamma - \delta\theta_1) - \cos\gamma] + 2C_h\sin^2\frac{\delta\theta_2}{2} + V\sin\delta\theta_2$$

..... (5.25)

Also, the boundary between one-point contact and two-point contact is defined by

$$l\sin\delta\theta + d\cos\delta\theta = D + \delta x_1 + r[\cos(\gamma - \delta\theta_1) - \cos\gamma]$$

..... (5.26)

Solution for assembly equations

Researchers have solved these equations using exact method.

Chamfer crossing

$$(\frac{1}{K_x}+\frac{1}{K_{x1}})\frac{K_{q1}dq_1}{D_1}[\sin(f-dq_1)-m\cos(f-dq_1)]+a\sin dq+d\sin^2\frac{dq}{2}$$

$$+r_c[\cos(g_c-dq_1)-\cos g_c]=\frac{Dz}{\tan f}$$

..... (5.27)

$$K_q dq+\frac{K_{q1}dq_1}{D_1}[(am+\frac{d}{2})\cos(f+dq-dq_1)]=0$$

where

$$D_1=(\Delta\zeta+C_v)(\sin\phi-\mu\cos\phi)+(\frac{\Delta\zeta}{\tan\phi}-C_h)(\cos\phi+\mu\sin\phi) \quad \text{..... (5.28)}$$

These can be further written in the form,

$$f_1(\theta)=\frac{\Delta z}{\tan\phi} \quad \text{and} \quad f_2(\theta)=0 \quad \text{..... (5.29)}$$

One-point contact

Equations for incremental angle change are obtained as,

$$\delta\theta=\frac{K_{\theta_1}\delta\theta_1(\delta\theta)[(a-1)-\frac{\mu d}{2}]}{K_\theta[(C_v-\mu C_h)\cos(\delta\theta-\delta\theta_1(\delta\theta))-(C_h+\mu C_v)\sin(\delta\theta-\delta\theta_1(\delta\theta))]} \quad \text{..(5.30)}$$

where

$$\delta\theta_1(\delta\theta)=\gamma-\cos^{-1}[\frac{f(\delta\theta)}{r}+\cos\gamma]$$

and

$$f(\delta\theta) = \Delta - \frac{CD}{2} - \frac{(\frac{i}{K_s} + \frac{l}{K_{xl}})K_\theta \delta\theta(\cos\delta\theta - \mu\sin\delta\theta)}{[(a-l) - \frac{\mu d}{2}]} - (a-l)\sin\delta\theta - d\sin^2\frac{\delta\theta}{2}$$

..... (5.31)

Two-point contact

The equations for this phase have been handled by researchers using the Newton-Raphson method reducing to six equations with six unknowns, with several partial derivatives.

$$K_s\delta_s - F_{n1}(\cos\delta\theta - \mu\sin\delta\theta) + F_{n2}(\cos\delta\theta_2 - \mu\sin\delta\theta_2) = 0$$

$$K_\theta\delta\theta - F_{n1}[(a-l) - \frac{\mu d}{2}] + F_{n2}[(a - \frac{\mu d}{2})\cos(\delta\theta + \delta\theta_2) - (a\mu + \frac{d}{2})\sin(\delta\theta + \delta\theta_2)] = 0$$

$$K_{\theta 1}\delta_{\theta 1} + F_{n1}[(C_h + \mu C_s)\sin(\delta\theta - \delta\theta_1) - (C_s - \mu C_h)\cos(\delta\theta - \delta\theta_1)] = 0$$

$$K_{\theta 2}\delta\theta_2\cos\delta\theta_2 + F_{n2}[l\cos\delta\theta - d\sin\delta\theta - C_h(\sin\delta\theta_1 - \sin\delta\theta_2)$$
$$-C_s\cos\delta\theta_1 + \mu C_h\cos\delta\theta_2 = 0$$

..... (5.32)

$$\delta x + \frac{F_{n1}(\cos\delta\theta - \mu\sin\delta\theta)}{K_{x1} + (a-l)\sin\delta\theta} + r\cos(\gamma - \delta\theta_1) + d\sin^2\frac{\delta\theta}{2} = \Delta - \frac{CD}{2} - C_h$$

$$l\sin\delta\theta\cos\delta\theta_2 + d\cos\delta\theta\cos\delta\theta_2 - r[\cos(\gamma - \delta\theta_1) - \cos\gamma]\cos\delta\theta_2$$
$$-C_h(1 - \cos\delta\theta_2)\cos\delta\theta_2 + [l\cos\delta\theta - d\sin\delta\theta - C_h(\sin\delta\theta_1 - \sin\delta\theta_2)$$
$$-C_s\cos\delta\theta_1]\sin\delta\theta_2 - D\cos\delta\theta_2 - \frac{F_{n1}(\cos\delta\theta - \mu\sin\delta\theta)\cos\delta\theta_2}{K_{x1}}$$
$$-\frac{F_{n2}(\cos\delta\theta_2 - \mu\sin\delta\theta_2)\cos\delta\theta_2}{K_{x2}} = 0$$

..... (5.33)

Final solutions for δx and $\delta\theta$, considering both rotational and lateral compliances in hole are as follows [Hennessey 1982].

Chamfer crossing with lateral and rotational compliance

$$\delta x = \frac{K_q Dz}{\tan f[K_x a^2(1-\frac{d}{2a\tan(f-b)}) + K_q + \frac{K_x K_q}{K_{xl}} + \frac{K_x K_q}{K_{ql}}(C_v+Dz)[(C_v+Dz)\tan(f-b) + \frac{Dz}{\tan f} - C_h)] / \tan(f-b)]}$$

$$\delta\theta = \frac{K_x a Dz}{\tan f[K_x a^2 + \frac{K_q}{(1-\frac{d}{2a\tan(f-b)})} + \frac{\frac{K_x K_q}{K_{xl}}}{(1-\frac{d}{2a\tan(f-b)})} + \frac{K_x K_q}{K_{ql}} \frac{(C_v+Dz)[(C_v+Dz)\tan(f-b) + (\frac{Dz}{\tan f} - C_h)] / \tan(f-b)}{(1-\frac{d}{2a\tan(f-b)})}]}$$

..... (5.34)

One-point contact with lateral and rotational compliance

$$\delta x = \frac{K_\theta(\Delta - \frac{CD}{2})}{K_x(a-1)[(a-1) - \frac{\mu d}{2}] + K_\theta + \frac{K_x K_\theta}{K_{xl}} + \frac{K_x K_\theta}{K_{\theta l}} C_v(C_v - \mu C_h)}$$

$$\delta\theta = \frac{K_x(\Delta - \frac{CD}{2})[(a-1) - \frac{\mu d}{2}]}{K_x(a-1)[(a-1) - \frac{\mu d}{2}] + K_\theta + \frac{K_x K_\theta}{K_{\theta l}} C_v(C_v - \mu C_h)}$$

..... (5.35)

5.3 *Selection of Task frames*

To generate the task-frame trajectory during task execution, it is required that the position and orientation of the reference frame, as well as the relation between the task frame and its reference frame, are known at all times. Two kinds of task-frame definitions are proposed: the task frame may be selected to have either a fixed or a variable relation with respect to some user specified reference frame [De Schutter.et.al 1988].

In the peg-in-hole problem, the task frame is fixed to the end-effector (more specifically, to the tip of the peg), X, Y and θ are force-controlled directions, whereas Z is the position-controlled direction. In other applications, the task frame may be defined fixed with respect to the global reference frame, to the object frame (provided this frame is known), or to some other frame in the task.

In order to have an accurate algorithm, particularly, in order to avoid the need for orthonormalising the rotation part of the homogeneous transformation matrix at each control interval, it is desirable to describe the relation between the task frame and its reference frame as accurately as possible in the task-frame definition. In view of this, only three kinds of relative motions are permitted:

1. Rotation about some axis of the task frame.
2. Translation along some axis of the task frame.

3. Simultaneous rotation and translation along the same task-frame axis.

5.4 Assumptions for Shaft-Hole Orientation

Possible shaft-hole contact configurations are shown in Fig. 5.8 There could be significant angular position error in the shaft Fig. 5.8a, or in the hole Fig. 5.8c, or in both, the shaft and the hole Fig. 5.8b & d. In this research, the most common case of angular position error, i.e., in the shaft, is addressed.

The parts used for assembly were a splined shaft and a corresponding hole. It is assumed that the shaft and hole arriving at the work cell have been inspected and are free from manufacturing defects. It is also assumed that the coordinates of the hole are known with respect to the base frame of the robot and the hole has a fixed orientation. The force sensor being used is the determining factor in providing force information.

5.5 The SET-UP

An Adept robot was used for the research, it is a four degree of freedom SCARA robot, with high positioning accuracies. At the end of the robot arm, a wrist sensor was mounted to sense changes in force during different stages of the assembly process. To the lower surface of the sensor the PCD was mounted and to the bottom face of PCD, the gripper was mounted for holding the shaft. Hardware set-up is discussed in Appendices D and F.

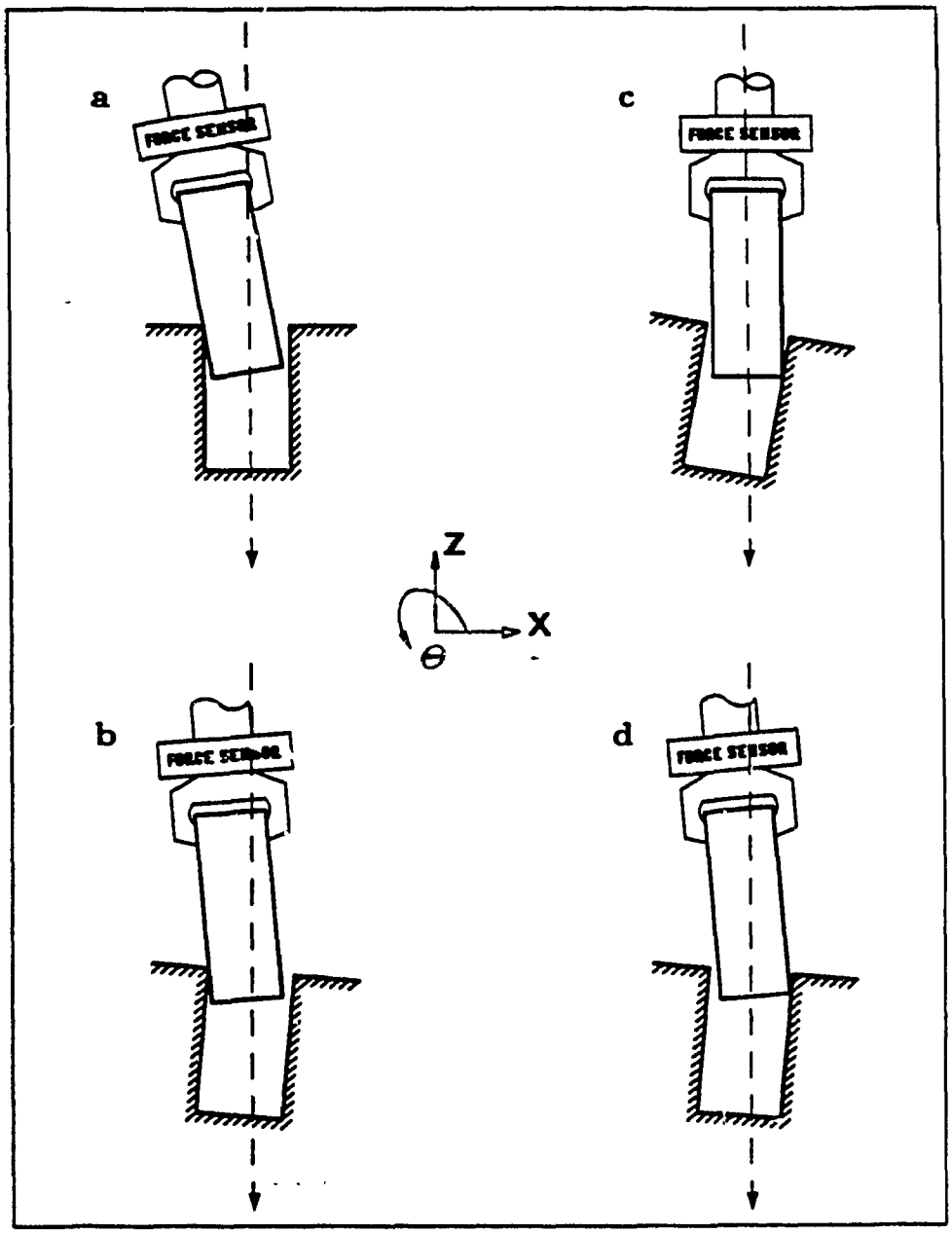


Fig. 5.8 Possible shaft-hole configurations [Vaaler 1991]

5.6 The Formulation

The shaft was brought to the vicinity of the hole, which is termed as gross motion by some researchers. At this stage the forces and moments are measured from the sensor. If the force in the Z axis is zero, it implies that the center line of the hole and the shaft are concentric and the robot generates position vectors in Z axis to complete assembly. During this process, Z position is monitored, the force in Z is measured for every δz change in distance, and the rate of change of force with distance in Z is monitored, to ensure assembly is complete.

Depending on the contact position of the shaft at the hole vicinity, there can be different assembly situations. If the force in Z is not zero, the algorithm calls for rotation of t! shaft up to a certain value in pre-selected steps (equal to half the spline angles), say in the clockwise direction, simultaneously checking forces in Z, and if a situation arises when the force in Z is zero, the algorithm automatically starts generating position vectors in the Z direction, to complete the assembly process, as in the previous case.

.

During clockwise rotation, if the shaft is not assembled until a certain pre-determined value, the robot rotates the shaft in the counter clockwise direction, to its original state and starts rotation in the counter clockwise direction in pre-selected steps, repeating the same procedure of force and distance monitoring. This counter clockwise rotation strategy works very well for keyed shafts and holes.

There could be a situation, when force in the Z direction at the start is zero, but as the shaft slides through the hole it might encounter a situation where it gets jammed. This jamming would normally occur beyond the compliance control limits. In such a situation the algorithm has been designed to retract the shaft and start the insertion from the beginning. If this condition occurs for the second time, for the same set of parts, the assembly task is aborted and an error message displayed to the operator.

Another situation during assembly is if the shaft misses the hole completely, for whatever reasons. The robot goes through a spiral search pattern for the next starting point and repeats the process. If the robot is unsuccessful after the spiral search, it then moves a set distance in the other axis, say Y-axis, and then repeats all these cases depending on the situation that it encounters. Fig. 5.9 shows the flow diagram of assembly algorithm.

There are standard termination conditions defined for this assembly process. One condition is exceeding a force threshold along the Common Assembly Path (CAP) combined with a sensed position indicate that the shaft and hole are in the neighborhood of being assembled. The Cartesian coordinates associated with each part were used to calculate the relative positions of the parts. However, various errors in robot and part location, did affect the accuracy of these positions.

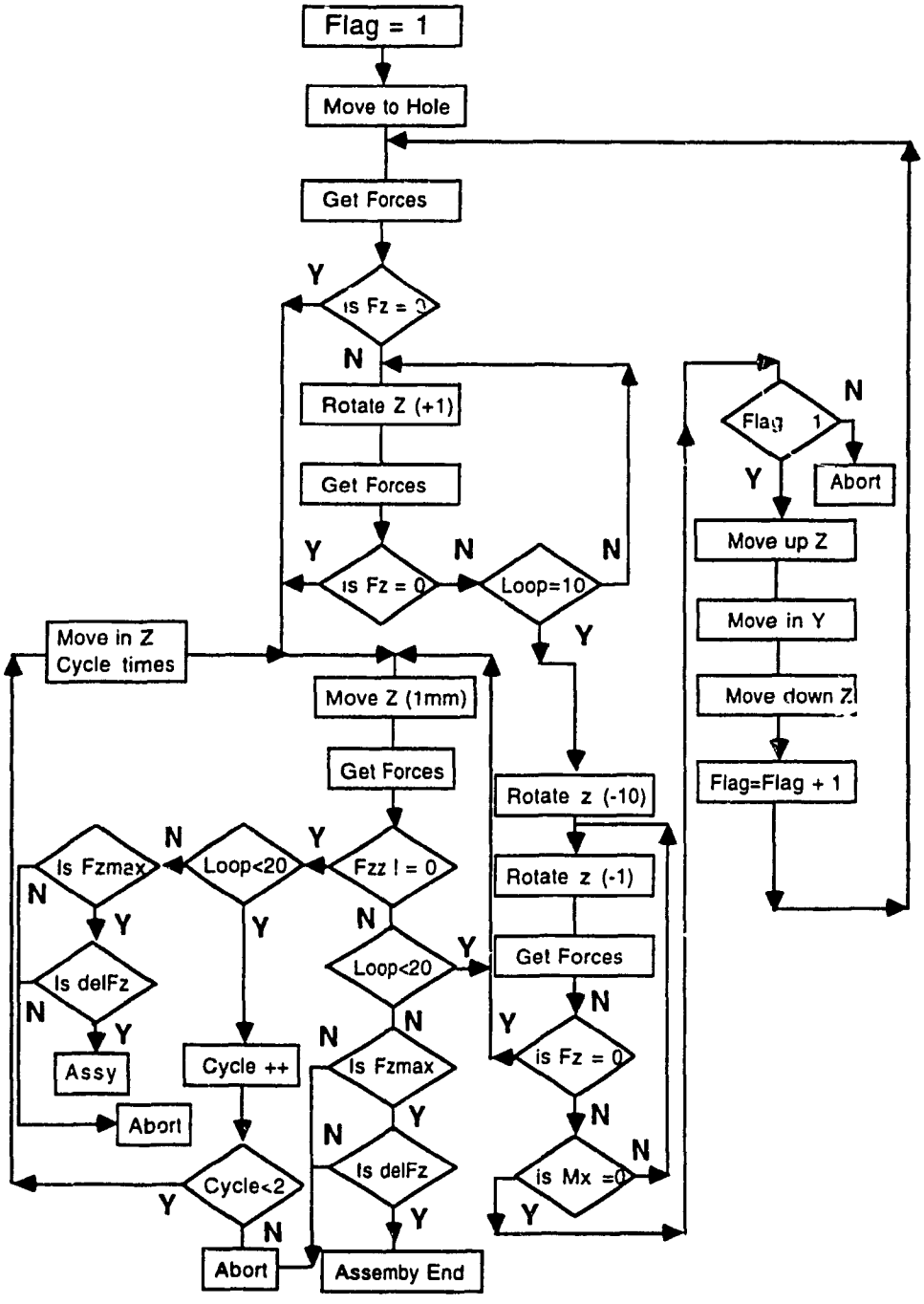


Fig. 5.9 Flow diagram of Assembly Algorithm

During the assembly task, the following parameters were monitored.

F_z	: measured force in Z direction
$\delta F_x/\delta x$	: incremental change in Fx for an incremental change in x
$\delta F_z/\delta z$	: incremental change in Fz for an incremental change in z
M_x	: measured moment in X direction
M_y	: measured moment in Y direction
X, Y	: position in X and Y
7	: to check if assembly is progressing and is completed

5.7 The Approach

Compliant motion with force feedback assumes physical contact between end-effector and the work piece or environment. Most references pay little or no attention to the control of transition phases between motion in free space and motion in contact with the environment [De Schutter.et.al 1988]. They especially underestimate the importance and the specific problems associated with the approach phase. The basis of this work is implicitly made on the following assumptions:

1. The approach phase is executed under position control: the robot moves at reduced speed while the force readings are continuously monitored; a conditional stop occurs upon detection of a predetermined force level. This strategy has been

termed as the "guarded move" [Will 1975]. However, a design rule for the approach speed as a function of the allowed collision force is not provided.

2. The move while out of contact is purely position controlled.

Identical schemes apply to both motion in contact and the approach phase; from a motion control point of view any distinction between the two phases is rather artificial.

A constant desired force is applied to the force controller. In the no-contact situation the actual force remains zero, and a steady-state approach velocity results.

In essence, this method yields a simple and acceptable solution to the approach problem because :

1. During the no-contact phase, the approach speed remains automatically limited.
2. An acceptable force response results after collision with the environment.

Instead of specifying a constant desired velocity, a constant force F_c can be specified in a velocity direction. Using force control in a velocity direction has advantages in some cases:

1. If an unexpected collision occurs, the collision force will be limited to F_c (apart from some limited overshoot).

2. Since the motion ends automatically after reaching a force F_c, this control method can be applied for a task with a force terminal condition.

To reduce computation time and increase robot response time, the transformations were carried out with respect to the robot end effector, although the position keeps changing during the assembly task. At the completion of the assembly, the base frame coordinates are calculated and compared to ensure correctness.

Incremental moves in translation are calculated as, $dA = \Delta A$

where,

$$A = \begin{bmatrix} n_x & o_x & a_x & p_x \\ n_y & o_y & a_y & p_y \\ n_z & o_z & a_z & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad \text{.....} \qquad (5.36)$$

is the differential translation and rotation transformation w.r.t. the base.

For transformations w.r.t. a coordinate frame { A } Fig. 5.10, where

$$\Delta = \begin{bmatrix} 0 & -\delta_z & \delta_y & d_x \\ \delta_z & 0 & -\delta_x & d_y \\ -\delta_y & \delta_x & 0 & d_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad \text{..... (5.37)}$$

and equivalent translation is $^{A}d = Tdx + Tdy + Tdz$

and equivalent rotation is $^{A}\delta = T\delta x + T\delta y + T\delta z$ ..... (5.38)

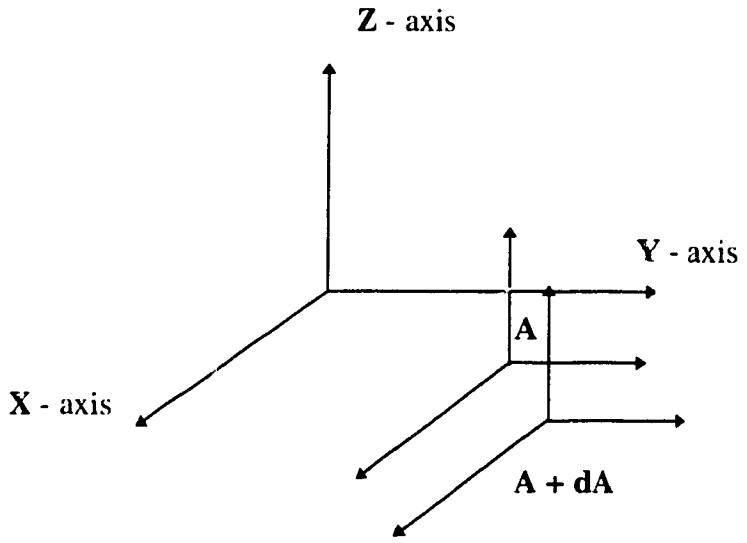


Fig. 5.10 Transformations w.r.t a coordinate frame [Craig]

where, $Tdx = n.[(\delta \times p) + d]$ $T\delta x = n.\delta$

$Tdy = o.[(\delta \times p) + d]$ $T\delta y = o.\delta$ ..... (5.39)

$Tdz = a.[(\delta \times p) + d]$ $T\delta z = a.\delta$

These basic incremental move matrices were incorporated in the algorithm to determine the orientation of the shaft and the hole as the assembly progressed.

5.8 Failures and Errors during assembly

There are 3 modes of failure that can be anticipated during the assembly process as shown in Fig. 5.11. A splined shaft is similar to a multiple shaft and hole assembly configuration before it is properly oriented, so monitoring the orientation was a major step in the assembly process.

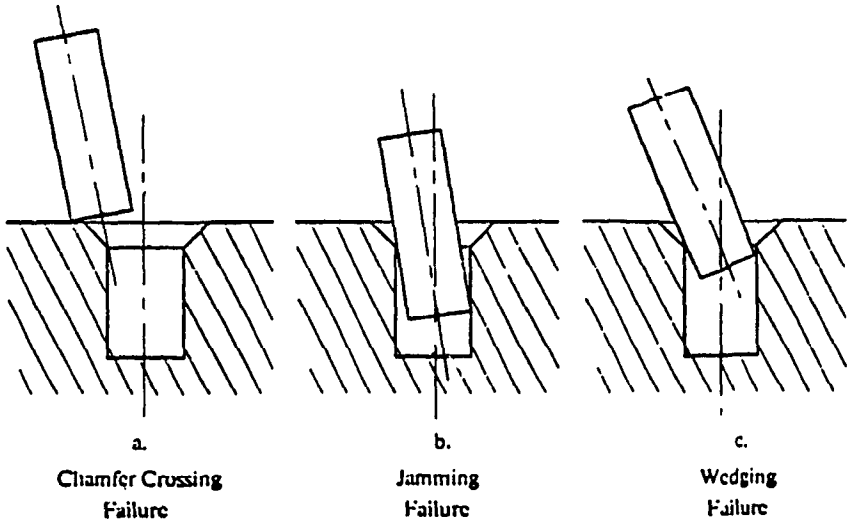


Fig. 5.11 Failures during assembly [Gordon 1987]

Various vectors corresponding to anticipated transformation errors during calibration and task execution phase are listed as follows.

Errors - Calibration phase

δT_w - error due to calibrated alignment of the sensor frame and the motion of the robot (world frame).

δT_{RAA} - error in robot position at the assembly approach

δT_{RS} - error in robot position at the sensor

δT_{PS} - error in location of object by sensor in sensor coordinate frame

δT_{PAA} - error due to initial alignment of part and assembly (due to user errors mainly)

Errors - Task Execution phase

δT_{RS} - error due to positioning of robot at the nominal sensing position

δT_{PS} - error in location of object by sensor in sensor coordinate frame

δT_{RAA} - error due to positioning of robot at the assembly position

δT_{RWWA} - vector corresponding to transformation of the apparent robot world coordinate system as the robot moves from a position near the sensor to a position near the assembly. This is primarily due to inaccuracies of robot's internal model of its kinematics.

5.9 Documenting the data

Documenting the data and storing the information for the validation of the algorithm was a major task in the work, as there were chances of missing relevant data in a pursuit to capture

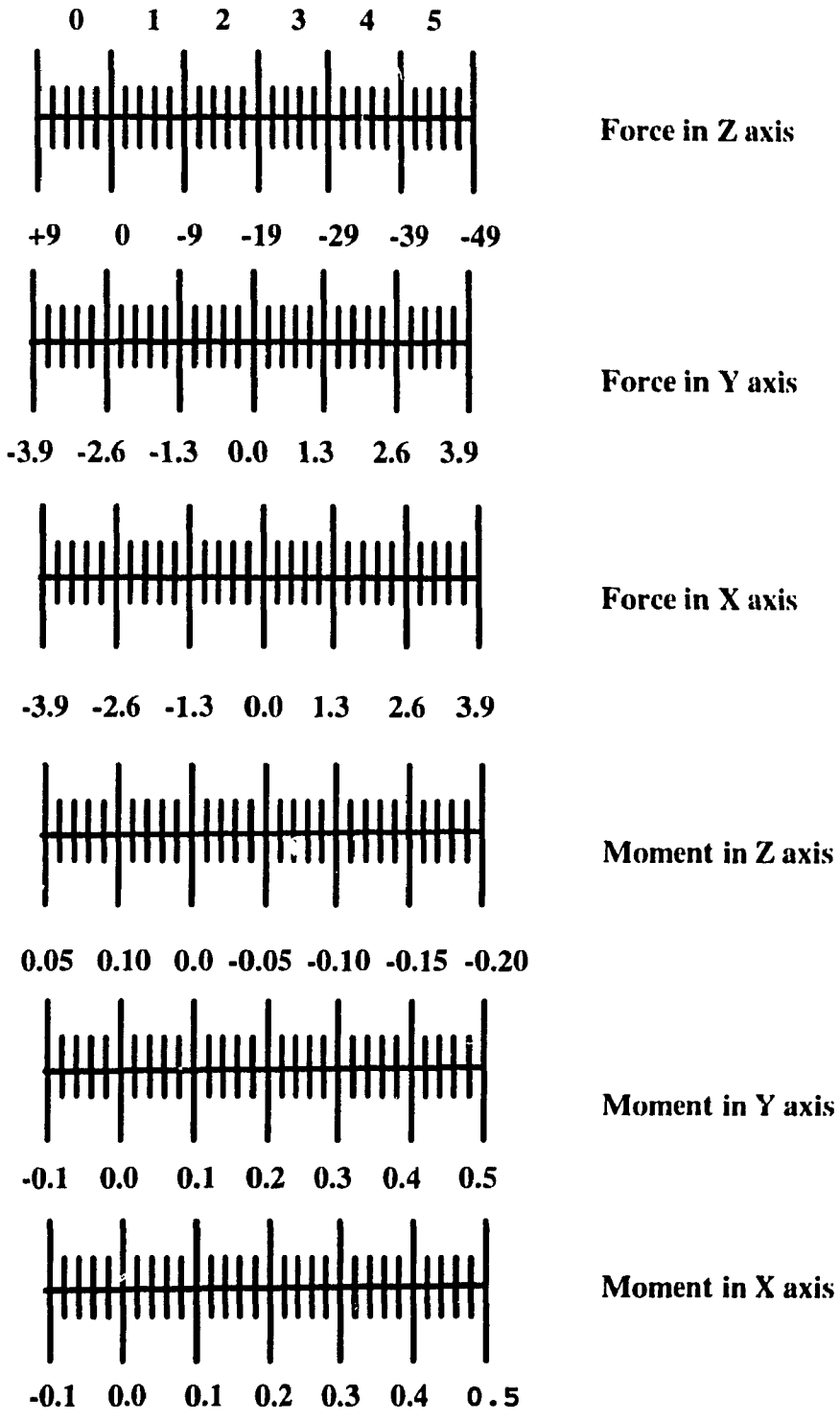


Fig. 5.12 Sample Force (Newtons) and Moment (N-m) discretised steps

a broad spectrum of information. When experts perform assembly tasks, they resolve the task into several steps [Ahn.et.al 1991, Vaaler 1991, Hara.et.al 1992] depending on the task complexity, this formed the basis to approach the assembly problem defined in this research. Experiments were conducted to determine these steps Fig. 5.12 shows some examples of these. An important criterion in determining the range of this resolution was to check the pre-determined steps incrementing the robot position. These incremental moves were in turn dependent on the resolutions of hardware being used, their stiffness and the defined threshold ranges. As there were no standards available, and this data varied with hardware, several experiments had to be conducted to adjust these ranges, as has been done by other researchers while working with other problems in this field. Care was also taken not to have a wide range, as it would have created some redundant orientations that the robot could not have reached, but the controller would have had to process, thus increasing the computation time and the time needed to browse through all steps in the range. As such, a trade off between information content with good resolution and learning time was made [Vaaler 1991, Ahn.et.al 1991].

The increments in the range were set based on the hardware noise, system gain resolutions and drift and these would have changed if robot errors mentioned above were non-existent.

An important aspect seen from Section 5.2 is that, evolving a model for implementation in assembly requires identifying and modeling the parameters, like friction, compliances

in the hole and the shaft and assessing the Common Assembly path which could be a difficult and complex exercise. As such, these models have to be approximated to a large extent. These approximated models do not have the flexibility and cannot be used for the 'real life' application for various reasons. However, to calculate the incremental angular and linear displacements during the evolution of the algorithm the model was used to benchmark certain aspects of the assembly task. On the other hand, other techniques - such as Machine Learning - do not require the development of an accurate model or the precise determination of system parameters. Broadly speaking Machine Learning is a pragmatic solution for a production environment.

Chapter 6

The Algorithm

Before plunging into the details of the algorithm, a brief introduction is given about the artificial intelligence time line, machine learning and the model.

6.1 The Artificial Intelligence Time line

The 1950s and 1960s was an era of Symbolic Processing and Algorithms and Heuristic Search [David 1995]. Heuristics are rules of thumb that limit the size of the space searched by getting the computer in the vicinity of the solution from the outset. The one weakness of the heuristic search is that, although it locates a solution faster, the solution is not necessarily the best solution. However, a good heuristic search gives up a little in finding a solution, while gaining a great deal in limiting the size of the search area.

The 1970s were characterized by the development of expert systems, which resulted from the realization of scientists that intelligence would not be achieved by searching through general space. They began to see the need for feeding task specific information to the computer that would give it an experience base relating to a specific task domain. This resulted in the development of expert systems.

Expert systems have two key components : (1) an inference engine and (2) an experience or knowledge base. The first component controls the application of information contained in the second component. The development and use of expert systems raised questions about the way knowledge is represented.

Improving the ways knowledge is represented so that it is more explicit, yet more concise, characterized the 1980s. The 1990s has so far seen a rapid growth in the use of artificial intelligence in automated assembly and other manufacturing applications as the concept of machine learning evolves.

It is generally accepted that a key characteristic of human intelligence is the ability to learn. For artificial intelligence to reach its potential, the concept of MACHINE LEARNING must be fully developed. Machines learn in one of three ways :

1. parameter adjustment

2. concept formation

3. evolution of structure

Parameter adjustment is the most basic of the three approaches to machine learning. It involves adjusting the values of the parameters of a predetermined representation. It can affect only the values and not the structure of the predetermined representation. Concept formulation involves grouping related objects into categories or groups. Evolution of structure makes use of neurocomputing. This involves the parallel activity of elements that are able to communicate the results of computations among themselves.

6.2 Machine Learning overview

This section enlightens the reader with the purpose of using machine learning in a particular task situation. The question that comes naturally to mind is why should machines learn? The first fact is that human learning is very slow and the second distinctive feature is, that there is no copy process. In contrast, once a program has been debugged in a computer, several copies can be made. An algorithm only has to be invented once and can be used forever. There is a strong reason to search for machine learning programs that will avoid the inefficiencies of human learning, although in principle, such programs cannot be constructed. The difficulty may be intrinsic in the task; human learning, though slow, may be close to optimally efficient [Herbert 1983].

AI has two goals, the first one is directed towards getting computers to be intelligent and do smart things so that humans do not have to do them. The other goal is also directed at using computers to simulate human beings, so that it is possible to know how humans work and perhaps help them to be a little better at their work.

Learning is any change in a system that allows it to perform better the second time on repetition of the same task or on another task drawn from the same population. The change should be more or less irreversible.

'Machine learning' is directed towards 'machine discovery', that is to say, many machine 'learning' systems are also discovery systems, they discover new knowledge that they subsequently retain [Herbert 1983].

Giving a machine the ability to learn, adapt, organize, or repair itself are among the oldest and most ambitious goals of engineers and scientists. In the early days of computing, these goals were central to the discipline of cybernetics. Substantial progress has been made in developing techniques for machine learning in highly restricted environments. Each of these, however, is tailored to a particular task, taking advantage of the assumptions and characteristics associated with the specific domain. The search for efficient, powerful, and general methods for machine learning has come only a short way [Buchanan.et.al].

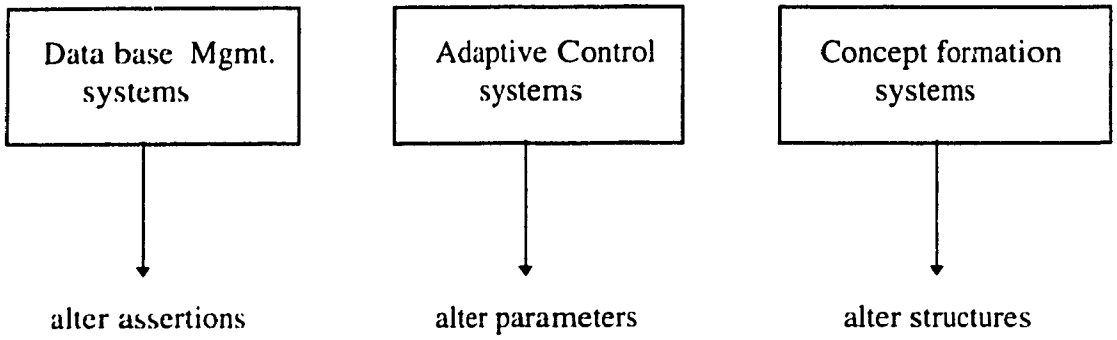


Fig. 6.1 Spectrum of Learning Systems [Buchanan.et.al]

The term "Learning system" is very broad based, and often misleading. In the context of this research, a learning system is considered to be any system which uses information obtained during one interaction with its environment to improve its performance during future interactions. This rough characterization may include man / machine systems, in which humans take on active roles as required functional components. In some systems there is a continuous interaction with the environment, with feedback and subsequent improvement. In other systems there is a sharp distinction between the interactions that constitute training and subsequent performance or predictions with no further training. Another way of differentiating between various learning systems is on the basis of what kinds of alterations they perform.

There are broadly three distinct approaches to machine learning and adaptation :

1. The adaptive control approach
2. The pattern recognition approach
3. The artificial intelligence approach

As this research uses artificial intelligence (AI), this approach is only dealt with. All AI learning programs written to date have strong limitations on their generality. Some apply to just one kind of problem, others work with several types of problems within a larger class defined by the representation of objects and their relationship in that domain.

Various levels of sophistication in learning systems have been defined :

1. learning by being programmed
2. learning by being told
3. learning from a series of examples, and
4. learning by discovery

There is a gradual shift in responsibility from the designer / teacher to the learning system / student. At the highest level, the system is able to find its own examples and carry on autonomously; at the lowest level the system is learning only in the sense that a programmer is explicitly programming it to do something.

6.3 Learning System Model

This section is concerned with a sample functional model that is useful for characterizing, comparing, and designing learning systems (LS's). Many of the functional components of a LS are essential to intelligent problem-solving systems in general [Smith.et.al 1977]: that is, learning (induction, concept formation, etc) is problem solving of one kind, which means that AI problem-solving methods and representations can be expected to apply to this task as well as to others.

The environment from which training instances are drawn, and in which an LS operates, may have a profound effect upon the LS design. LS environments can be divided into two major categories : those that provide the correct response for each training instance (supervised learning) and those that do not (unsupervised learning). Supervised learning systems operate within a stimulus-response environment in which the desired LS output is supplied with each training instance. Unsupervised LS's operate within an environment of instances for which the correct response is not directly available.

Environments can be further categorized as 'Noise-free' or 'Noisy'. Noise-free environments, provide instances paired with correct responses which the system assumes to be perfectly reliable. Noisy environments, on the other hand, do not provide such perfect information, as is usually the case when empirical data are involved.

The components of a LS model are shown in Fig. 6.2. The 'performance element' is responsible for generating an output in response to each new stimulus. The 'instance selector' selects suitable training instances from the environment to present to the performance element. The 'critic' analyzes the output of the performance element in terms of some standard of performance. The 'learning element' makes specific changes to the system in response to the analysis of the critic. Communication among the functional components is shown via a 'blackboard' to ensure that each functional component has access to all required system information.

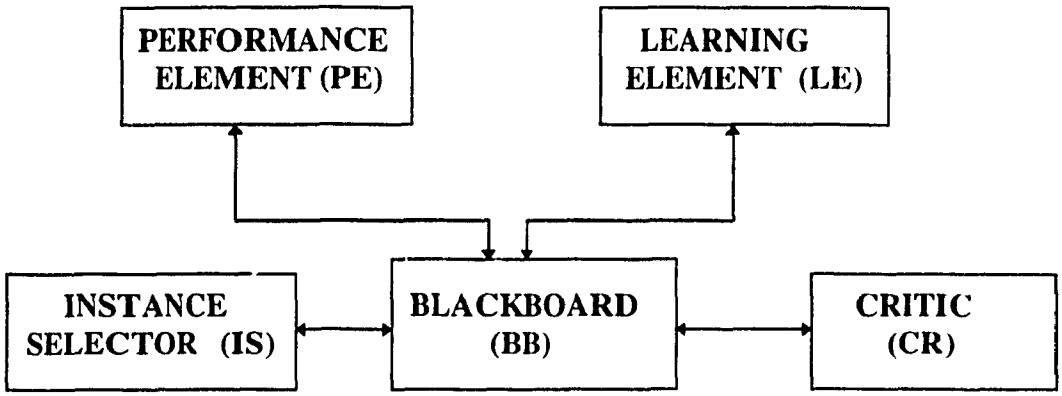


Fig. 6.2 Components of a learning System [Buchanan.et.al]

Finally, the LS operates within the constraints of a 'world-model' which contains the general assumptions and methods that define the domain of activity of the system.

6.4 Proposed Machine Learning Algorithm

Certain aspects of the machine learning algorithm have evolved from the work of various researchers earlier [Vaaler 1991, Smith.et.al 1977]. The Machine learning algorithm in this research is divided into four components [Buchanan.et.al]; Initiator, Executing Element, Censor, and Transformer. The initiator initializes and starts the system. This corresponds directly with INITIALIZE_SYSTEM (Block 1, Fig. 6.3). The Executing element is the output of the algorithm and is responsible for generating a control action, in this case a move in X and θ (Block 3, Fig. 6.3). The Censor and the Transformer are embedded in the assembly algorithm (Block 6, Fig. 6.3). The Censor evaluates the quality of the data and or selects a subset of relevant data from this set. This evaluation is used by the Transformer to transform the learned information into a form usable by the Executing Element. In brief, the functions of various elements are as follows:

INITIATOR	- Initialize System
EXECUTING ELEMENT (EE)	- Responsible for generating control action
CENSOR	- Review and / or Select
TRANSFORMER	- Changes learned information for use by EE

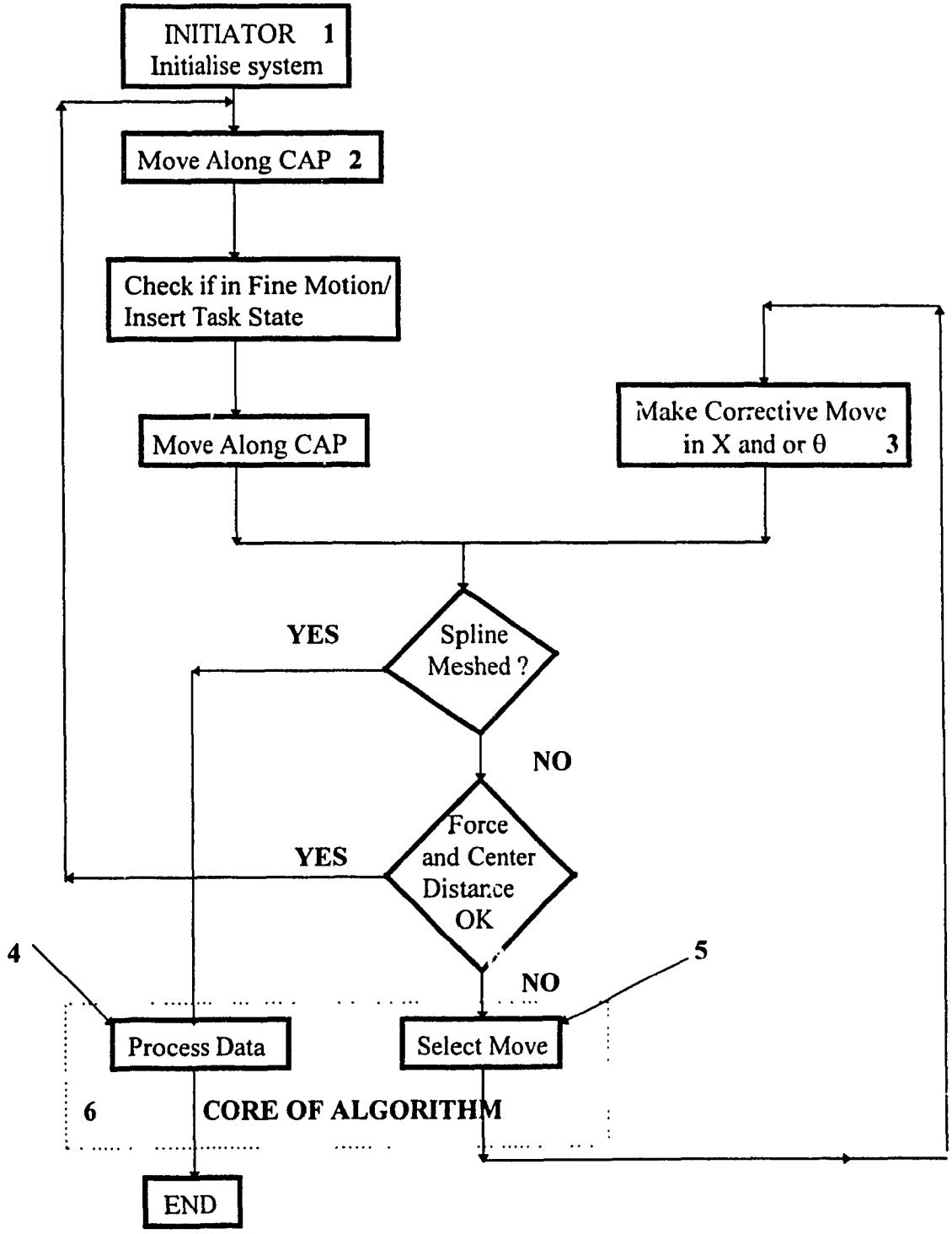


Fig. 6.3 Proposed Machine Learning Algorithm

6.5 The Approach

The assembly task was divided into two sub-tasks, the Search Task and Insert Task. The Search task is the same as Fine motion as termed by some researchers and the Insert task is the actual shaft-hole insertion process. To be in line with the literature, the term Fine motion will be used in this thesis. During Fine motion the learning is iterative, as the force and moment information monitored and recorded is related to some a-priori defined actions. In the Insert task learning consists of six actions mainly, and mapping the relationship that relates the force and moment information to the six types of action. The moves are stored in a data base in pairs of response from sensor and the corresponding move.

A binary tree type data base is used. A set *(D)* was defined that consists of all the real values and all ordered n-tuples. An evaluating function *(E)* was defined, which helps discriminate between the present and previous moves. An information space *(I)* was defined as a n-dimensional vector space which had all records of forces and moments from the sensor. Another n-dimensional vector space *(M)* was defined for the type of moves to be performed for certain values of forces and moments. Therefore, I and M are subsets of D and

i is an element of I representing force and moment information in the information space I and is an input to the learning system.

m is an element of *M* representing a correct move in vector move space and is also an input or output in the learning system.

INITIALISE : To start with, the data base is empty as such a move is randomly selected or can be taught a-priori. The random move was selected from a normal distribution. This distribution was used to generate random initial position errors and represent a cumulative effect of the shaft, hole and robot position errors.

EXECUTING MOVES : Increment *i* by 1 and the new M_{now} was input, the shaft was moved along this assembly path by this increment. The increment depends on the robot encoder resolution, system stiffnesses and the amplifier. A force threshold condition was also defined in X, Y and Z axes to ensure that the robot sensor and system do not damage the shaft-hole, the sensor or the robot. If this force limit is exceeded, program goes to Block 6 of Fig. 6.3. The algorithm now selects or generates a response (in direction and distance) corresponding to the force limit that was exceeded. The robot executes that move and moves along CAP to complete the assembly as per the termination condition or the robot encounters a situation in which it fails and aborts assembly.

TERMINATION CONDITIONS : For the purpose of this work three termination conditions were defined. As the shaft moves into the hole, Z position was monitored, a limit was defined within the maximum Z error value in the vicinity of the bottom of the

hole. Another condition defined was that, the force in Z reached a value of 3 times that of the maximum force that could be generated in one or two-point contact. The third and final condition was to monitor and analyze the rate of change of force in Z direction with every increment in Z, this ensured that the algorithm had achieved successful assembly and was not faced with a jamming or wedging condition.

CENSOR (REVIEWING) : After the robot had made an incremental move it was necessary to check if the move was a success. In order to check this, the evaluating function was evaluated. If the condition was satisfied, the move was a success. If not, the program returned to Block 6 of Fig. 6.3 to select a move.

After every move the forces from the sensor were recorded and compared to the threshold values. During the moving process, if the force was above the threshold and (X,Y,Z) were within the limits, a move in θ was made until forces went below the threshold, if still not successful, the move was abandoned and a move in another direction was tried. If forces were below threshold, the program continued to move the robot in that particular direction, until the forces were again more than the threshold value and the process repeated.

TRANSFORMER : When the assembly was complete, the pairs of all moves and the corresponding force and moment values were recorded and simultaneously, the linear

distance X and angular changes $\Delta\theta$ were also recorded. The recording of X and $\Delta\theta$, helped in defining branching points in future moves in the decision tree. Every move made is added to previous moves. In future assemblies if any of these conditions is encountered, the robot automatically switches over to the best move as a result of learning, thus reducing time and resulting in a good move. As learning advances, eventually there will be a state achieved in which all moves may have been made and the output of all incremental moves will be based on the previous assembly trials. At the end of an assembly run, the data was updated and a global information rack formed, which had the improved data content with all the best moves.

6.6 Content of Learning

Broadly speaking, the contents of learning during the Fine motion and Insert task were similar. The input to the learning system consisted of the force moment information, consisting of the 6 components obtained from the force sensor $I = (F_x, F_y, F_z, M_x, M_y, M_z)$ at the instant the shaft contacts the periphery of the hole, representing the forces and moments in X, Y and Z, respectively.

During fine motion, the main purpose was to have corrective motion so as to eliminate the position errors, for which it was necessary to take a corrective action in the opposite direction to the position error. For a linear move, the distance to be moved was in step-wise increments, equal to the robot positioning accuracy. Also, the number of step-wise

increments for corrective moves was given to the robot in advance depending on the magnitude of the position error and other hardware limitations.

Rotational motion was another aspect that was to be considered and was in effect during 2-point contact. This had to be specified with its direction and rotation angle. As seen in the previous chapter, rotation is either in the clockwise or counter clockwise direction about the centre of the shaft. The shaft was allowed to rotate 10 degrees in clockwise direction and then in counter clockwise direction depending on the situation encountered, thus the learning system could have several values in the information space *I*.

In effect, the mapping relationship of the force and moment information that had been monitored in an assembly task with respect to the respective actions was established by the learning process. Therefore, it was very important to know the exact resolution of the force sensor, the PCD, the amplifier, the robot encoder and other stiffnesses, as any deviation in these would have affected the learning process, even for the best developed algorithm.

During the insertion task, corrective moves were mandatory to avoid excessive forces between the shaft and the hole. As discussed, threshold values for each axis of the force sensor were pre-set. Whenever the force value exceeded the pre-set values a corrective move was called for. The corrective moves for the insertion task were the same as in the

case of fine motion. Linear corrective moves in ± X and ± Y were carried out and for rotational moves about the Z axis with a pre-set value of ±1 degree.

6.7 Actual Learning

As discussed above, learning is essentially the mapping of force / moment information with respect to position vectors. A mixed flavor of Learning by Examples and Learning by Induction are incorporated in this research. The actual learning process is divided into two steps. The first step is to acquire the information from the sensor and the second step is to assign weights to this data and evaluate a function.

Weights W_1, W_2, ..., W_n are assigned to the discretised force values, Fig. 5.12 in Chapter 5, that the robot encounters during each move. For instance, if the robot senses -10 N in Z axis, the discretised force value will be 2, Fig. 6.5, this force value will be assigned a weight W_2 (say). The highest value of the weight is assigned to the lowest discretised value, implying that the force state that has the lowest value is in the nearest vicinity of the CAP. An arbitrary range of 0.5 to 1.0 was selected for the weights.

At a particular instance in assembly, the force / moment values are checked in the Information Space I, weights are assigned and the force direction that has the highest weight is selected by the algorithm. Now the Move Space M is checked for the next move, if there is no suitable move in this space, a new position vector is generated and the

robot moves along that direction. This not only ensures that the best move is made, but also that the robot is going along the CAP. There is a chance that the robot does not learn a globally optimal path. To avoid this problem, the following aspect is included in the learning process. If, for example. there is a case when at some point the robot finds that there are some similar situations that it had previously visited, the algorithm scans the data, and determines the E-function for all these situations, the one with the lowest value is selected, and a corrective move is selected either from move space M or generated, as discussed earlier. This way the robot either decides to learn from the examples that already exist in its database or generates a move by induction / evaluation along the CAP. The best moves are then updated for future assemblies. This approach seems reasonable and later during experimentation gave encouraging results.

To make the learning process robust, the incremental move is also selected based on the search tree illustrated below. When the robot encounters a particular situation, after the calculation of weights, the algorithm searches the Move space M as per the binary tree Fig. 6.4, thus checking for best moves in X and or θ.

The evaluation function in the insertion task is defined as

$$E = (W_1*F_x^2 + W_2*F_y^2 + W_3*F_z^2 + W_4*M_x^2 + W_5*M_y^2 + W_6*M_z^2)^{1/2}$$

where F_x ,F_y ,F_z,M_x,M_y,M_z are assigned values 0 to 5 based on the range shown in Fig. 6.5, from the regions that have been discretised for easier learning. If for example the

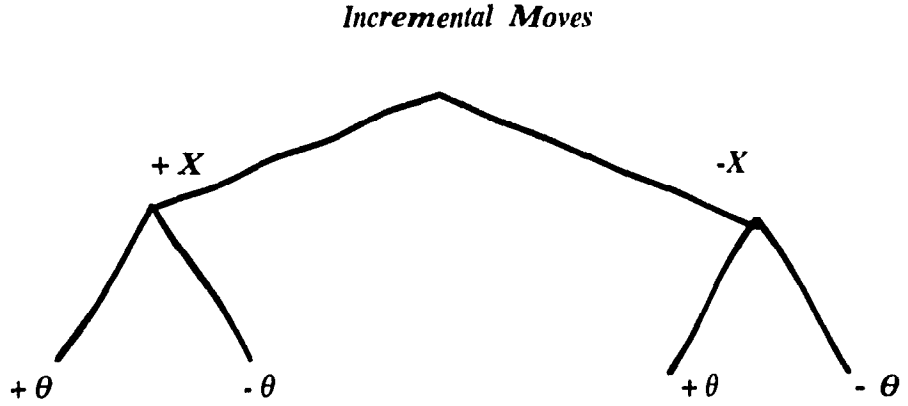


Fig. 6.4 Search Tree for Corrective Moves [Vaaler 1991]

force in Z for a particular step was 8 Newtons, a value of 0 is assigned, for a moment of -0.15 N-m in Z axis a value of 4 was assigned and so on.

The executed function is a

Success,	if	E (after) < E (before)
Failure,	if	otherwise

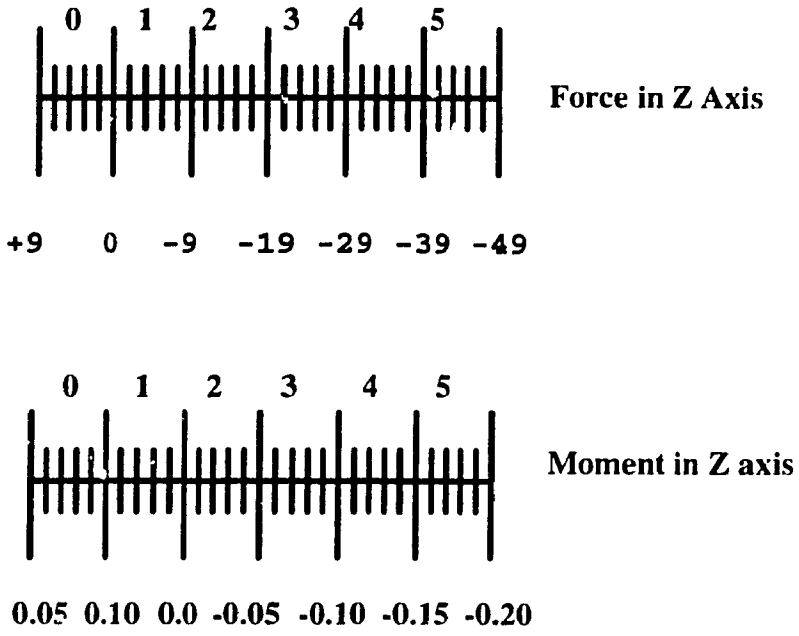


Fig. 6.5 Force / Moment sample discretised steps

6.7.1 Learning the Good Move

To elaborate further on the actual learning, further explanation of an assembly attempt will be discussed. During an assembly attempt, the robot visits several locations as it proceeds through the task. For each location visited, the X, Y, Z positions are recorded, corresponding forces and moments for each axis are also recorded. Now, the evaluation function for each of these locations is estimated. For instance if the robot in an assembly attempt has visited locations L1 to L20, it has corresponding evaluation functions E1 to E20. The assembly could have taken place in a sequence L1-L4-L7-L12-L16-L19, which is decided based on the lowest value of the evaluation functions. Other locations in the assembly process may have higher values as such their contribution is not considered in the

assembly process. The algorithm then, stores this sequence along with the estimation values in the same sequence E1-E4-E7-E12-E16-E19.

As the assemblies are repeated over a period of time, the database acquires similar moves and corresponding E-functions. For every new assembly attempt, the algorithm browses through its database. If again a location is visited, which the robot may have visited earlier, it automatically switches to the earlier sequence and completes the assembly instead of generating vectors all over again. Unsuccessful assembly trials are not stored in the data base, which helps reduces the learning time and the computation as well. There could be an instance where the robot encounters the following situation. For instance assembly attempt 27 has a sequence L3-L5-L8-L9-L12-L16-L20 and assembly attempt 38 has a sequence L2-L4-L7-L9-L11-L14-L18. The robot is attempting assembly no. 43 and is half way through, i.e., it has passed L1-L4-L6-L9, for its next move the algorithm scans the data base and realises that it could follow sequence 27 or 38 to complete the assembly. At this point of time, the algorithm compares the total value of both evaluation functions for the remaining portion of assembly, and the one with a lower value over rides the other. This way it is ensured that the best move is selected for assembly.

6.8 Hybrid Compliance Control Strategy

The control strategy used in this case is a combination of an implicit Hybrid approach and a PCD, and is called the Hybrid Compliance Control (HCC) strategy. As discussed in the

literature review and in Chapter 4, there are different types of control: passive, active and hybrid. In this research a passive compliance device which is non-tuneable for a particular task is used in conjunction with force feedback as shown in Fig. 6.6. The blocks enclosed within the dotted region constitute the software aspects of the research and others, excepting the robot, are the hardware that was developed and can be modified depending on the application.

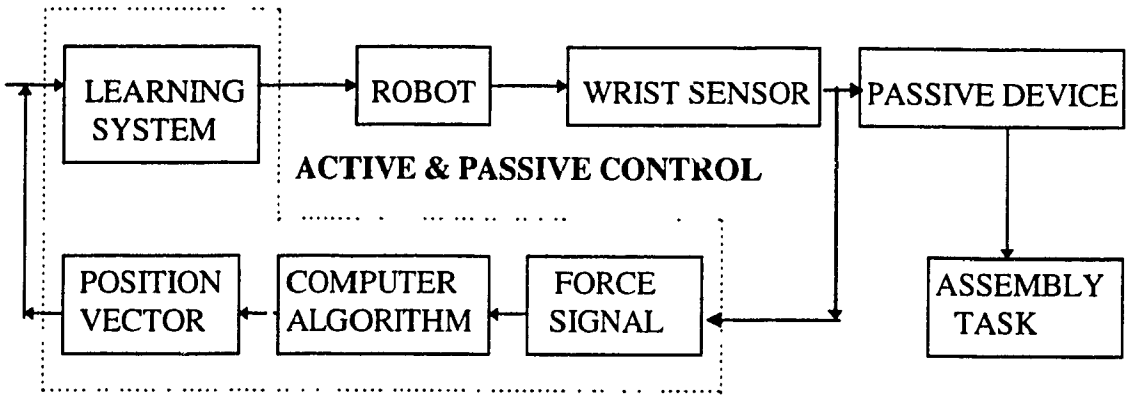


Fig. 6.6 Hybrid Compliance Control Strategy for the assembly task

The robot approaches the desired CAP through the influence of the contact forces on the PCD as well as by sensing the direction of move, based on data available and position vectors generated in the algorithm as a result of force information from the wrist sensor. This has the advantages of using the passive compliance and the simultaneous generation of the vectors, thus resulting in faster and more precise assembly. This was done with a view to hasten the robot learning process, and increase the success rate of assemblies.

6.8.1 Advantages of Hybrid Compliance Control

At first the concept of having both the passive and active compliance control in the HCC strategy appeared to be introducing redundancy. On implementation it was found that the HCC had several advantages with significantly low cost impact. Researchers have established that a control system can be attenuated up to certain limits for any application and after that limit is exceeded, mechanical aspects of the system have to be fine tuned. In a robotic work cell the inherent errors can be modeled and accounted for, within limits only. The manufacturing irregularities are random in nature and are difficult to characterize in a control algorithm. These uncertainties are accommodated by the mechanics of such a passive device. Another advantage using this concept is that in the event of failure of either of the compliance control modes, the other mode exists as a fallback and does not interrupt work operations, thus preventing loss of revenue and down time. For multi-operation jobs the degree of compliance for each operation is different, depending on the needs the compliance can be manipulated either by switching between active and passive modes or by having both the modes in effect and tuning them to the levels required. This can be an effective strategy for instance, when deburring and assembly of same components have to performed by the robot in the work cell

The option of using neural networks in the control as well as the learning process was looked into, but there were certain aspects of neural networks that did not appear encouraging. In the neural networks training process, the operator is unable to decipher the

relationship between the inputs and outputs, which is a black box in this field to date. In a neural network scenario, a set of inputs results in a set of outputs, without being able to access the training process. This was considered to be a major handicap in the present work, as mapping of forces and the position of the robot arm needed close monitoring while the learning process was / is being established. The discretisation steps need to be varied and the optimal settings to be arrived at, for best results. With the use of neural networks, another arbitrary variable was the number of hidden layers to be considered for a particular situation, there are no specific criterion for this setting, although some sketchy theories are being propounded in the literature.

Chapter 7

Simulation and Experiments

This chapter discusses the theoretical simulations and the experiments carried out. Various assumptions made to carry out the simulations have been listed. The codes for the simulation written in C are also attached in appendix G, for sake of completeness. Windows working environment for the entire process was used.

7.1 Simulation Assumptions

There were some assumptions made for the simulation process. The modeling was done using a normal probability function and a random variable with binomial function . This

ascertained some aspects of the behavior and pick the most appropriate visiting states of the robot. The major assumptions that were made are:

a - The splined shaft and the hole have sufficient clearance to mesh.

b - The shaft and hole are chamfered.

c - The diameter of the shaft / hole does not affect the results.

d - Changing the geometry of the shaft-hole affects learning rate.

e - The simulation carried out for two external gears mating, can be extended to internal gear mating.

e - Meshing of splined shaft in the hole is equivalent to assembling multiple shafts in multiple holes.

f - The probability distribution for starting moves is Gaussian.

g - For the random varaibles, values were generated assuming certain a-priori conditions taken from experts at assembly shops.

h - Non-linearities in the robot (i.e., kinematic errors) do not affect the actual assembly process, or the initial search process.

i - Probabilities of not meshing the first time and not meshing the second time were assumed to be different. Simulation was also carried out for equal probabilities of meshing during first and subsequent assembly attempts.

7.2 Iterations

Several trial runs were performed using different probabilities for every assembly attempt. The trends from results of these iterations have been incorporated in the main assembly algorithm for experimental work.

A sample simulation output is shown in Fig. 7.1. The details of a sample simulation trial are listed below with probabilities and sample size.

TRIAL : A random shaft rotation direction was considered and the probabilities of NOT MESHing were assigned arbitrary values to form a reference point. The probabilities for subsequent trials were then incremented.
The shaft is first rotated in one direction only, i.e., clockwise.
The Probability of NOT MESHing the first time taken as 95 percent.
The Probability of NOT MESHing the second time taken as 70 percent.
Sample run for 39,000 assembly attempts [(5 days) x (20 hours x 60) minutes x (6.5 assemblies, i.e., 6 to 7 per minute)].
The plot shows that the angle of rotation required for the shaft to align into the hole lies in the region of 0 to 5 degrees.

Similar trials were done on the computer with different probabilities and direction of shaft rotation and the results obtained were found to follow a trend. The results are tabulated in

Fig. 7.2. Probability distributions for the start-up stage were also changed during these trials, but no significant change was observed in the simulation results. Probabilities of NOT MESHing were chosen in a way to encompass a wide range of situations in actual assembly.

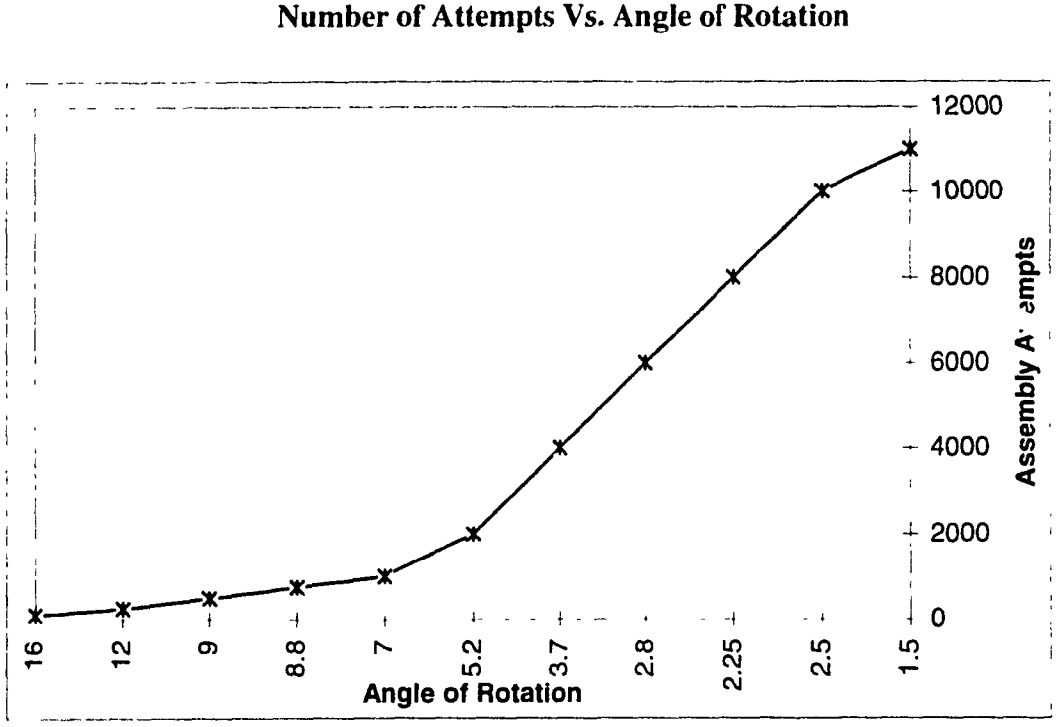


Fig. 7.1 Sample Simulation Output

Trial Number	Rotation Direction	Probability NOT MESHing 1st Time	Probability NOT MESHing 2nd Time	Maximum Rotation Range from Simulation
1	CW	0.95	0.70	0 to 5
2	CW	0.90	0.70	0 to 7
3	CW	0.80	0.70	0 to 7
4	CW	0.70	0.70	0 to 6
5	CW	0.70	0.80	0 to 5
6	CW	0.70	0.90	0 to 7
7	CW	0.70	0.95	0 to 5
8	CCW	0.70	0.95	0 to 5
9	CCW	0.70	0.90	0 to 6
10	CCW	0.70	0.80	0 to 6
11	CCW	0.70	0.70	0 to 8
12	CCW	0.80	0.70	0 to 6
13	CCW	0.90	0.70	0 to 6
14	CCW	0.95	0.70	0 to 5

Fig. 7.2 Simulation trials results

7.3 *Experiments*

Series of experiments were performed to evaluate the performance of the algorithm. Experiments were also performed to determine the incremental steps for moving the robot during assembly tasks.

7.3.1 State Resolutions and Increments

In order to determine the smallest step size and resolution in X, Y, Z and θ for the assembly process, experiments were carried out. A simple shaft 20 mm in diameter and a 20.05 mm hole and a splined shaft (SAE 6B) of 7/8 inch diameter with hole clearance of 0.05 mm were taken to carry out these increments and resolution tests. During the dry run, the force in X, Fx, the force in Z, Fz, the incremental change in Fx with a change in x, and an incremental change in Fz with a change in z as well as moments in X and Y were measured. These were repeated 15 times for rotations from 0 to 7.5 degrees, in steps of 0.5 degrees rotation in clockwise direction and the same was repeated in counter clockwise direction. There was no significant difference observed between the two cases, from experimental results. The slight variation was probably due to the settling time required for the amplifier and other electronics in the system. The drift in values can be attributed to repeatability errors in the robot. Figs. 7.3 and 7.4 show the outputs of X, Y Forces and moments, respectively and Figs. 7.5 and 7.6 show the Z force and moment, respectively for a sample test for determining resolutions. As a result of the experimentation following decisions were taken:

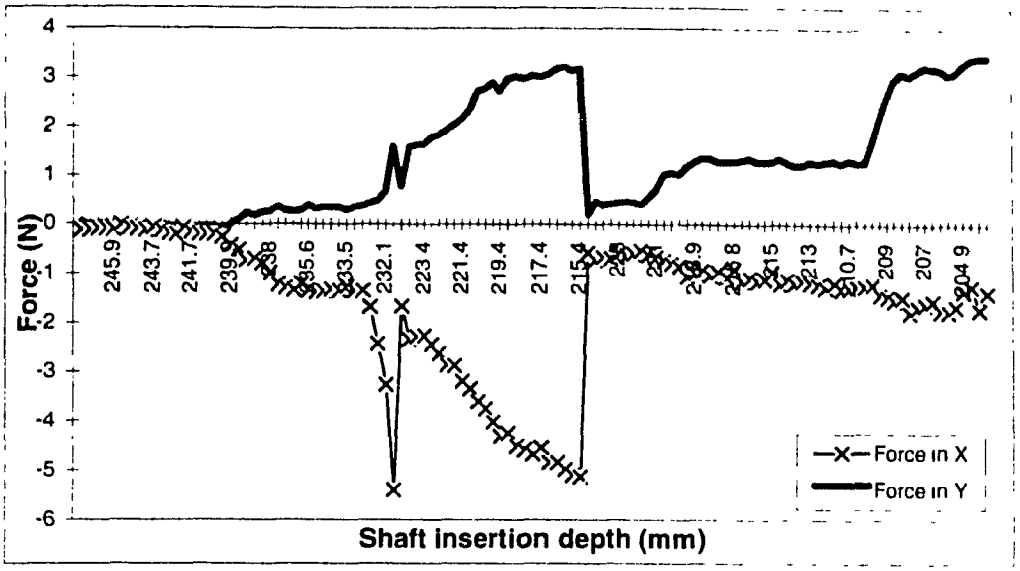


Fig. 7.3 Sample Output for X, Y forces vs. insertion depth

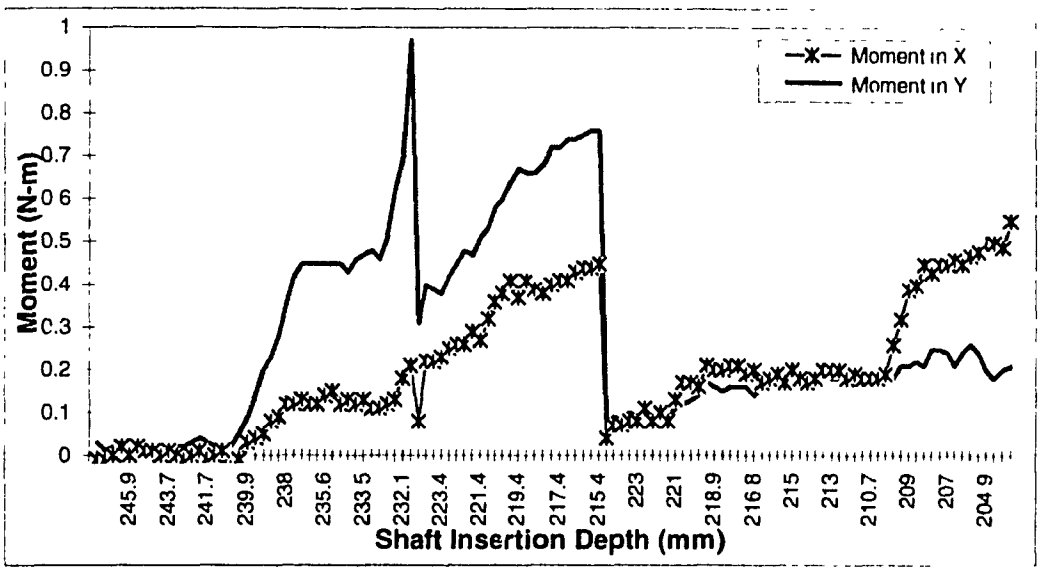


Fig. 7.4 Sample Output for X, Y moments vs. insertion depth

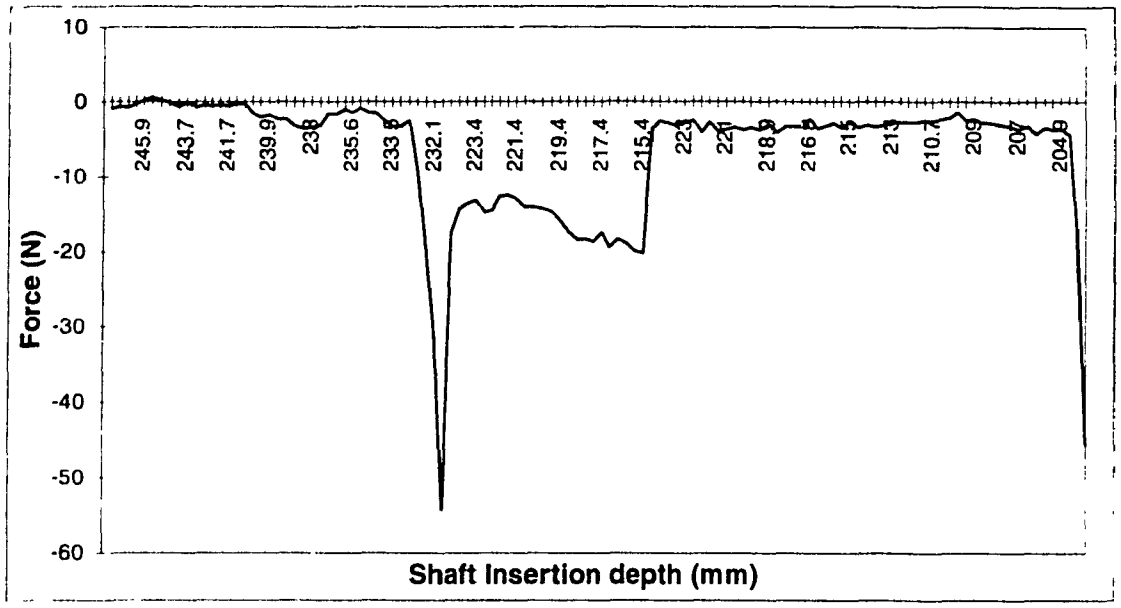


Fig. 7.5 Sample Output for Z force vs. insertion depth

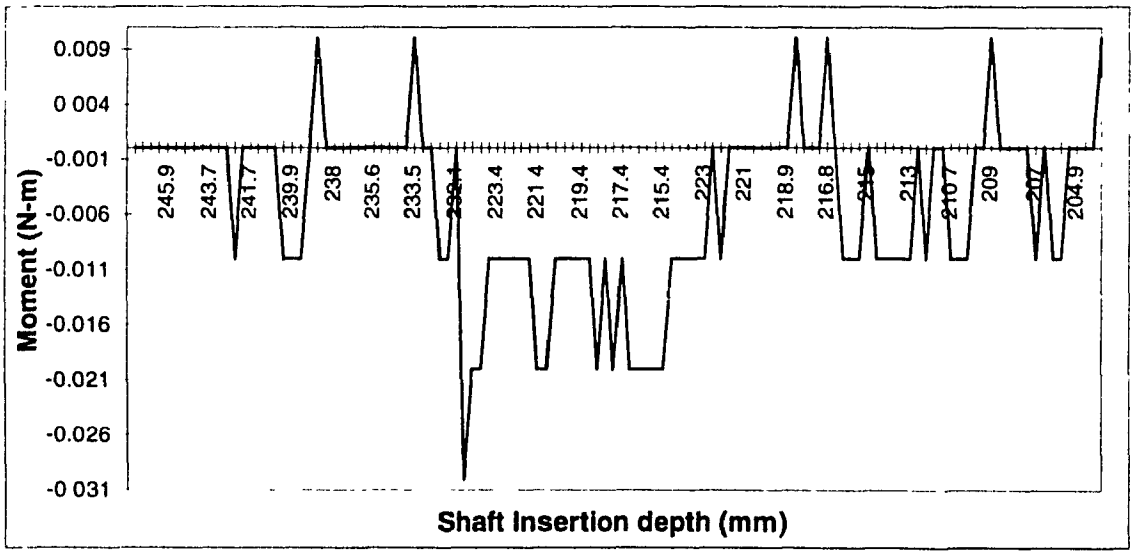


Fig. 7.6 Sample Output for Z moment vs. insertion depth

The smallest incremental move in Z was fixed at 0.4 mm.

Force threshold was set to - 30 Newtons and + 10 N in Z, although the robot had a larger capacity.

Force thresholds in X and Y were set at ± 10 N

Incremental X and Y moves were fixed at 0.5 mm, keeping in view the compliance device`s resolution.

7.3.2 Bench Marks for Algorithm

Bench marks must be set up to see how well the system performs and what amount of learning information is available to the machine learning algorithm. To establish bench marks, moves with zero or near-zero corrections were not included in the grading index, as they were not helping in learning the assembly task.

The concept of *Grading Index (GI)* is similar to that of the Performance Index by Vaaler `91, in that it is used to check the performance of the system and the learning content.

The ***Grading Index GI_S for system performance*** is given as;

GI_S = Ratio of the number of moves carried out for task execution to the total number of moves actually made.

The higher the value of this GI_S, the better is the system performing. The best possibl. value of GI_S is 1.0. The initial 15 assembly runs were not used to evaluate the GI_S as it was felt that the system needed to reach a settle down.

Grading Index for learning GI_L , is used to check the availability of learning content and is given as,

GI_L = Ratio of the number of new visited locations to the total number of visited locations.

The smaller the value of this index, the better is the learning information, indicating that fewer new locations are visited and the robot is learning from the already available locations in its database. Smallest value of GI_L is 0.

The information was collected for 100 successive trials and was plotted. On the norizontal axis was the assembly attempt and on the vertical axis was the Grading Index (System performance) Instead of indicating the total number of locations visited, the total number of assemblies is indicated because it is felt that this parameter gives the overall picture of the learning system. Plots in Figs. 7.7 and 7.8 display the system performance and learning rate. Fig. 7.9 shows the plot of both the grading indices, which indicate that the robot is learning at a satisfactory rate. The results appear to be encouraging as the grading index is promising.

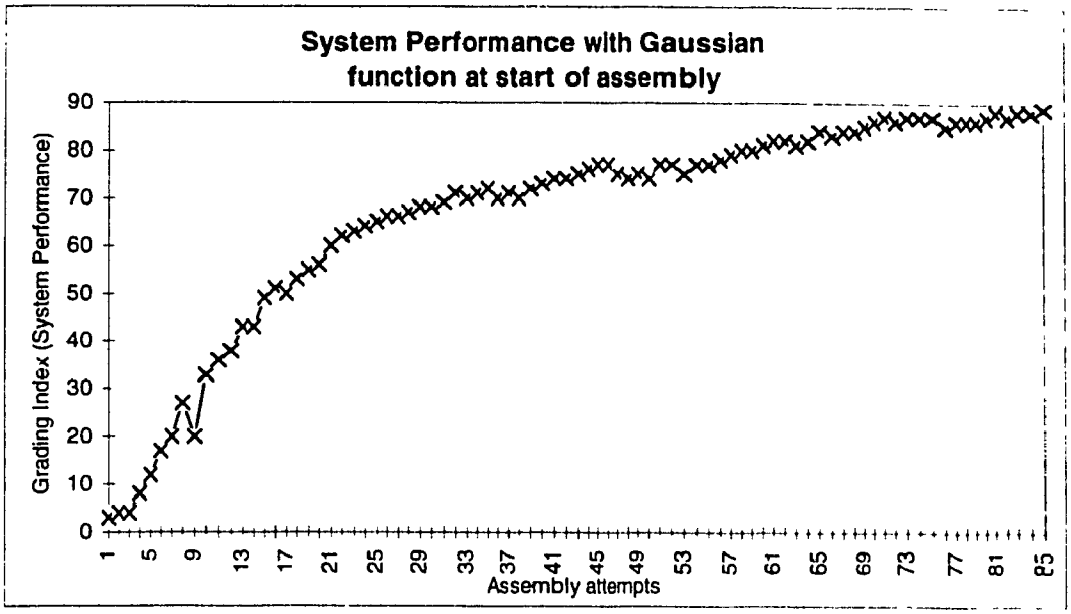


Fig. 7.7 System Performance versus GI_S

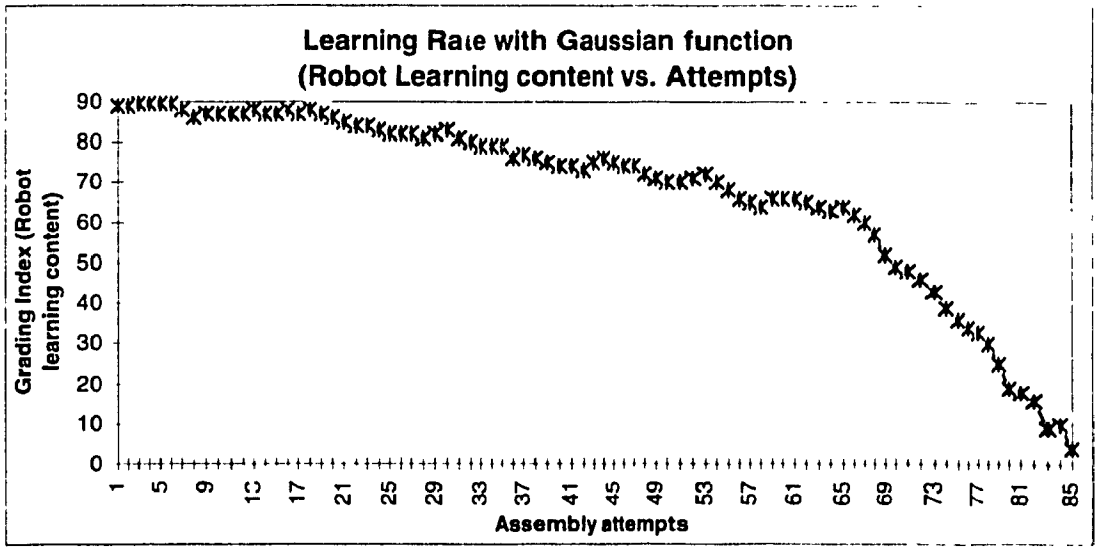


Fig. 7.8 Convergence of algorithm versus GI_L

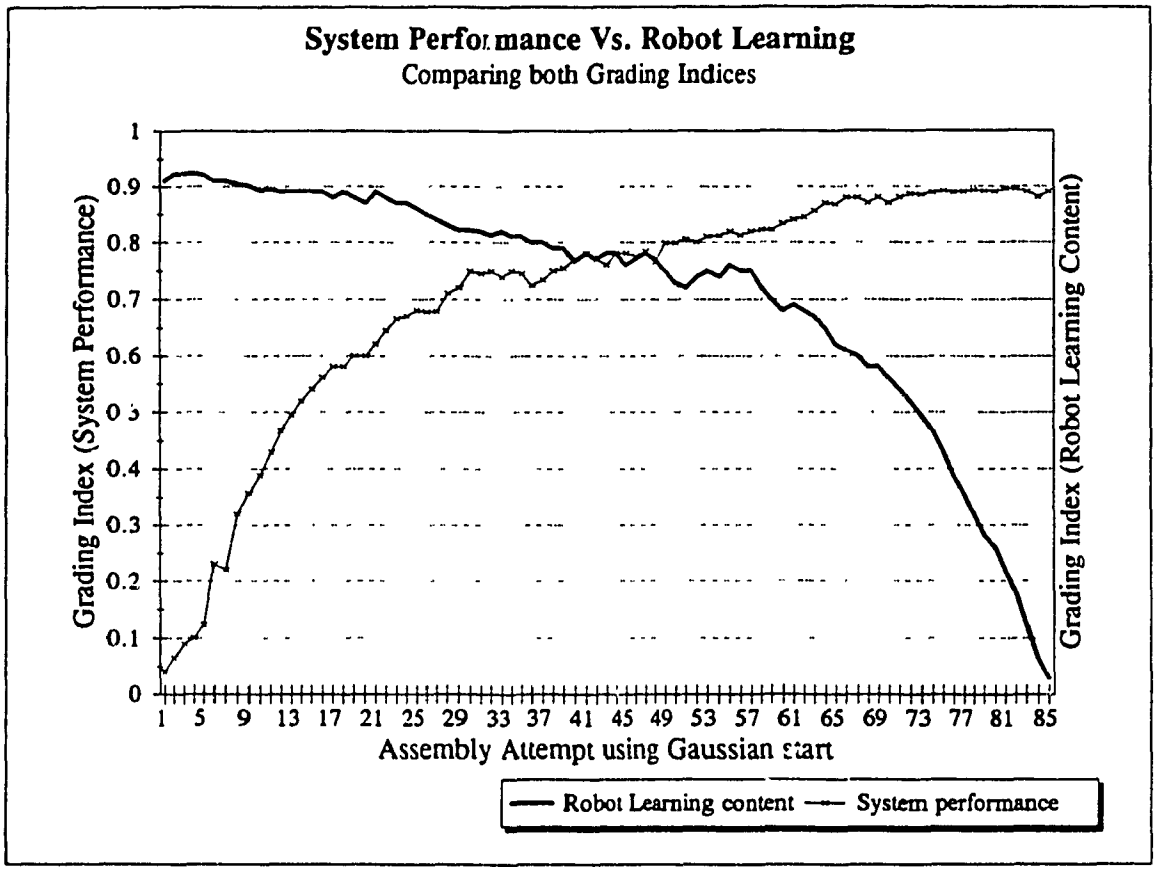


Fig. 7.9 Comparing both Grading Indices

7.3.3 Determining the Rotation angle

This set of experiments was carried out to determine the range of rotation angle required for the shaft to assemble with the hole. As the only information available was the force information, it was necessary to know the angular rotation of the shaft as assembly takes place. Several tests were carried out and data was plotted between the force levels measured and the angle at that instant. However. the spline width is an important factor in this case, as the rotation angle steps vary for different number of splines. On the horizontal axis is the rotation angle during assembly and on the vertical axis is the force level. Figs. 7.10 and 7.12 show results in clockwise direction and Figs. 7.11 and 7.13 show results in the counter clockwise direction for X, Y and Z forces.

7.3.4 Varying the Range

Experiments were done to vary the range and steps of the data. During the assembly process, the data was divided into a set of 6 different regions. The values obtained during assembly were rounded off to the nearest range points. In this way the total number of locations that were visited during assembly were lesser, indicating an improvement in the learning Grading Index GI_L.

By removing this constraint on the data limits, it was observed that the total number of new locations visited increased significantly along with the branching points. To elaborate this aspect, for instance, if a force range along Z is set to a maximum of 10 N

and an angular resolution of 1.0 degrees, and if a data point in Z equal to 10.1 N is encountered

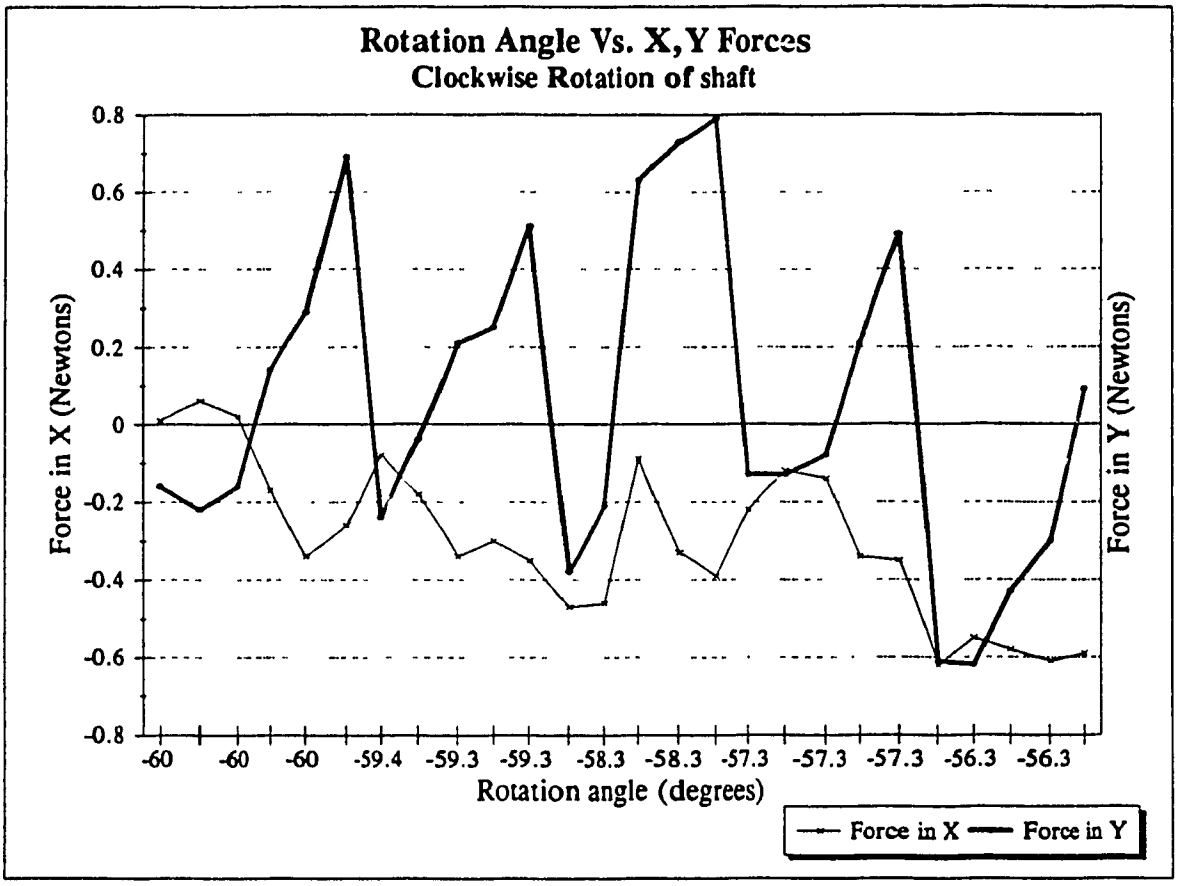


Fig. 7.10 Sample X,Y Force states vs. Rotation angle (clockwise)

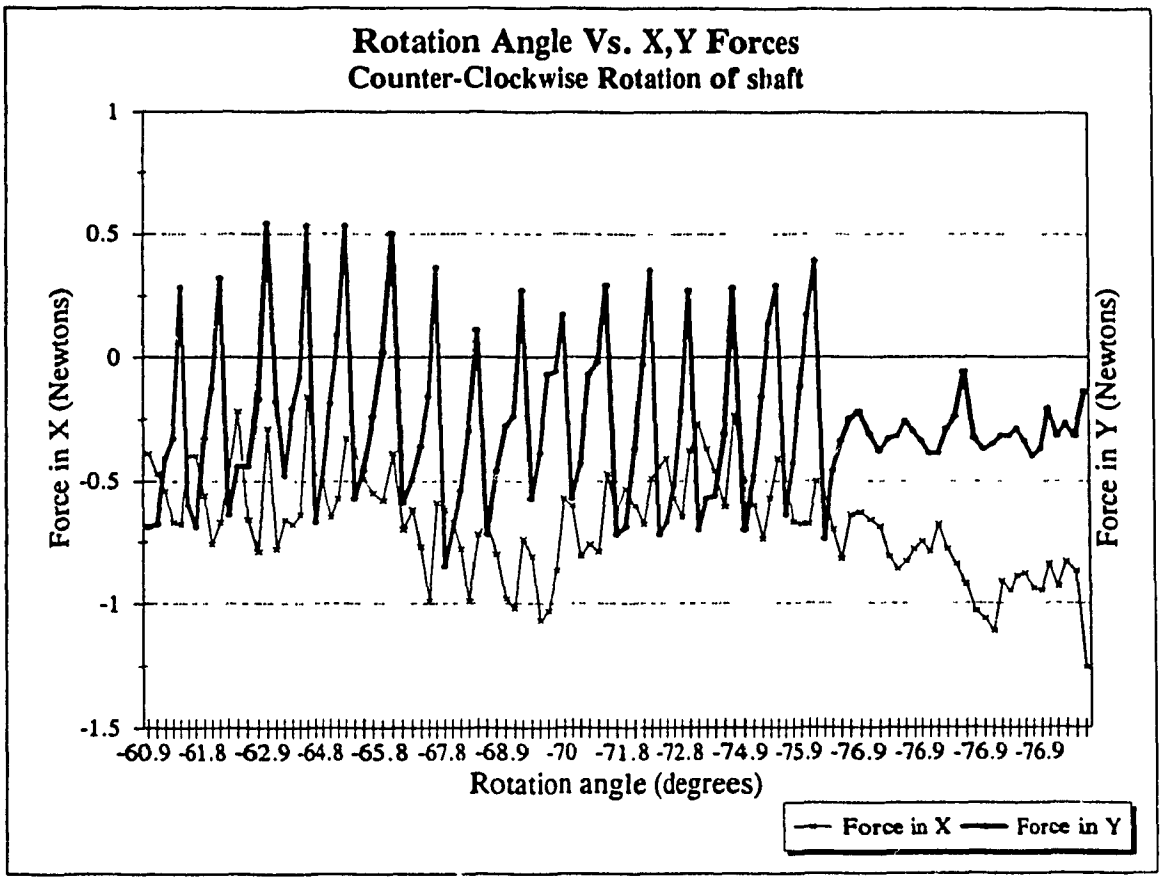


Fig. 7.11 Sample X,Y force states vs. Rotation angle (CCW)

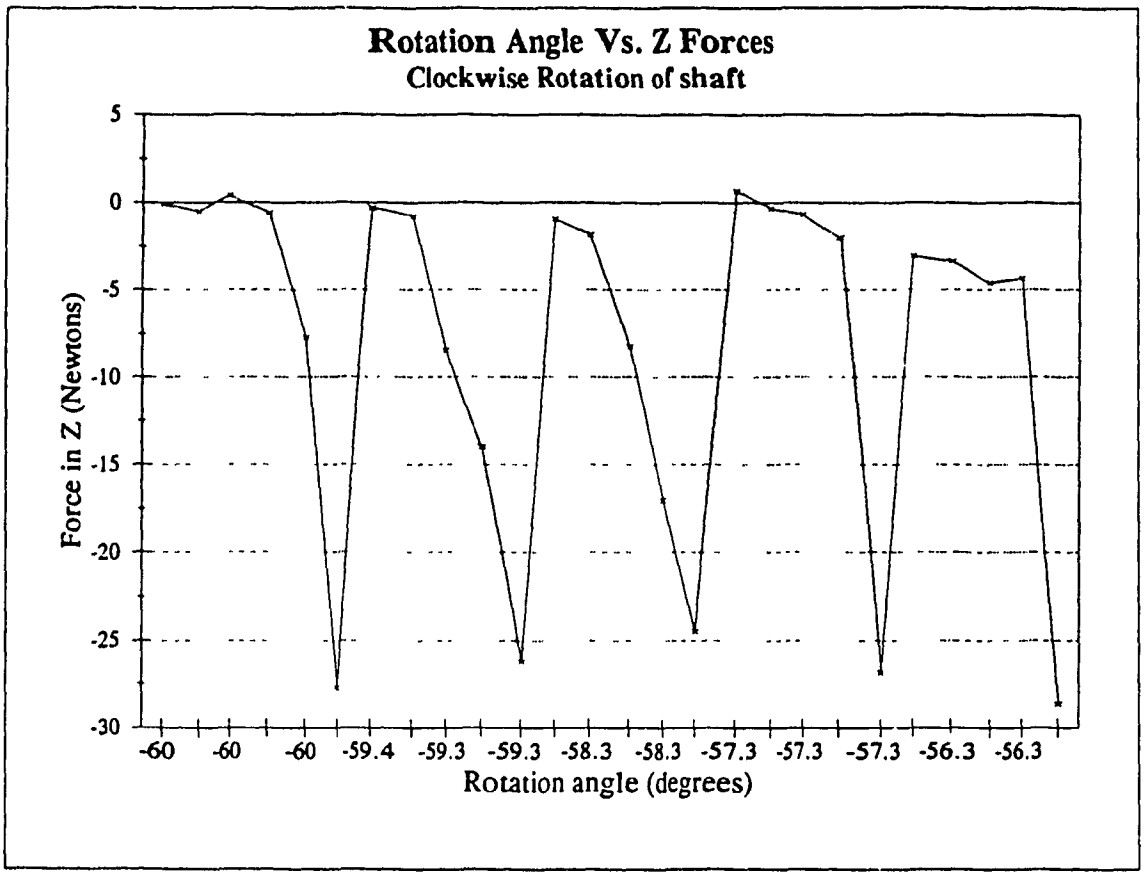


Fig. 7.12 Sample Z force states vs. Rotation angle (clockwise)

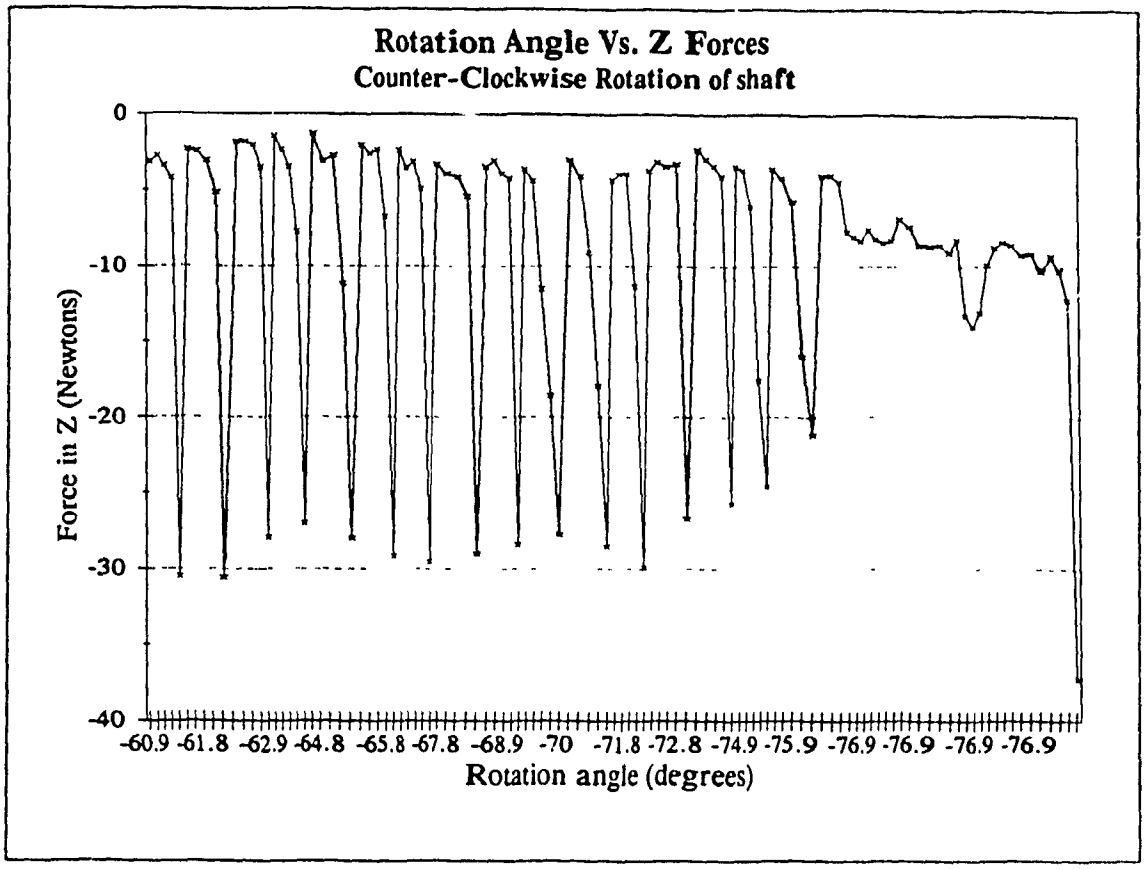


Fig. 7.13 Sample Z force states vs. Rotation angle (CCW)

for an angular rotation of 0.5 degrees, a new data limit is assigned to this move. In this way the total number of new locations visited increased, indicating that the learning process was slower. By removing the constraints on data, the difficulty that was envisaged was that the computation time increased significantly, as the algorithm had to browse through a larger data base and calculate weights for the best move. A sample of data points without constraints is shown in Fig. 7.14.

Assembly attempt	**Required rotations for assembly**	**Total rotations performed**	**Extra Locations visited**
16	6	7	1
17	7	7	0
18	4	7	3
19	6	8	2
20	7	9	2

Fig. 7.14 Sample data points without constraints

In the data above, assembly attempts from 16 onwards are tabulated because for the first 15 attempts the control system of the robot is allowed to reach a steady state.

7.3.5 Setting upper and lower thresholds

In the event that the robot achieved a location that was outside the threshold values that were set for assembly, the data point was rounded off to the nearest threshold value. This set of experiments were different from the previous set, in the way that they were

performed to set the upper and lower thresholds and not particularly the range of the data points. In practice, any force value exceeding the threshold triggered a corrective move from the Move Space M in the algorithm.

The thresholds were set to particular values by checking the hysterisis, non-linearities in the system, the range of the force sensor, its sensitivity and the PCD characteristics.

7.3.6 Changing Probability Distribution

As discussed in earlier chapters, when the robot starts assembly the Information Space I, and the Move Space M, do not contain any data points. For the robot to learn and perform assemblies at this stage, random moves were generated using different probability distributions. Experiments were carried out with two distributions, one using the Gaussian distribution and the other with a Random distribution. As seen from Figs. 7.15 to 7.17, the learning rate and system performance did not vary significantly. This is due to the fact that although the distribution is random, it repeats after several points creating the same moves again.

7.3.7 Tests without PCD

Assembly runs were performed without the PCD. In this case, the set-up consisted of the robot, wrist sensor and the end effector. Other input and output parameters were not

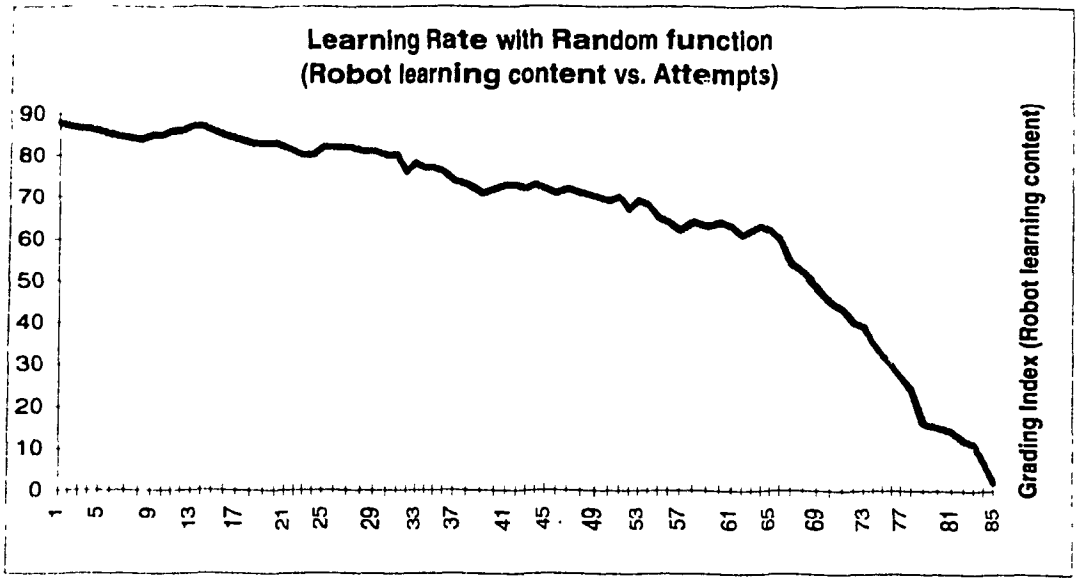


Fig. 7.15 Learning rate with Random distribution

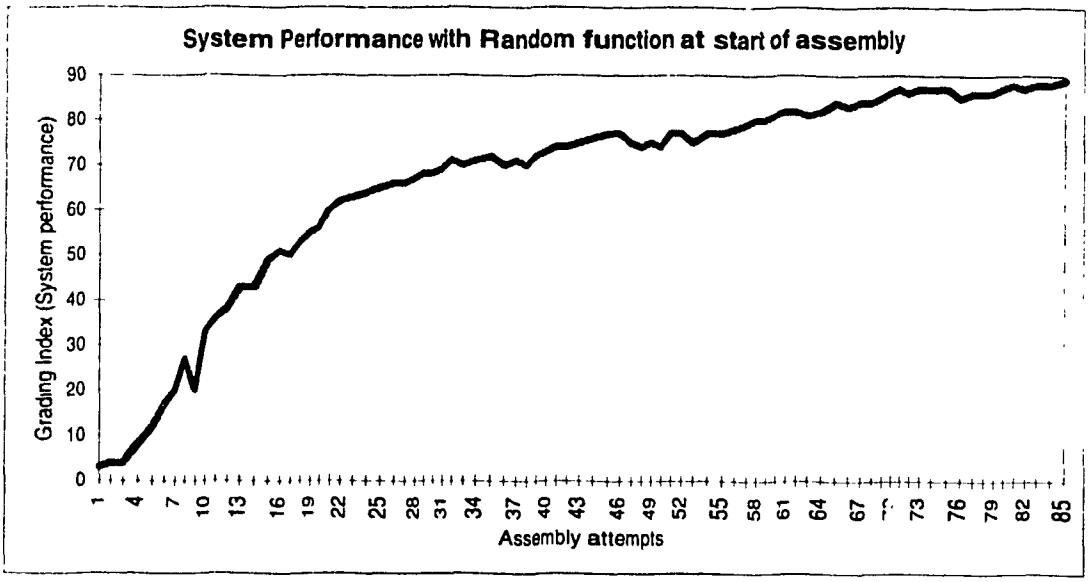


Fig. 7.16 System Performance with Random distribution

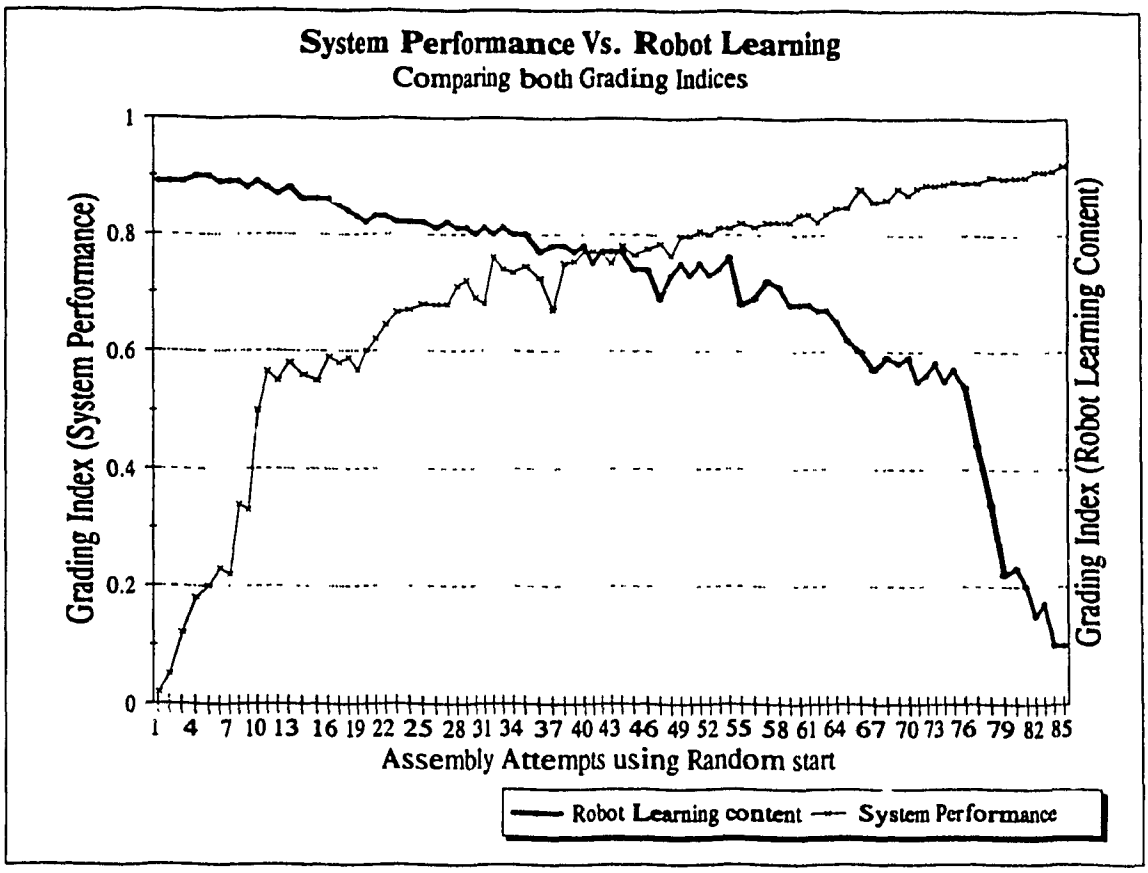


Fig. 7.17 Comparing the Grading Indices

changed. By removing the PCD from the system, the payload of the system increased by about 100 grams, but this did not affect the results, because the experiential payloads were very well within limits. An important aspect without the PCD was that the force range could be reduced for the data points, however, the force values reached very high values. The process showed a better learning behavior since lesser locations were visited. Results of forces and moments are plotted in Figs. 7.18 and 7.19. This might give an indication that by not having a PCD, the algorithm and the system are superior in the learning process. However this is not the case, as there are more restrictions on the moving of the shaft into the hole and locations which are essentially to be visited, cannot be visited as the system gets stiffer. More wedging situations tend to occur without the PCD in the system, shown in Figs. 7.25 to 7.27, from the acquired force information.

7.3.8 Changing discretisation of Force information

These experiments were carried out by changing discretisation of the Information Space I. Smaller values were set for the force sensor information being fed to the algorithm. This set of experiments was done with a view to optimize the computation time and the learning rate of the robot. By increasing the discretisation level, a significant change in the number of locations visited was observed. Also, the total number of moves increased, as such the overall effect on the learning behavior of the robot was not very different, excepting that the computation time of the algorithm increased. This may not be in the interest of faster learning of the robot, nevertheless, these experiments needed to be done

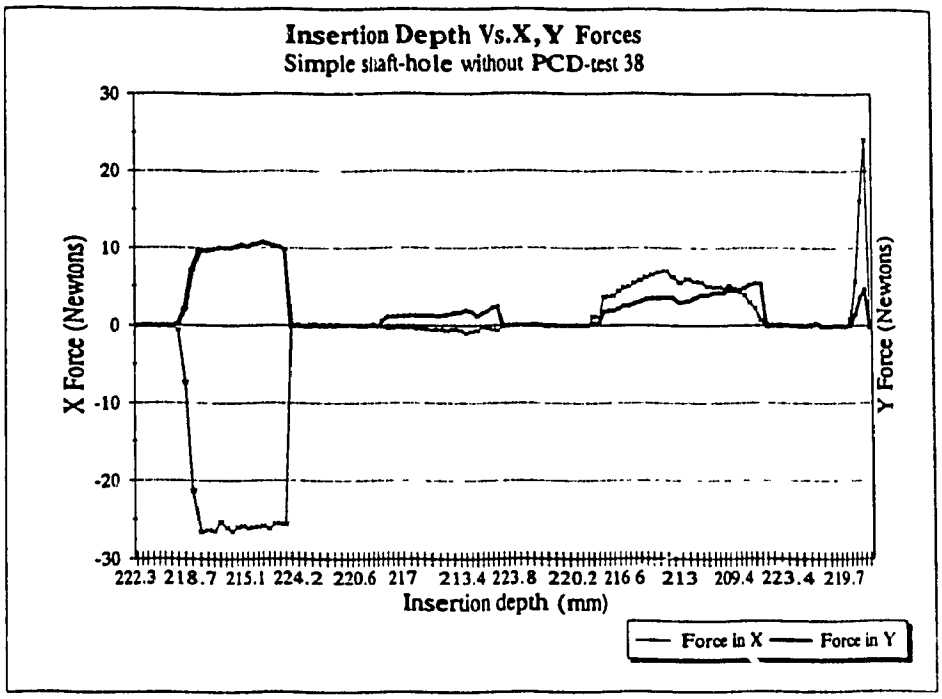
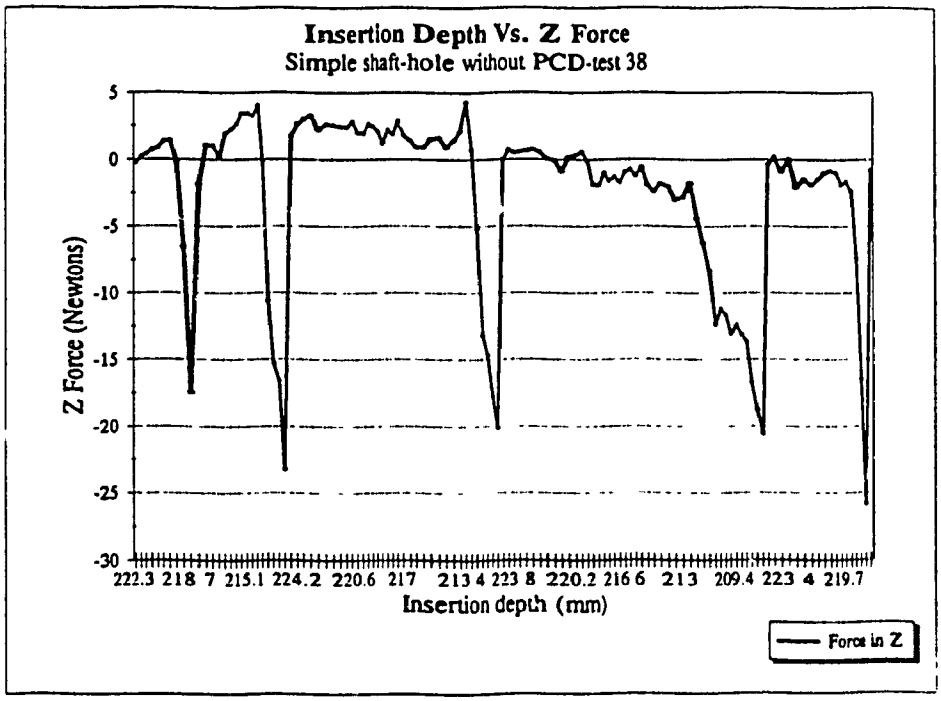


Fig. 7.18 XYZ Forces without using a PCD



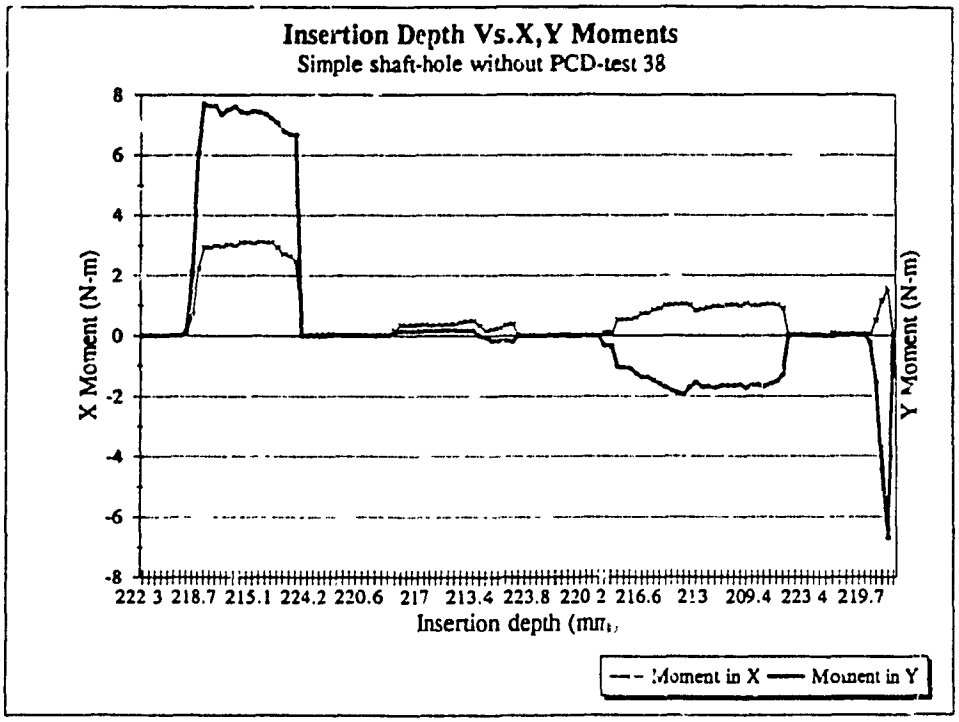
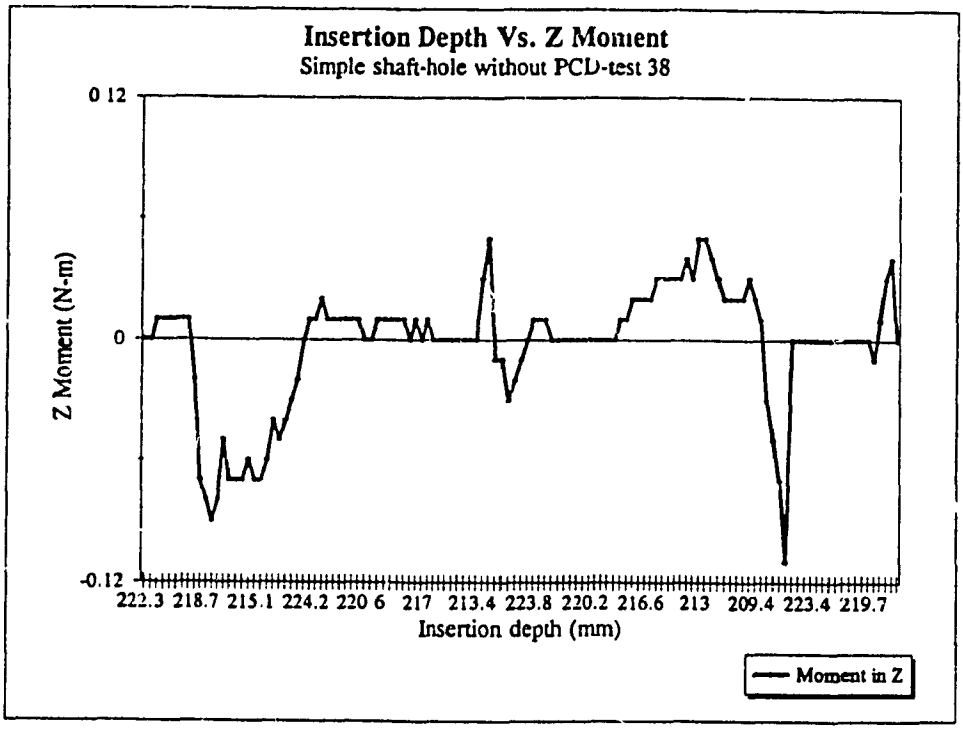


Fig. 7.19 XYZ Moments without using a PCD



to determine the discretisation of the force information.

7.3.9 Changing part clearances

The splined shaft and hole that were used for the assembly experiments had been selected to have a clearance fit. This set of shaft and hole were changed for a different set with lesser part clearances. A sample result of the forces and moments with lesser part clearance is shown in Figs. 7.20 and 7.21. In this set of experiments the total number of locations visited were higher and it was observed that the robot had execution difficulties at times, since the thresholds for force information and the range had changed and the robot could not respond to these stringent requirements. The learning rate improved initially but as the shaft and hole got into contact for rotation, erratic results were seen.

7.3.10 Assembly learning time

This series of experiments were done to check the learning rate of the robot. The experiments were carried out with the standard conditions of threshold and force range data points, with the PCD in the system. It was seen that the assemblies performed at the start by the robot took 107 seconds, but as the robot learned, the assembly of the splined shaft and hole took 14 seconds. This appeared to be a promising learning rate by the robot for the algorithm developed. Also, this is a good indication of the reduced computation time for the algorithm as assemblies are repeated by the robot over a time period. Figs. 7.22 to 7.25 show a comparison of the learning rates of forces and moments for simple and splined shaft and holes, in X, Y and Z axes.

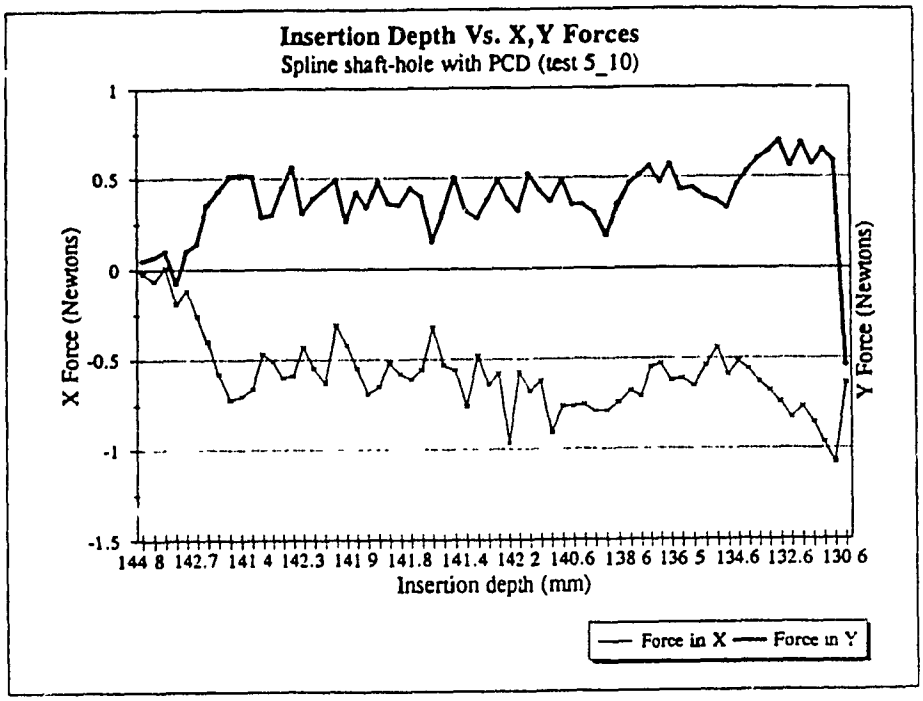
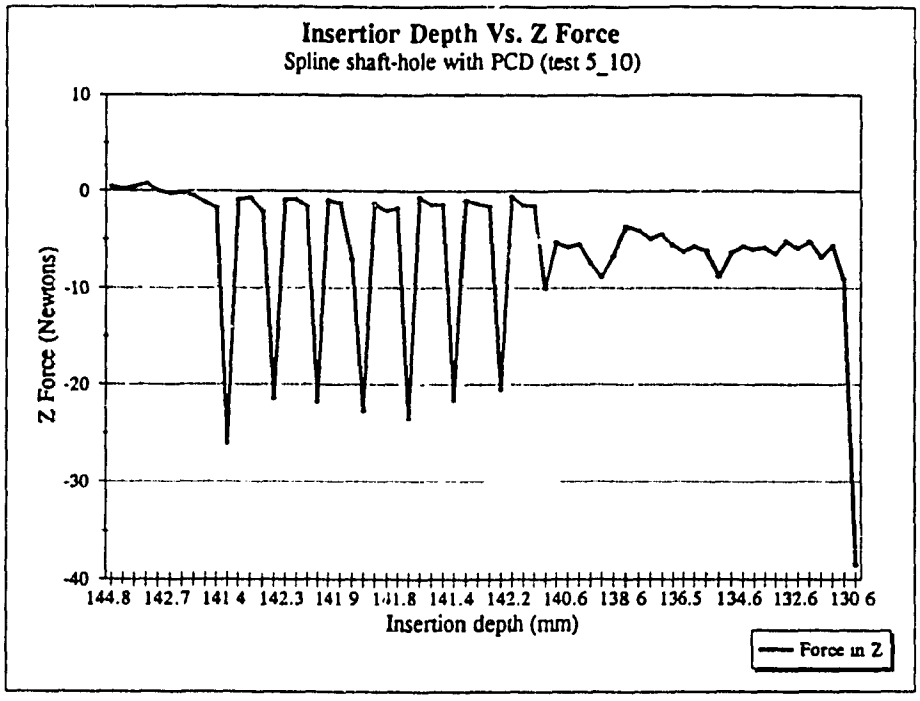


Fig. 7.20 Using PCD, XYZ Forces for parts with lesser clearance



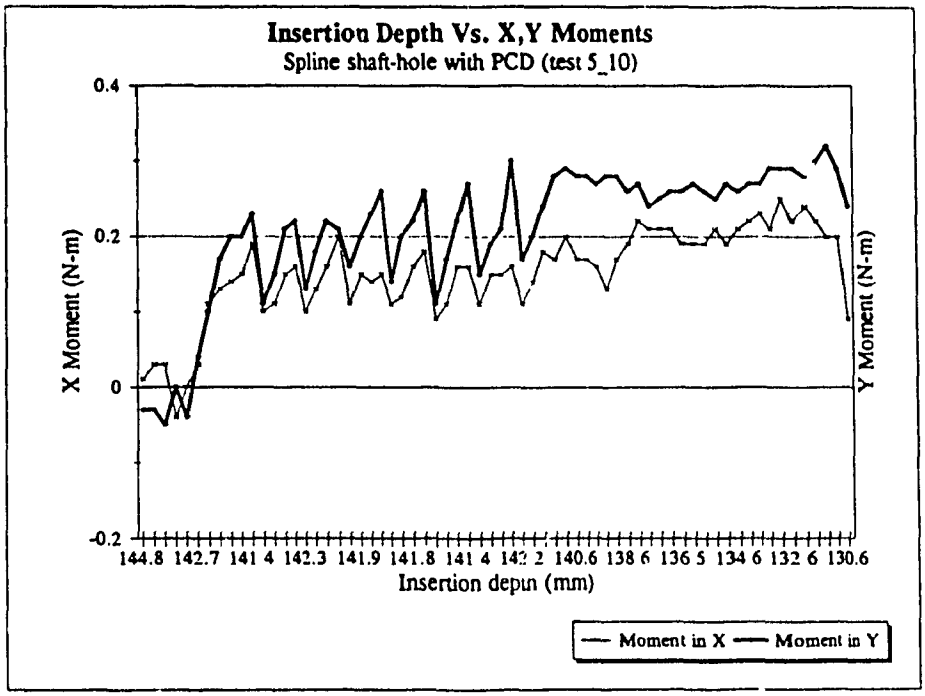
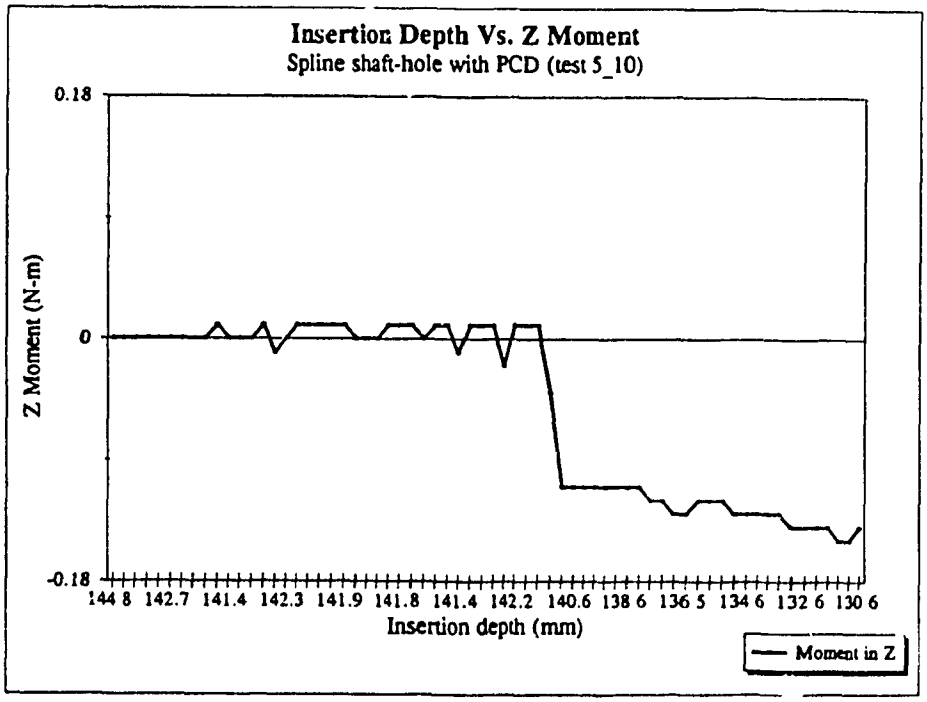


Fig. 7.21 Using PCD, XYZ Moments with lesser part clearance



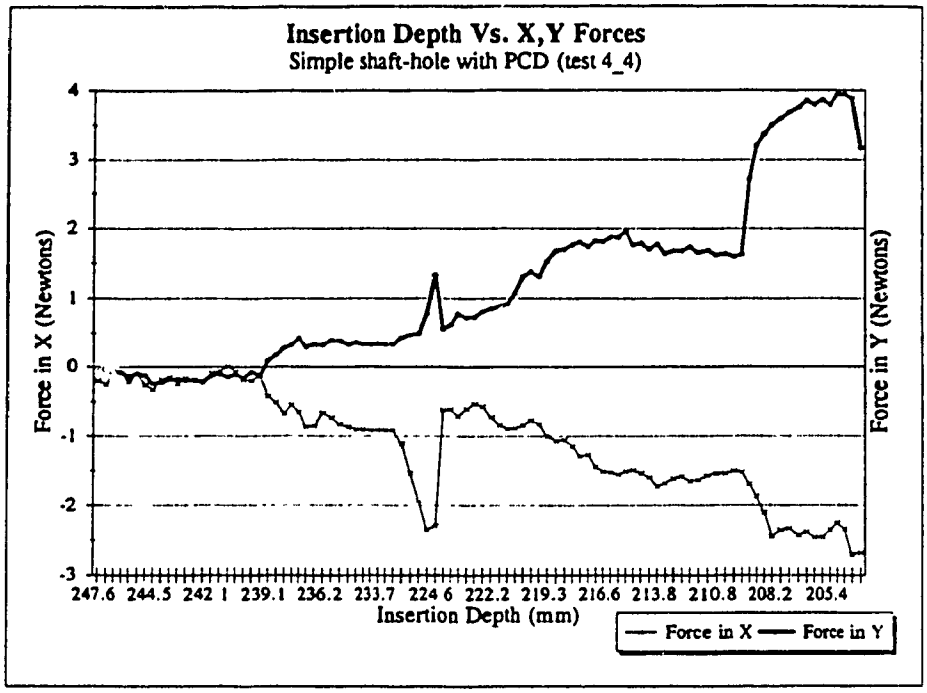
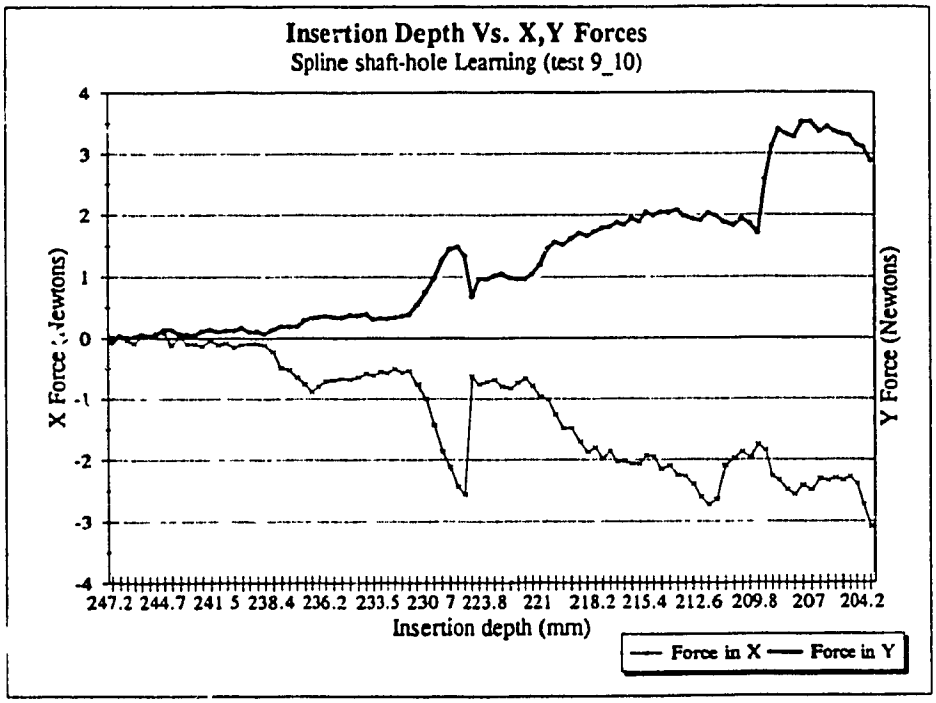


Fig. 7.22 Comparing the learning of XY Force states for spline and simple shafts



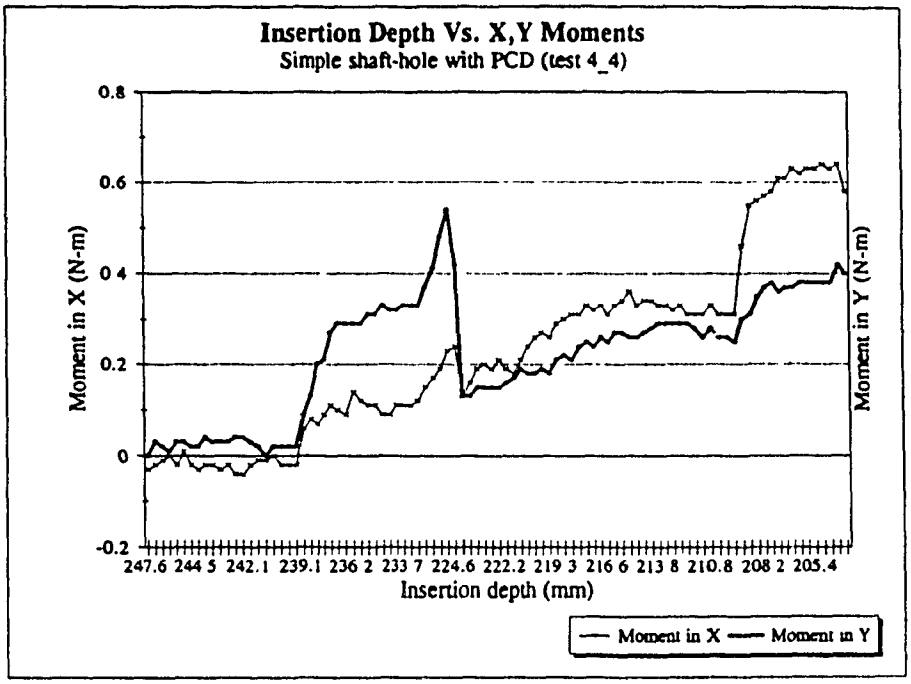
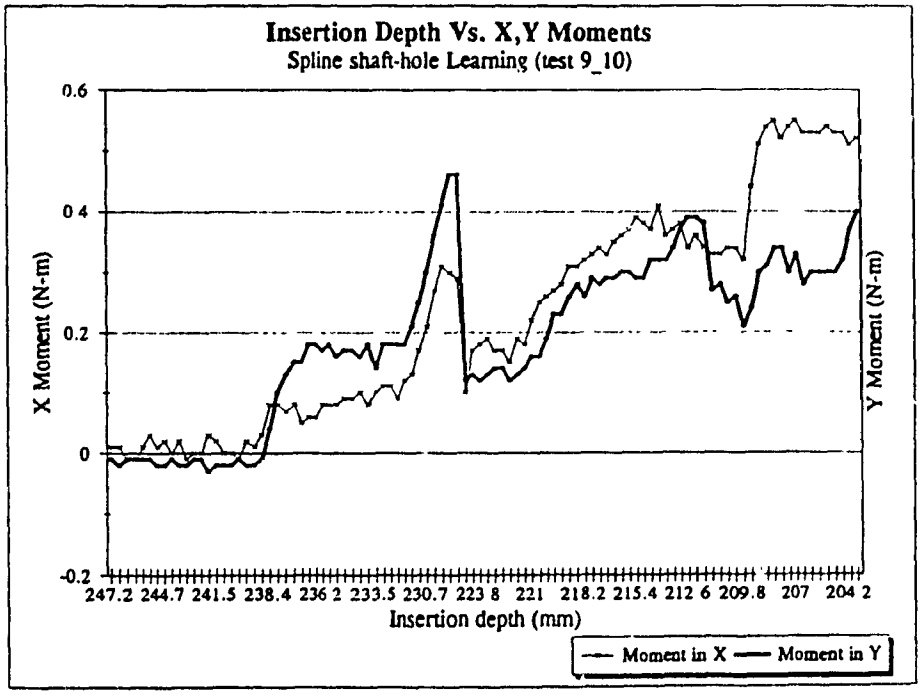


Fig. 7.23 Comparing the learning of XY Moment states for spline and simple shafts



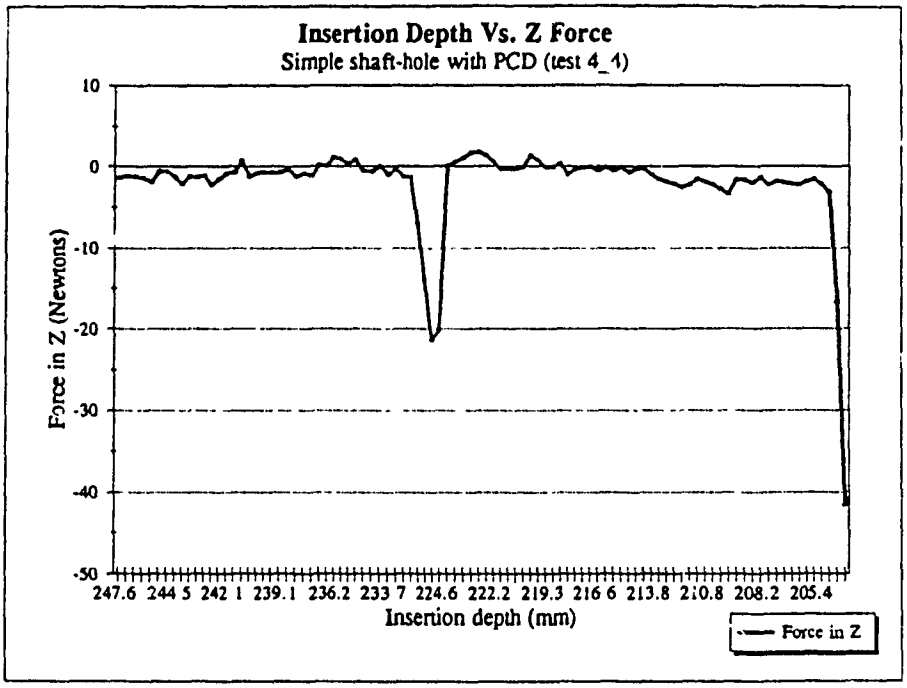
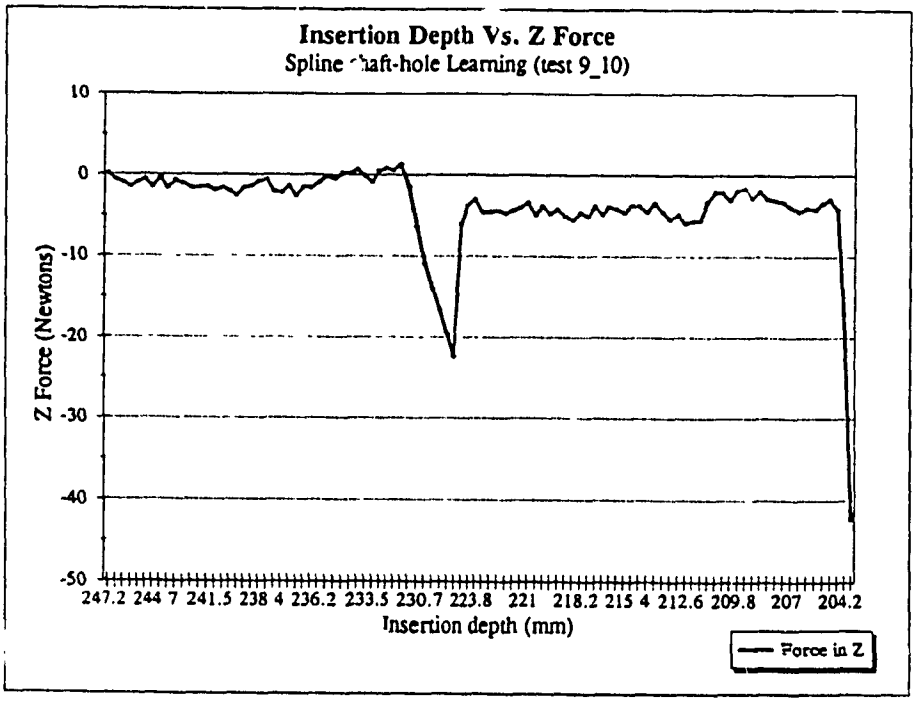


Fig. 7.24 Comparing the learning of Z force states for spline and simple shafts



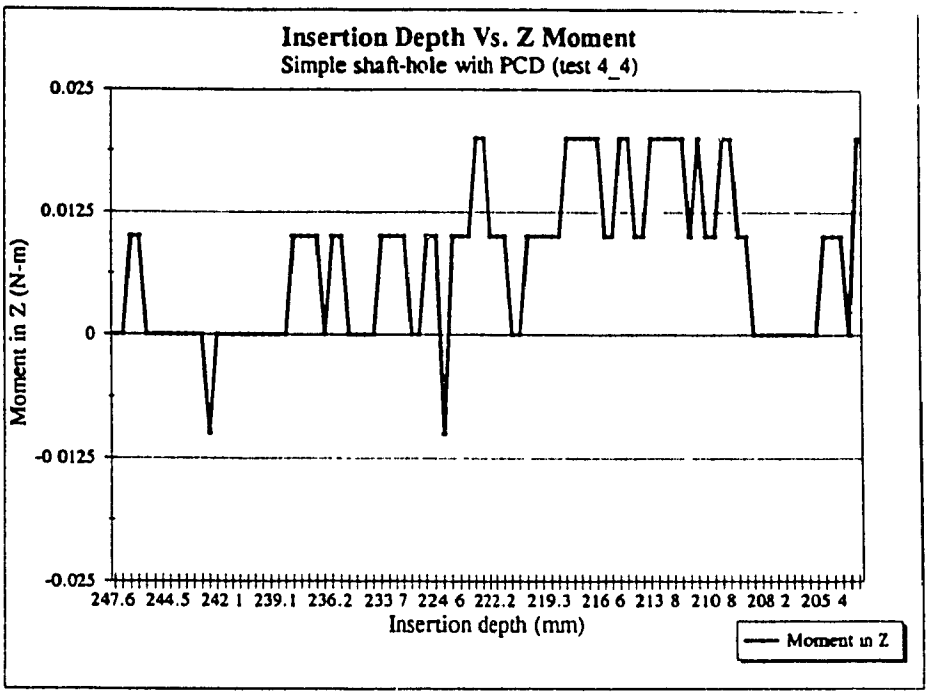
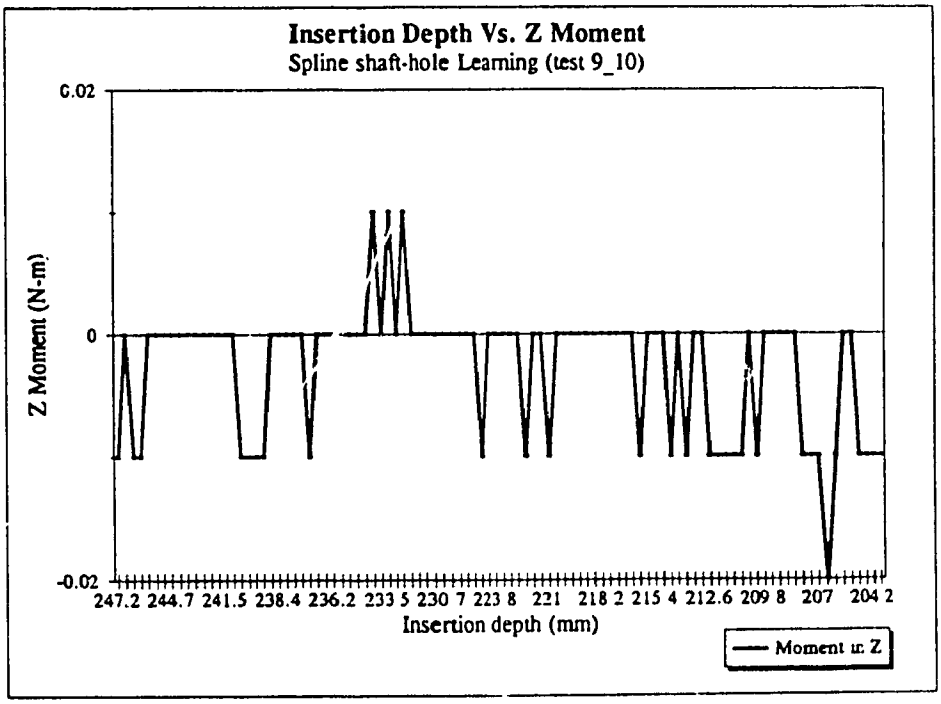


Fig. 7.25 Comparing the learning of Z moment states for spline and simple shafts



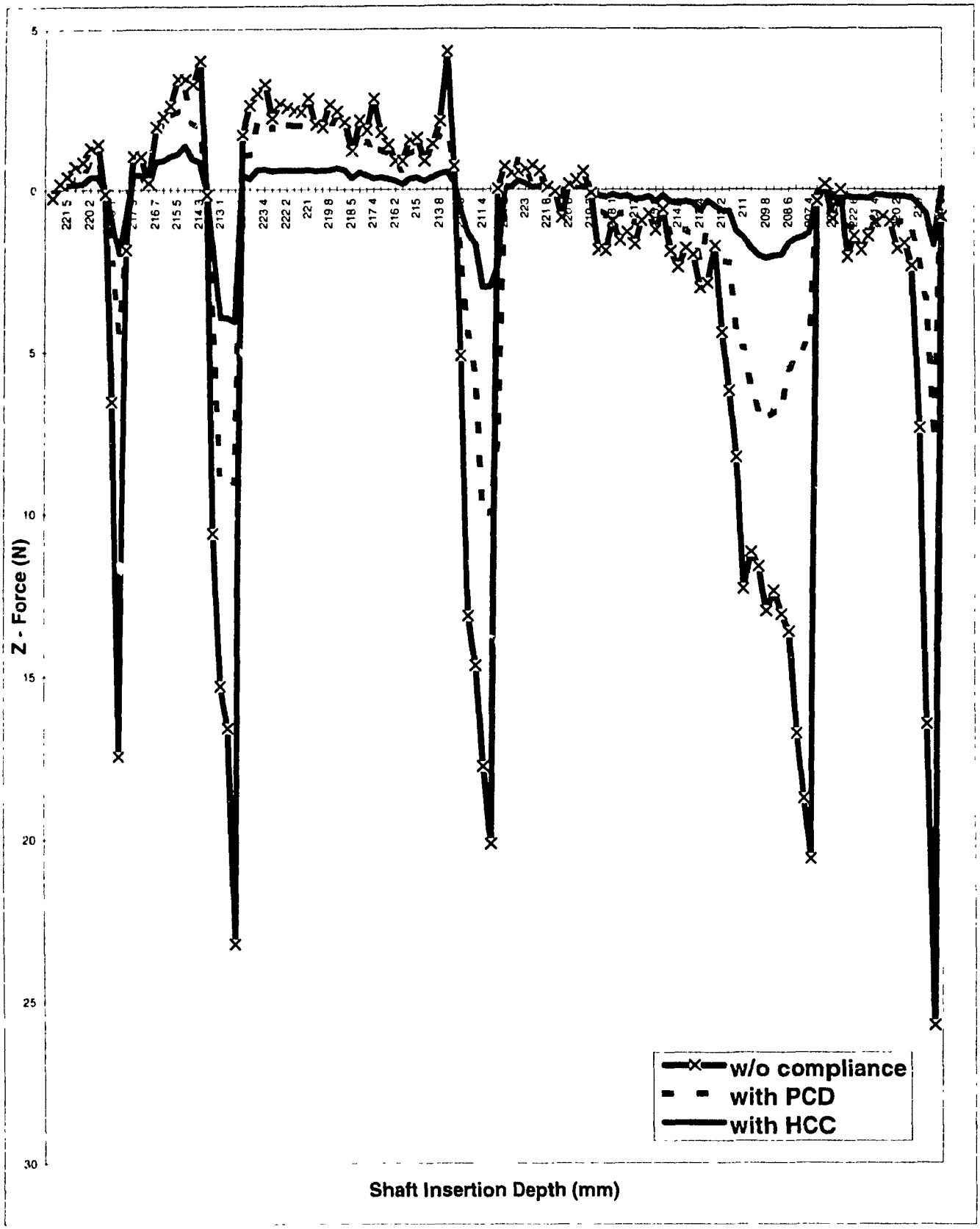


Fig. 7.26 Measured Z-force levels using different levels of compliance

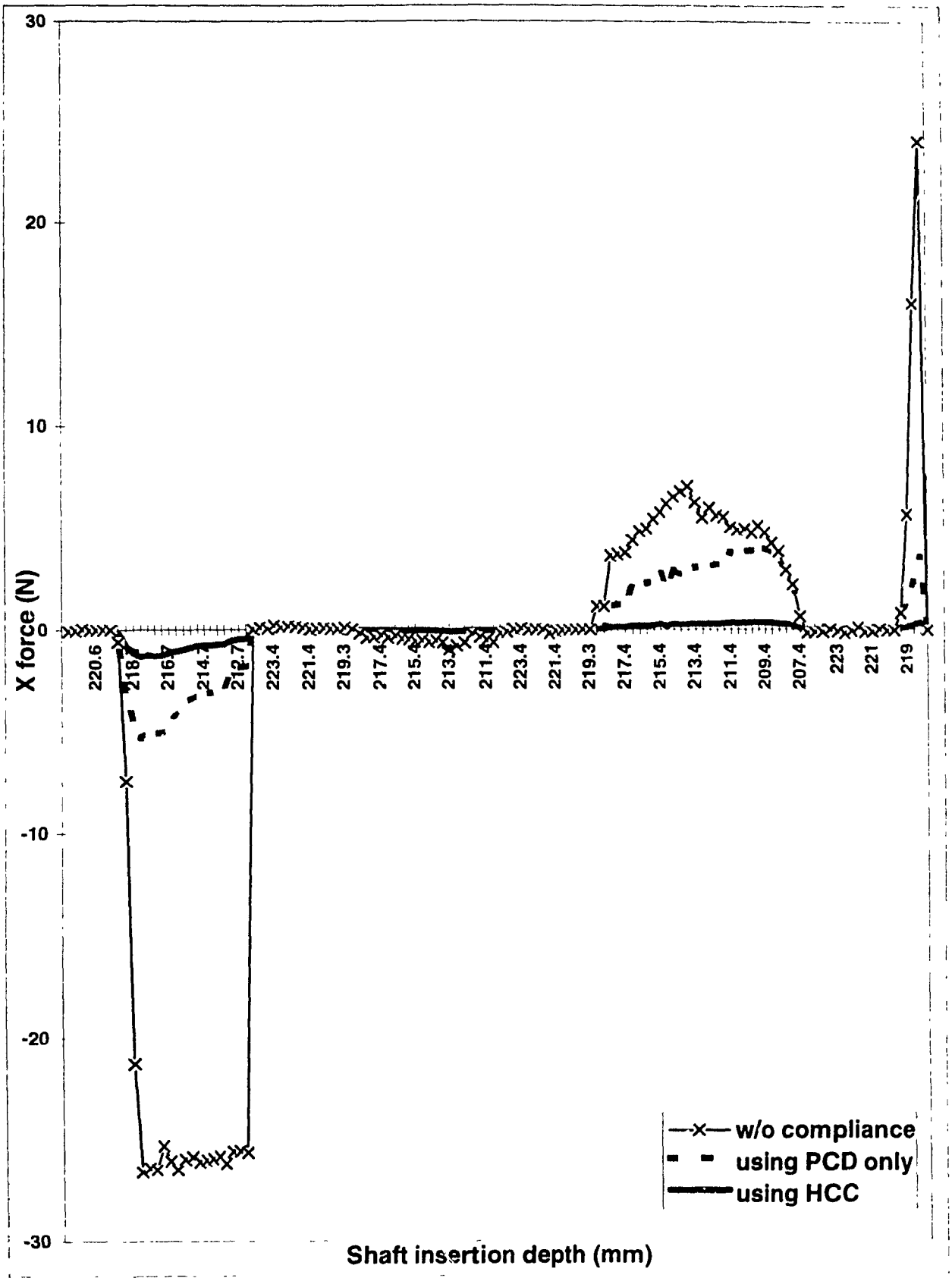


Fig. 7.27 Measured X-force levels with different compliances

Chapter 8

Conclusions

At the outset, the goals of this work were set with a view to making contributions and niches in the existing robotic assembly horizon, both in the industry and the research laboratory domains. As this work is concluding, it is heartening to learn that the efforts put in were successful in attaining these objectives to a certain extent.

8.1 Research Contributions

This research work has made the following research contributions to the robotic assembly domain:

1. The Passive compliance device developed and employed in the experimentation, proved to be successful. The design and manufacture of the device is very simple and its modular concept make it hig eful for easy and fast adaptation in the industry in any robot work cell.

2. This research used a novel Hybrid Compliance Control (HCC) concept. The PCD in conjunction with the force feedback generated position vectors, creates an excellent information and move space for the robot, improving the robot learning process significantly.

3. The HCC concept has introduced an interesting aspect of a tunable compliance in any robotic task without major modifications in the process. Over repeated task executions it is possible to arrive at an optimal level of compliance for a particular task. Introducing redundancy in the HCC has been a positive step in the direction of implementing tunable compliance in any task.

4. An effective mapping technique between force information and the position was developed. The idea of using weights, towards force information at specific locations of the spline and the hole and generating a corrective move from the Move Space in the Fine motion and Insertion stages has been successful.

5. The research was directed towards splined shaft and hole assembly, and as seen from the experiments the algorithm has been shown to perform very well for these monotonous repetitive tasks. The algorithm's capability to select the best moves at the start from a defined distribution was demonstrated.

6. This research has introduced the machine learning concept with a combined flavor of both, learning by examples and by induction. This concept was not seen in the literature, especially in the robotic assembly domain, thus making it as one of the prime contributions in the horizons of robotic assembly.

7. The combination of learning strategies in this research has improved the convergence rate of learning by the robot and is an encouraging criterion, especially when such a strategy is being tried for the first time in robotic assembly.

8. Different force mapping techniques have been tried for simple shaft-hole assemblies, but the technique developed in this research is a concept that has not been applied to splined shaft and hole assembly and this makes it a very strong contribution to the robotic assembly domain.

9. The algorithm has been so developed that the parts being assembled do not have influence on its structure. Only, the robot learning time, and the learning

content could vary depending on the geometry of the parts being assembled, but the behavior of the algorithm is not affected. This increases the versatility of application of this algorithm.

10. The machine learning algorithm was developed ingeniously, by not incorporating the vision system and its associated electronics and software, thus reducing costs significantly and also doing away with the halo of high technology normally encompassing the robot. This aspect will be appealing to the industry and the managers and technicians working on production lines.

8.2 Observations

There were some observations made during experimentation which deserve to be mentioned in this chapter.

1. The type of robot being used for an assembly task is important. If the robot resolution is lower than the clearance of parts being assembled, then incremental moves may not be possible to perform the assembly process.

2. In this research, the discretising of data points was carried out by trial and error, like some other researchers, instead some automatic technique could be used to

carry out this process, which would not only be faster but also may result in an optimal result.

3. The computation time considerably reduced as the tasks were repeated over a period of time using this algorithm, this was evident from the assembly time that reduced from 107 to 14 seconds.

4. While performing tests without the PCD, it was observed that the system was stiffer and this change was very significant. This could pose a serious problem in maneuvering the shaft and hole during the assembly task. Also, the behavior could change if the center of compliance of the passive device was different. Though the computation time reduces without the PCD, in effect the minimal number of locations required to be visited by the robot is reduced and there are chances that the robot has missed some good moves.

5. It is important that the range and threshold be set after the sensor and passive device are fitted on the robot, as these influence the set limits to a large extent.

6. When the robot starts, the probability distribution selected for generation of moves does not seem to affect the learning process as a whole. There are slight variations initially but overall it does seem to behave the same.

7. To determine the rotation angle and resolution, a major influencing factor is the number of splines on the shaft / hole. As expected, the rotation angle and the learning rate are not significantly influenced by the direction of shaft rotation during the fine motion stage.

8. Learning rates in some tests, especially when clockwise and counter-clockwise rotations were performed seem to be slightly different at certain points, this can only be attributed to noise in the electronics, EMI and human errors. The algorithm has been developed to ensure that robot learning is not affected by shaft rotation direction.

9. During the assembly process, the geometry of parts or their state must not change, i.e., if the shaft buckles then the learning process would adversely be affected and may fail eventually, due to erroneous force information.

10. The assembly time of 14 seconds at convergence of the algorithm may appear very high, since in practice it may not take that much time. To improve this time, a more stringent discretising can be done, the robot dynamics and interfacing of sensor with PCD can be shortened, to avoid overhang at the end of the robot arm.

This chapter concludes with remarks that the author has established and evolved an algorithm using machine learning for splined shafts and holes, using the hardware

developed and utilizing the supporting software for the force sensor available in the laboratory. From the experiments it is evident that a niche has been created in the domain of robotic assembly especially using machine learning for teaching the robot.

Chapter 9

Future Work

As a result of the experiments there are some areas which have been identified for further research work. Each sub-system, hardware and algorithm have been considered separately and the respective future issues discussed.

PASSIVE COMPLIANCE DEVICE

A new version of the Passive Compliance Device that will make it more versatile in different applications could be built with the following modifications.

Performance Enhancement of PCD

On the Float, there can be two sections diametrically opposite to ensure that the rotational limit of the device is within a specified range, depending on the application.

To locate the springs on the top and bottom plates easily, small levers with screws can be fitted. When these screws are loosened, the lever can be rotated with the screw as the pivot allowing the spring to be removed. On tightening the screw, the spring is fixed and located on the plates firmly, thus reducing the PCD assembly time.

Another way to hasten the device's assembly is to drill holes on the outer surface of the plates matching the spring locating holes, and fix screws in these holes with a length that limits only 2 coils of the spring.

Another version of the device with 4 springs could be developed, this device will have greater lateral stiffness and torsional stiffness, by virtue of incorporating 4 springs of the same stiffness. The cross-sectional drawings of this version for top plate, float and the bottom plate are shown in Figs. 9.1 to 9.3, respectively.

Weight reduction of PCD

The thickness of the top and bottom flanges in the housing can be reduced by 2 mm, to help reduce weight, and more fasteners with smaller thread length can be used. Also, the thickness of the float can be reduced by 3 mm.

The PCD can be manufactured using a reinforced plastic / composite which will significantly reduce the weight. A word of caution on a plastic PCD is that the locating holes of the springs have to be reinforced, to avoid breaking at these spots during the torsional actions of the PCD.

WRIST SENSOR

In order to have increased compliance, a suitable alloy / material can be used for the straining rods. This increased compliance may help avoid the use of the separate PCD.

ALGORITHM

The work can be extended to keyed shafts and holes. The only additional input to the algorithm will be the vicinity range of the key and the key way. This can be calculated after running a few trials or a vision system can be incorporated that will give the required gross coordinates to the algorithm.

When the algorithm starts assembly, the generation of moves can be carried out using fuzzy logic instead of using the different probability distributions. This may result in a better move initiation strategy.

If for an assembly task the CAP is not known or difficult to achieve, a path to best fit this CAP can be generated automatically in the algorithm. This aspect will add a new

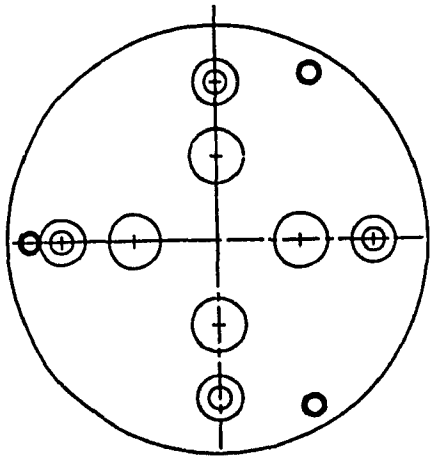


Fig. 9.1 Top plate for PCD (future version)

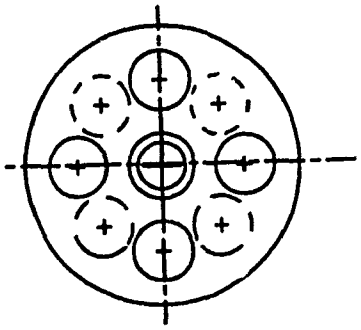


Fig. 9.2 Float for PCD (future version)

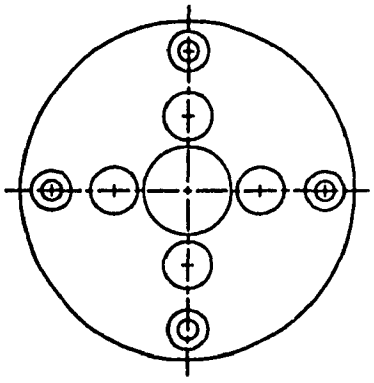


Fig. 9.3 Bottom plate for PCD (future version)

dimension to the research work, making the algorithm universally applicable, in the sense that the geometry of parts being assembled will no longer matter.

The algorithm has the scope to be extended to 3 D assembly of a wider range of parts. It is also suggested that the machine learning can be incorporated using only the learning by examples or learning by induction technique and the results can be analyzed. However, the combination has facilitated faster learning of the robot.

A-priori force information in the Information space can be very useful in the learning process. Although, this has been used to some extent in this research, it seems to have lost its effect in view of the weighting functions. The aspect of trying to generate moves exclusively on a-priori information can be pursued.

Though the algorithm is designed for splined shaft-hole assembly it can be used for simpler tasks like shaft-hole assembly, flange and shaft assemblies, bearings and shafts assembly and can be extended to other industries, not limited to the automotive industry.

REFERENCES

1. D S Ahn, H S Cho, K Ide, F Miyazaki, S Arimoto, "Learning Task Strategies in Robotic assembly systems", *Robotica*, 10(5): 409-418, 1992.

2. Alain Desrochers, N Krouglicoff , Daniel Croteau, "Developpment d'un Capteur d'Efforts integre pour le Controle Adaptif de Trajectories Robotisees", *1993 DND Workshop on Advanced Technologies in KBS & Robotics*, 14 - 17 November 1993, Ottawa.

3. Andrew Kusiak, "Intelligent Manufacturing Systems", 1992, *Prentice Hall Inc.*, New Jersey.

4. Arun Jaura, N Krouglicof, M O M Osman, "Robotic Assembly Using Machine Learning in a Manufacturing Cell", IRIS-PRECARN V Annual conference on AI and KBS, Vancouver, June 13-15, 1995.

5. Arun Jaura, N Krouglicof, T S Sankar, "Dynamics of a Robot arm for an Optimized Mounting Configuration", IRIS-PRECARN Annual conference on AI and KBS, Ottawa, June 1993.

6. Arun Jaura, N Krouglicof, M O M Osman, "Machine Learning based Logic Branching Weighted Algorithm for Robotic Assembly" -- *IEEE Transactions on Systems, Man and Cybernetics [Submitted].*

7. Arun Jaura, N Krouglicof, M O M Osman, "Modular Passive Compliance Device for cost reduction in Robotic Assembly work cells" -- *International Journal on Robotics and Computer-Integrated Manufacturing [Submitted].*

8. Arun Jaura, N Krouglicof, M O M Osman, "Progressive Machine Learning Algorithm for Robotic Assembly Tasks" -- *IEEE Transactions on Robotics and Automation [Submitted].*

9. Arun Jaura, N Krouglicof, M O M Osman, "Automatic Robotic Assembly Path generation using Machine Learning and Hybrid Compliance Control" -- *International Journal of Robotics Research [Submitted].*

10. Arun Jaura, N Krouglicof, M O M Osman, "Stewart Platform based Six degrees of Freedom Robot Wrist Sensor for Smart Assembly" -- *Journal of Manufacturing Systems [Submitted].*

11. Arun Jaura, N Krouglicof, M O M Osman, "Hybrid Compliance Control for Intelligent Assembly in a Robot Work Cell" -- *International Journal of Production Research [Submitted].*

12. Arun Jaura, N Krouglicof, M O M Osman, "Smart Handling and Assembly of Nuclear Systems using Machine Learning and Compliance Control by Robots" -- *for 7th American Nuclear Society Conf. on Robotics and Remote Systems [Accepted].*

13. Arun Jaura, N Krouglicof, M O M Osman, "Hybrid Compliance Technique for Improved Dexterity in Automated Assembly Operations" -- *Measurement Science and Technology [Submitted].*

14. Assurance Technologies Inc., North Carolina, `Compensator Description and Selection Guide'.

15. Assurance Technologies Inc., North Carolina, *Product Catalogs* on RCC Compensator.

16. E Bayo, J R Stubb, "Six Axis ForceSensor Evaluation and a New Type of Optimal Frame / Truss design for Robotics Applications", *Journal of Robotics Systems*, 6(2), pp. 191 - 208

17. A K Bejczy, "Smart Sensors for smart hands", *Progress in Astronautics and Aeronautics*, 67, pp. 281.

18. Boo Ho Yang, Haruhiko Asada, "Hybrid Linguistic / Numeric Control of Deburring Robots based on Human Skills", *Proceedings of IEEE International Conference on Robotics and Automation*, Vol. 3, pp. 1467 - 1474, 1992.

19. Brian J Waibel, H Kazerooni, "Theory and Experiments on the Stability of Robotic Compliance Control", *IEEE Transactions on Robotics and Automation*, Vol.7, no. 1, pp. 95 - 104, February 1991.

20. D R Broome, Q Wang, A R Greig, "Adaptive Compliant Control for an Inspection robot system", *IEE Proceedings-D*, Vol.140, no.3, pp. 191 - 197, May 1993.

21. Bruce G Buchanan, Tom M Mitchell, R G Smith, C R Johnson, "Models of Learning Systems", *Encyclopedia of Computer Science and Technology*, vol. 11.

22. M E Caine, "Chamferless Assembly of Rectangular parts in two and three Dimensions", *MS thesis*, 1985, MIT, Cambridge

23. Charles E Wilson and Peter Sadler, "Kinematics and Dynamics of Machinery", *Harper Collins College Publishers*, NY

24. M R Cutkosky, P K Wright, "Position sensing Wrists for Industrial Manipulators", *12th International Symposium on Industrial Robots*, pp. 427 - 438, 1982.

25. Daniel Croteau, "Developpement d'un Capteur d'efforts a six degrees de liberte", *Master's thesis*, ETS, Montreal, August 1995.

26. Daniel E Whitney, "Historical Perspective and State of the Art in Robot Force Control", *The International Journal of Robotics Research*, Vol.6, no.1, pp. 3 - 13, Spring 1987.

27. David L Goetsch, "Advanced Manufacturing Technology", by *SME & Delmar Publishers* Inc., 1995

28. T L DeFazio. et.al, "Feedback in Robotics for Assembly and Manufacture", 1982, R-1536, Charles Stark Laboratory, *MIT Report*, Cambridge

29. J De Schutter and H Van Brussel, "Compliant Robot Motion I - A Formalism for specifying compliant motion tasks", *The International journal of Robotics Research*, vol. 7, no. 4, August 1988.

30. J De Schutter and H Van Brussel, "Compliant Motion II - A control approach on external control loops", *The International journal of Robotics Research*, vol. 7, no.4, August 1988.

31. Denis Richard, "Developpement d'un Capteur de position sans Contact pour la Robotique", *Master's thesis*, ETS, Montreal, April 1995.

32. A M Dore, E K Lo, "Economic evaluation of robot-based assembly systems", *International Journal of Production Research*, 1991, vol.29, no.2, 267 -76

33. Dragon Stokic, Miomir Vukobratovic, Dragan Hristic, "Implementation of Force feedback in Manipulation Robots", *The International Journal of Robotics Research*, Vol.5, no.1, pp. 66 - 76, Spring 1986.

34. S H Drake, "Using Compliance in Lieu of Sensory Feedback for Automatic Assembly", *Doctor of Science thesis, MIT,* September 1977.

35. Eric E Aboaf, Steven M Drucker, Christopher G Atkeson, "Task Level Robot learning : Juggling a Tennis Ball more Accurately", *Proceedings of IEEE International Conference on Robotics and Automation*, Vol. 3, pp. 1290 - 1295, 1989.

36. Geoffrey Boothroyd, "Assembly Automation and Product Design", 1989, *Marcel Dekker, Inc.*, New York.

37. S J Gordon, "Automated assembly using Feature localisation", *Ph.D thesis*, February 1987, MIT, Cambridge.

38. Gullapalli. V, Roderic A Grupen, Andrew G Barto, "Learning Reactive Admittance Control", *Proceedings of IEEE International Conference on Robotics and Automation*, Vol. 3, pp. 1475 - 1480, 1992.

39. Haruhiko Asada, Boo- Ho yang, "Skill Acquisition from Human Experts through Pattern processing of Teaching data", *Proceedings of IEEE International Conference on Robotics and Automation*, Vol. 3, pp. 1302 - 1307, 1989.

40. Haruhiko Asada, Haruo Izumi, "Direct Teaching and Automatic Program Generation for the Hybrid Control of Robot Manipulators", *Proceedings of IEEE International Conference on Robotics and Automation*, Vol. 3, pp. 1401 - 1406, 1987.

41. Haruhiko Asada, Yukio Asari, "The Direct Teaching of Tool Manipulation Skills via the Impedance Identification of Human Motions", *Proceedings of IEEE International Conference on Robotics and Automation*, Vol. 3, pp. 1269 - 1274, 1988.

42. M P Hennessey, "Compliant Part Mating and Minimum Energy Chamfer Design", *MS thesis*, 1982, MIT, Cambridge.

43. Herbert A Simon, "Why Should Machines Learn ?", Machine Learning, Vol. 1, 1983, *Tioga Publishing Co.*, Palo alto, California.

44. Hirochika Inoue, "Force Feedback in Precise Assembly tasks", MIT Report AD-A011 369 prepared for Office of Naval Research, *Advanced Research Projects Agency*, August 1974.

45. S H Hopkins, C J Bland, M H WU, "Force Sensing an aid to assembly", *International Journal of Productiona Research*, 1991, vol. 29, no. 2, 293 - 301.

46. J K Salisury, "Active Stiffness Control of a Manipulator in Cartesian Coordinates", *Proceedings 19th IEEE Conference Decision Control*, pp. 95 - 100, 1980.

47. Jack Rebman, "Compliance for Robotic Assembly Using Elastomeric Technology", *9th International Symposium on Industrial Robots*, pp. 153 - 166, 1979.

48. James L Peterson, "Petri Net Theory and the Modeling of Systems", 1993, *Prentice-Hall, Inc.*, NJ.

49. Joseph Edward Shigley and Larry D Mitchell, "Mechanical Engineering Design", 1983, *McGraw-Hill Book Co.*,.

50. Kei Hara & Ryuichi Yokogawa, "Precision Insertion by Heuristic Search with Fuzzy Pattern Matching", *JSME International Journal*, series III, vol. 35, no.2, 1992.

51. C J Klien, "Generation and Evaluation of Assembly Sequence alternatives", *MS thesis*, 1987, MIT, Cambridge.

52. E J Lanzolitta. "Robot Skill Acquisition for Impact Tasks", *MS thesis*, 1992, MIT, Cambridge.

53. T S Lindsay, P R Sinha, P P Richard, "An Instrumented Compliant wrist for Robotic Applications", *Proceedings of the IEEE International Conference on Robotics and Automation*, Vol.2, pp. 648 - 653, 1993.

54. S Lee, H Asada, "Assembly of parts with irregular surfaces using active force sensing", *IEEE Journal of Robotics and Automation*, pp. 2639 - 2644, 1994.

55. J Leysen, H Van Brussel, J De Schutter, "An Augmented task level programming system for force controlled assembly operations", *IEEE Journal of Robotics and Automation*, pp. 1258 - 1263, 1992.

56. Mathew T Mason, "Compliance and Force Control for Computer Controlled Manipulators", *IEEE Transactions on Systems, Man and Cybernetics*, Vol. SMC 11, no.6, pp. 418 - 432, 1981.

57. Maureen Candill, Charles Butler, "Understanding Neural Networks: Computer Explorations", Vol. 1 and 2, 1992, *The MIT Press, Massachusetts*.

58. B J McCarragher, "A Discrete Event Dynamics Systems Approach to Robotic Assembly Tasks", *Ph.D Thesis*, 1992, MIT, Cambridge.

59. Miomir Vukobratovic & Atanansko Tuneski, "Contact Control concepts in Manipulator Robotics - An overview", *IEEE Transactions on Industrial electronics*, vol. 41, no.1, February 1994.

60.N Mimura, Y Funahashi, "Parameter Identification of Contact conditions by active Force sensing", *IEEE Journal of Robotics and Automation*, pp. 2645-2650, 1994.

61. Mongi A Abidi, Rafael C Gonzalez, "Data Fusion in Robotics and Machine Intelligence", 1992, *Academic Press Inc., San Diego.*

62. C C Nguyen, Z Antrazi, Z L Zhou, C E Campbel Jr, "Analysis and Experimentation of a Stewart platform based Force/Torque Sensor", *International Journal of Robotics and Automation*, Vol. 7, no. 3, pp. 133 - 140.

63. C C Nguyen, S S Antrazi, Z Zhou, "Analysis and Implementation of a 6 DOF Stewart platform-based force sesnor for passive compliant robotic assembly", *Conference Proceedings - IEEE SOUTHEASTCON*, 2, pp. 880 -884.

64. Nicholas Krouglicof, Alain Desrochers, Denis Richard, "On Line Robot Trajectory generation through Real-Time Optical Tracking".

65. S R Oh, R L Hollis, S E Salcudean, "Precision Assembly with a Magnetically Levitated Wrist", *Proceedings of IEEE International Conference on Robotics and Automation*, pp. 127 - 133, 1993.

66. Onda Nobuhiko, Asakawa Kazuo, Kamada Toru, Akita Tadashi, "Precise Robotic Assembly using Active Compliance Control", *16th International Symposium on Industrial Robots*, pp. 445 - 455, 1986.

67. D T Pham, S H Yeo, "Gripper Design and Selection", *International Journal of Production Research*, vol. 29, 1991.

68. Philip D Wasserman, "Neural Computing ; Theory and Practice", 1989, *Van Nostrand Reihold, New York.*

69. H Qiao, B S Dalay, R M Parkin, "Precise robotic chamferless peg-hole insertion operation without force feedback and RCC", *IMechE Journal, Proceedings of Institution of Mech Engrs*, pp. 89 -104, 1994.

70. T Radhakrishnan, "Combined effects of Linear and Angular errors in PC board assembly", *International Journal of Production Research*, 1992, vol. 30, no. 5, 1037 - 44

71. M H Raibert and J J Craig, "Hybrid Position / Force Control of Manipulators", *Journal of Dynamic Systems, Measurement and Control*, Vol. 102, pp. 126 - 133, June 1981.

72. Ralph L Hollis, S Salcudean, A Peter Allen, "A six Degree of freedom Magnetically Levitated Variable Compliance Fine Motion Wrist: Design, Modelling and Control", *IEEE Transactions on Robotics and Automation*, Vol.7, no.3, pp. 320 - 332, June 1991.

73. C Reboulet, A Robert, H Poilve, A Gaillet, "Hybrid Position Force application - Application to assembly", *15th International Symposium on Industrial Robots*, pp. 157 - 164, 1985.

74. Richard P Paul, "Problems and Research Issues Associated with the Hybrid Control of Force and Displacement", *Proceedings of the IEEE Conference on Robotics and Automation*, pp. 1966 - 1971, 1987.

75. Robert N Boggs, "Shear Pads let Device correct Misalignments", *Design News*, August 1992.

76. Sheng Liu, Haruhiko Asada, "Transfering Manipulative Skills to Robots: Representation and Acquisition of Tool Manipulative Skills using a Process Dynamics Model", *Transactions of the ASME: Journal of Dynamic Systems, Measurement and Control*, Vol. 114, pp. 220 - 228, 1992.

77. S N Simunovic, "An Information Approach to Part Mating", *Doctor of Science thesis*, 1979, MIT, Cambridge.

78. Steven D Eppinger, Warren P Seering, "Three Dynamic Problems in Robot Force Control", *IEEE Transactions on Robotics and Automation*, Vol. 8, no.6, pp. 751 - 758, December 1992.

79. D Stewart, "A Platform with Six Degrees of Freedom", *Proceedings of Institute of Mechanical Engineering*, 1965, 180 (5), Pt. I, pp. 371 - 386.

80. Susan Gottschlich, C Ramos, D Lyons, "Assembly and task Planning: A Taxonomy", *IEEE Robotics and Automation magazine*, pp. 4-12, September 1994.

81. Tomas Lozano Perez, Matthew T Mason, Russell H Taylor, "Automatic Synthesis of Fine Motion Strategies for Robots", *The International journal of Robotics Research*, vol.3, no.1, spring 1984.

82. M Uchiyama, Y Nakamura, K Hakomori, "Evaluation of robot force sensor using singular value decomposition", Journal of the Robotics Applications, *Journal of the Robotics Society of Japan*, 5(1), pp. 4 -10.

83. M Uchiyama, E Bayo, E Palma-Villalon, "A Systematic Procedure to minimise a Performance Index for Robot Force Sensors", *Transactions of the ASME*, 113, pp. 388 -394.

84. E G Vaaler, "A Machine Learning Algorithm (Logic Branching) for Automated Assembly", *Doctor of Science thesis*, 1991, MIT, Cambridge.

85. H Van Brussel, J Simons, "The Adaptable Compliance concept and its use for Automatic assembly by Active force feedback accommodations", *9th International Symposium on Industrial Robots*, pp. 167 - 181, 1979.

86. P C Watson, S H Drake, "Pedestal and Wrist force sensor for automatic assembly", *Proceedings of the 5th International Symposium on Industrial Robots*, pp. 501 - 511.

87. D E Whitney, "Force Feedack Control of Manipulator fine motions", *ASME Journal of Dynamic Systems, Measurement and Control*, pp. 91 - 97, 1977.

88. D E Whitney, "Quasi-static Assemly of Compliantly Supported Rigid Parts", *ASME Journal of Dynamic Systes, Measurement and Contol*, vol.104, pp. 65 - 77, 1982.

89. D E Whitney, J L Nevins, "What is Remote Center Compliance and What can it do?", *9th International Symposium on Industrial Robots*, pp. 135 - 152, 1979.

90. R F Wolffenbuttel, K M Mahmoud, P P L Regtien, "Multiaxis Compliant wrist sensor for use in Automated assembly with Industrial Robots", *IEEE Instrumentation Measurement Technology conference*, pp. 54 - 59, 1990.

91. R F Wolffenbuttel, K M Mahmoud, P P L Regtien, "Compliant Capacitive Wrist sensor for use in Industrial Robots", *IEEE transactions on Instrumentation and Measurement*, Vol. 39, no.6, pp. 991 - 997, December 1990.

92. R F Wolffenbuttel, P P L Regtien, "Capacitance to Phase Angle Conversion for the Detection of extremely small Capacities", *IEEE Transactions on Instrumentation and Measurement*, Vol. IM-36, no. 4, pp. 868 - 872, December 1987.

93. Yangsheng Xu, R P Paul, "Robotic Instrumented Compliant Wrist", *Journal of Engineering for Industry, Transactions of the ASME*, Vol. 114, pp. 120 - 123, February 1992.

94. Yangsheng Xu, R P Paul, "A Robot Compliant Wrist system for Automated assembly", *Proceedings of IEEE International Conference on Robotics and Automation*, pp. 1750 - 1755, 1990.

95. Yeung Dit-Yan, George A Gekay, "Using a Context Sensitive Learning Network for Robot Arm Control", *Proceedings of IEEE International Conference on Robotics and Automation*, Vol. 3, pp. 1441 - 1447, 1989.

Appendix A

A1 FORCE / MOMENT COUPLING MATRIX

This appendix discusses the derivation of the coupling matrix of the Wrist sensor. As mentioned in the main text, the present sensor was a scaled down model of a prototype sensor designed and built at École de technologie supérieure (ÉTS). The coupling matrix is an important part of the measuring system, as such it is being included here in order to ensure completeness of the work.

A2 FORCES ON EACH ELEMENT

The matrix for the coupling elements for different components of the force effort is given below. The three force components and three moment components are obtained by multiplying the force-moment matrix by the six forces on each element of the sensor. In order to get the values of these terms, it is essential that the geometry of the sensor is known. In practice, the top and bottom plate are connected by these six elements, Fig. A1 and Fig. A2 represent the configuration of the sensor and the orientation of the sensor, respectively. From Fig. A3, the force components in the vertical and horizontal directions can be calculated and they will essentially be the same.

The vertical component of force is given by multiplying the sine of the angle with the force acting on the element and acts in the direction of the Z axis.

$$\text{The Vertical component is} = Fi\frac{h}{L} \qquad \text{.....} \qquad \text{A1}$$

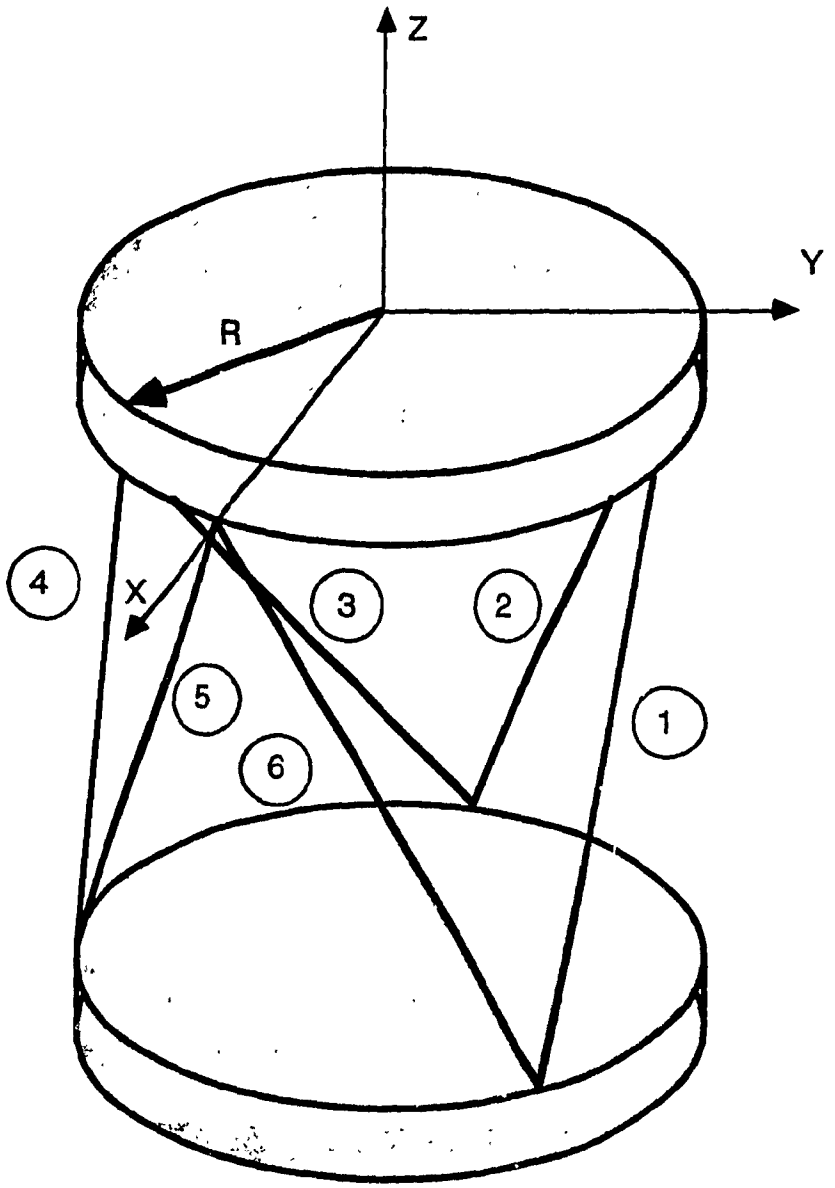


Fig. A1 Geometry of the force sensor [Croteau 1995]

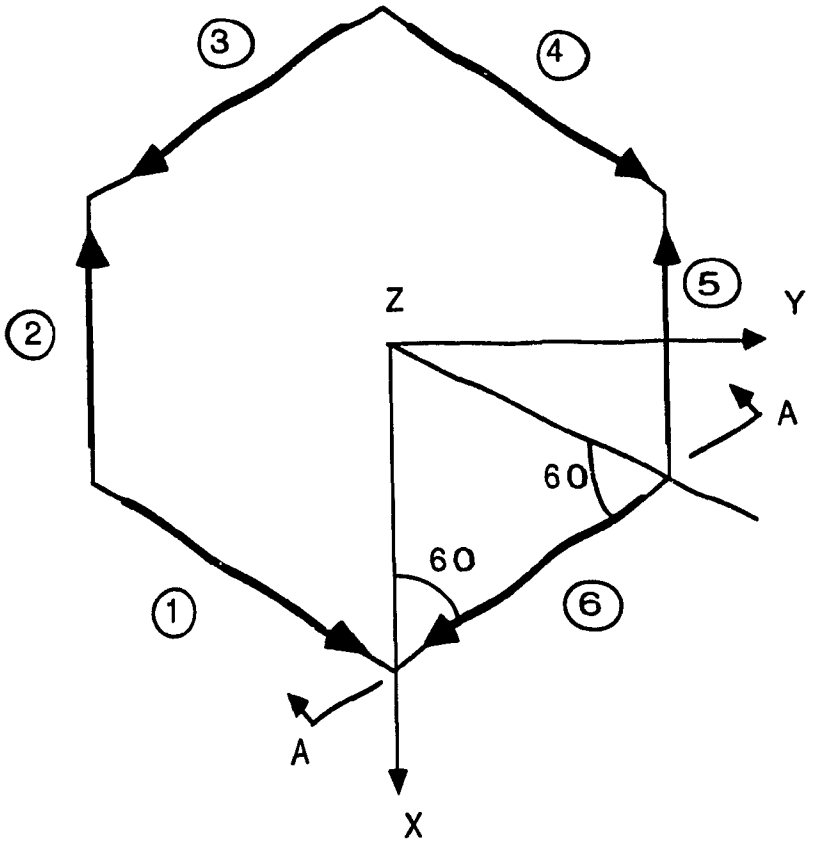


Fig. A2 Orientation of Sensor [Croteau 1995]

and the horizontal component of the force is given by multiplying the cosine of the angle which is $(R - 2d) / L$, with the force acting on the element and acts in either the X or Y direction.

$$\text{The Horizontal component} = Fi\frac{(R-2d)}{L} \quad \ldots\ldots\ldots\ldots \quad A2$$

As such, for each element the vertical and horizontal forces are given as:

I - Element 1

$$F_{1X} = F_1 \frac{(R-2d)}{L}.Sin30 \quad \ldots\ldots\ldots\ldots \quad A3$$

$$F_{1Y} = F_1 \frac{(R-2d)}{L}.Cos30 \qquad \text{..... A4}$$

$$F_{1Z} = F_1 \frac{h}{L} \qquad \text{..... A5}$$

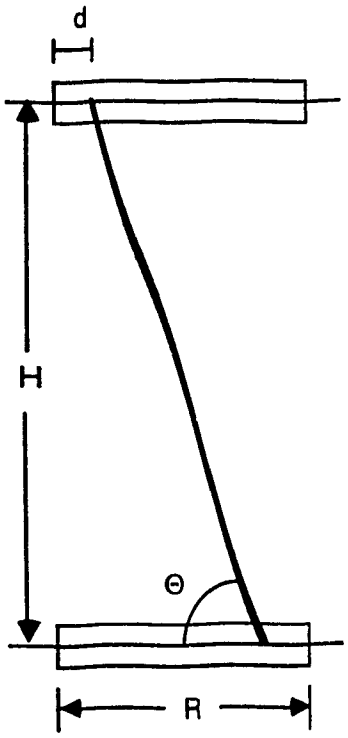


Fig. A3 Section at AA in figure A2.

II - Element 2

$$F_{2X} = F_2 \frac{(R-2d)}{L}.Sin30 \qquad \text{..... A6}$$

$$F_{2Y} = F_2 \frac{(R-2d)}{L}.Cos30 \qquad \text{..... A7}$$

$$F_{2Z} = F_2 \frac{h}{L} \qquad \text{..... A8}$$

III - Element 3

$$F_{3X} = F_3 \frac{(R-2d)}{L}.Sin30 \quad \text{.....} \quad A9$$

$$F_{3Y} = F_3 \frac{(R-2d)}{L}.Cos30 \quad \text{.....} \quad A10$$

$$F_{3Z} = F_3 \frac{h}{L} \quad \text{.....} \quad A11$$

IV - Element 4

$$F_{4X} = F_4 \frac{(R-2d)}{L}.Sin30 \quad \text{.....} \quad A12$$

$$F_{4Y} = F_4 \frac{(R-2d)}{L}.Cos30 \quad \text{.....} \quad A13$$

$$F_{4Z} = F_4 \frac{h}{L} \quad \text{.....} \quad A14$$

V - Element 5

$$F_{5X} = F_5 \frac{(R-2d)}{L}.Sin30 \quad \text{.....} \quad A15$$

$$F_{5Y} = F_5 \frac{(R-2d)}{L}.Cos30 \quad \text{.....} \quad A16$$

$$F_{5Z} = F_5 \frac{h}{L} \quad \text{.....} \quad A17$$

VI - Element 6

$$F_{6X} = F_6 \frac{(R-2d)}{L}.Sin30 \quad \text{.....} \quad A18$$

$$F_{6Y} = F_6 \frac{(R-2d)}{L}.Cos30 \quad \text{.....} \quad A19$$

$$F_{6Z} = F_6 \frac{h}{L} \quad \text{.....} \quad A20$$

A3 FORCE IN X - AXIS

The first line in the matrix is to determine the force applied in the X axis of the sensor. A force applied in the positive direction on the X axis has each of the elements in the following state.

element 1 - tension

element 2 - tension

element 3 - compression

element 4 - tension

element 5 - tension

element 6 - compression

To calculate the positive effort in X axis, the terms corresponding to the elements in compression are given a negative sign. As such the F_X force is given by,

$$F_X = F_{1X} + F_{2X} - F_{3X} + F_{4X} + F_{5X} - F_{6X} \quad \ldots \quad \text{A21}$$

The expression for each F_{iX} has been derived already. As such, the first line of the matrix is obtained by multiplying the force vector in each element with these components.

$$\left[\frac{(R-2d)}{L}\ \frac{(R-2d)}{L}\ \frac{-(R-2d)}{L}\ \frac{(R-2d)}{L}\ \frac{(R-2d)}{L}\ \frac{-(R-2d)}{L}\right] \quad \ldots\ldots \quad \text{A22}$$

Multiply each term by the force measured corresponding to the contribution in X by each of the elements.

A4 FORCE IN Y - AXIS

Similarly the force in the Y axis can be calculated and the second line for the matrix can be arrived at. On application of a force in the positive sense around the Y axis produces the following effect in each element.

element 1	- tension
element 2	- compression
element 3	- no effect
element 4	- tension
element 5	- compression
element 6	- no effect

As such the value of force F_Y is given as,

$$F_Y = F_{1Y} - F_{2Y} + F_{4Y} - F_{5Y} \qquad \text{..... A23}$$

Expressions for each F_{iY} component have been derived earlier. Therefore, the second line of the matrix is obtained by multiplying the force vector in each element with each component.

$$\left[\frac{(R-2d)\sqrt{3}}{2L}\ \frac{-(R-2d)\sqrt{3}}{2L}\ 0\ \frac{(R-2d)\sqrt{3}}{2L}\ \frac{-(R-2d)\sqrt{3}}{2L}\ 0\right] \quad \text{..... A24}$$

A5 FORCE IN Z - AXIS

For the third line in the matrix, applying a force in the positive direction on the Z axis, produces the following effect on each element.

element 1 - tension

element 2 - tension

element 3 - tension

element 4 - tension

element 5 - tension

element 6 - tension

The force in Z is given as

$$F_Z = F_{1Z} + F_{2Z} + F_{3Z} + F_{4Z} + F_{5Z} + F_{6Z} \qquad \text{A25}$$

The expression corresponding to each F_{iZ} has been derived earlier. The force in Z is determined by multiplying these elements with the force vector in each element.

$$\frac{h}{L}\ \frac{h}{L}\ \frac{h}{L}\ \frac{h}{L}\ \frac{h}{L}\ \frac{h}{L} \qquad \text{..... A26}$$

A6 MOMENT ABOUT X - AXIS

To determine the moment terms for each element, the procedure is the same, as that to determine the forces. It is necessary to find the distance from the point of attachment of these elements as shown in Figs. A4 to A6 .

On application of a positive moment about the X axis on the top plate, each element is in the following state.

element 1 - compression

element 2 - tension

element 3 - tension

element 4 - tension

element 5 - compression

element 6 - compression

The terms associated with the compression have a negative sign. The vertical component in each element opposes an effort due to the couple created in the Y axis. The moment in X axis is given as:

$$M_X = - F_{1Z}\ dcos30 + F_{2z}\ dcos30 + F_{3Z}\ Rcos30 + F_{4Z}\ (Rcos30 - dcos30) - F_{5Z}\ (Rcos30 - dcos30) - F_{6Z}\ Rcos30 \quad \text{.....} \quad \text{A27}$$

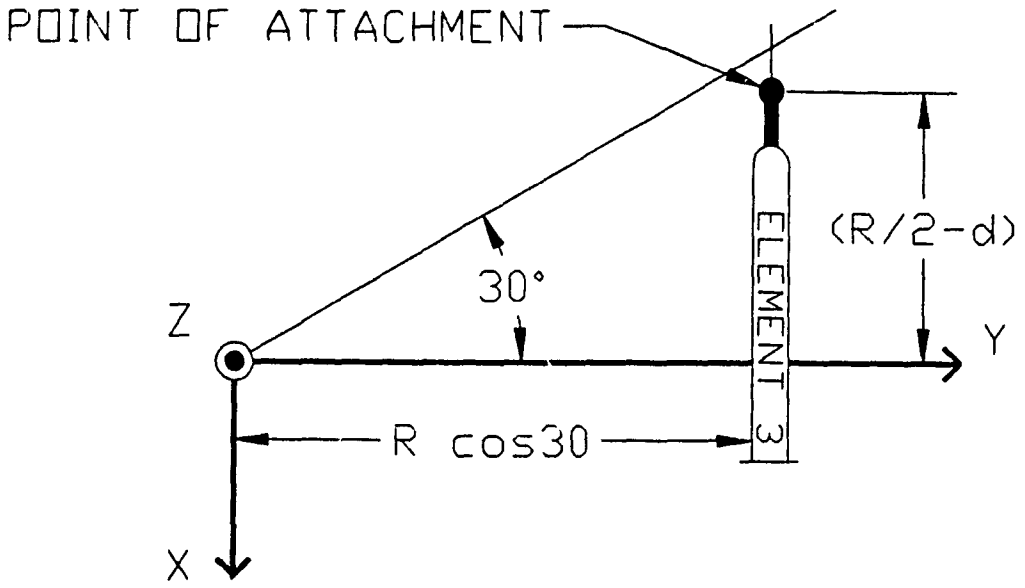


Fig. A4 Attachment of Element 1 or 2 with top plate [Croteau 1995]

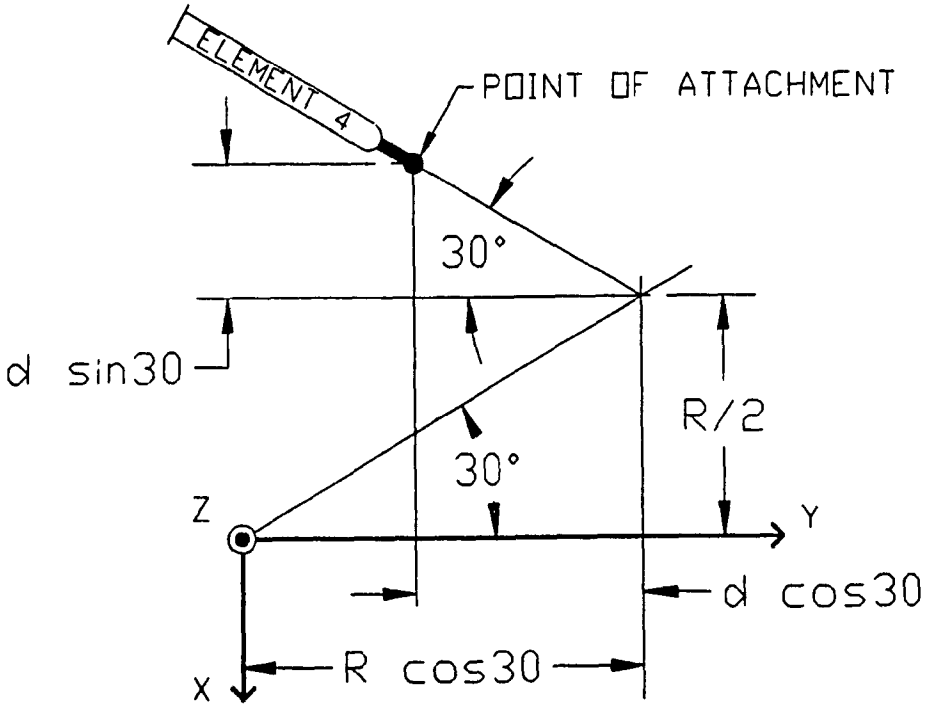


Fig. A5 Attachment of Element 3 or 4 with top plate [Croteau 1995]

substituting respective values for the vertical components in the above, the fourth line in the matrix is given as

$$\left[\frac{-hd\sqrt{3}}{2L}\ \frac{hd\sqrt{3}}{2L}\ \frac{hR\sqrt{3}}{2L}\ \frac{h(R-d)\sqrt{3}}{2L}\ \frac{-h(R-d)\sqrt{3}}{2L}\ \frac{-hR\sqrt{3}}{2L}\right] \quad \text{..... A28}$$

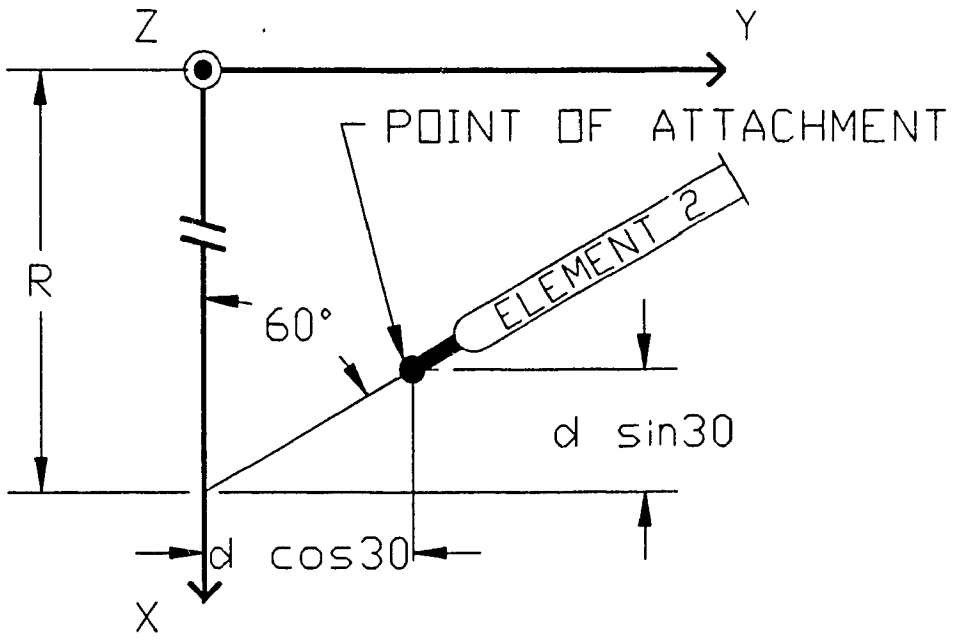


Fig. A6 Attachment of Element 5 or 6 with top plate [Croteau 1995]

A7 MOMENT ABOUT Y-AXIS

To determine the terms of the moment about the Y axis, the process is the same as that carried out in the x axis above. On application of a positive moment about the Y axis on the top plate, each element will be in the following state.

element 1 - compression

element 2 - compression

element 3 - tension

element 4 - tension

element 5 - tension

element 6 - tension

The expression for moment in Y axis is given as:

$$M_Y = -F_{1Z}(R - d\sin 30) - F_{2z}(R - d\sin 30) + F_{3Z}(R/2 - d) + F_{4Z}(R/2 + d\sin 30) + F_{5Z}(R/2 + d\sin 30) + F_{6Z}(R/2 - d) \quad \text{..... A29}$$

Substituting the respective vertical components in the expression, the fifth line of the matrix is given as:

$$\left[\frac{-h(R-\frac{d}{2})}{L} \quad \frac{-h(R-\frac{d}{2})}{L} \quad \frac{h(\frac{R}{2}-d)}{L} \quad \frac{h(R+d)}{2L} \quad \frac{h(R+d)}{2L} \quad \frac{h(\frac{R}{2}-d)}{L}\right] \quad \text{..... A30}$$

A8 MOMENT ABOUT Z-AXIS

To complete the coupling matrix, the terms associated with the moments in z axis are to be determined. The terms to be determined are more complex than those associated with the X and Y axes. Applying a positive moment about the Z axis, produces the following state in each element.

element 1	- tension
element 2	- compression
element 3	- tension
element 4	- compression
element 5	- tension
element 6	- compression

As the derivation of each term is complex, the terms for every element will be derived separately.

$$Mz_4 = \frac{-F_4(R-2d)}{2L} * \frac{(R-d)\sqrt{3}}{2} - \frac{F_4(R-2d)\sqrt{3}}{2L} * \frac{(R+d)}{2}$$

..... A31

ELEMENT 1:

The total moment about the Z axis is obtained by summing up the moments in the Z for each of the elements.

$$M_Z = M_{1Z} + M_{2Z} + M_{3Z} + M_{4Z} + M_{5Z} + M_{6Z} \quad \text{..... A32}$$

In this expression the moment for element 1 about the Z axis is given as:

$$M_{Z1} = F_{1X}\, d\cos 30 + F_{1Y}(R - d\sin 30) \quad \text{..... A33}$$

Substituting for F_{1X} and F_{1Y} in the above, the Moment M_{Z1} is obtained as

$$M_{Z1} = \frac{F_1(R-2d)}{2L} * \frac{d\sqrt{3}}{2} + \frac{F_1(R-2d)\sqrt{3}}{2L} * \left[R - \frac{d}{2}\right] \quad \text{..... A34}$$

Simplifying further gives the first term in the sixth line of the matrix as:

$$M_{61} = \frac{R(R - 2d)\sqrt{3}}{2L} \quad \text{..... A35}$$

Similarly, other terms of the sixth line are derived for each element.

ELEMENT 2:

$$M_{Z2} = - F_{2X}\, d\cos 30 - F_{2Y}(R - d\sin 30) \quad \text{..... A36}$$

substituting for F_{2X} *and* F_{2Y}, the expression for M_{Z2} is given as

$$M_{Z2} = \frac{-F_2(R-2d)}{2L} * \frac{d\sqrt{3}}{2} - \frac{F_2(R-2d)\sqrt{3}}{2L} * \left[R - \frac{d}{2}\right] \quad \text{..... A37}$$

Simplifying further, the second term in the sixth line of the matrix is given as

$$M_{62} = \frac{-R(R-2d)\sqrt{3}}{2L} \quad \text{.....} \quad \text{A38}$$

ELEMENT 3:

From element 3 the contribution is,

$$M_{Z3} = F_{3x}R\cos 30 \quad \text{.....} \quad \text{A39}$$

On substituting the value of F_{3X} the expression results in to,

$$M_{Z3} = \frac{F_3R(R-2d)\sqrt{3}}{2L} \quad \text{.....} \quad \text{A40}$$

Simplifying further, reduces the expression to,

$$M_{63} = \frac{R(R-2d)\sqrt{3}}{2L} \quad \text{.....} \quad \text{A41}$$

ELEMENT 4:

For element 4, two components contribute to M_{Z4} and are given as,

$$M_{Z4} = -F_{4x}(R\cos 30 - d\cos 30) - F_{4y}\left[\frac{R}{2} + d\sin 30\right] \quad \text{.....} \quad \text{A42}$$

substituting for the F_4 terms, the expression simplifies to,

Finally, the value of M_{64} in the coupling matrix is,

$$M_{64} = \frac{-R(R-2d)\sqrt{3}}{2L} \quad \text{.....} \quad \text{A43}$$

ELEMENT 5:

For element 5 the moment is calculated as,

$$M_{Z5} = F_{5X}(R\cos 30 - d\cos 30) - F_{5Y}\left[\frac{R}{2} + d\sin 30\right] \quad \text{..... A44}$$

simplifying further, on substituting the values of F_5,

$$M_{Z5} = \frac{F_5(R-2d)}{2L} * \frac{(R-d)\sqrt{3}}{2} + \frac{F_5(R-2d)\sqrt{3}}{2L} * \frac{(R+d)}{2}$$

..... A45

The fifth term in the sixth line of the matrix, after simplification is given as,

$$M_{65} = \frac{R(R-2d)\sqrt{3}}{2L} \quad \text{..... A46}$$

ELEMENT 6 :

The contribution due to element 6 for the moment about the z axis is given as,

$$M_{Z6} = -F_{6X} R\cos 30 \quad \text{..... A47}$$

Substituting for F_{6X}, the expression results in,

$$M_{Z6} = \frac{-F_6 R(R-2d)\sqrt{3}}{2L} \quad \text{..... A48}$$

The final term in the coupling matrix is given as,

$$M_{66} = \frac{R(R-2d)\sqrt{3}}{2L} \quad \text{..... A49}$$

A9 COUPLING MATRIX

In the matrix, substituting

$$a = \frac{(R - 2d)}{2L} \quad \text{and} \quad b = \frac{h}{L},$$

The coupling matrix, with all the terms is given as,

$$M = \begin{bmatrix} a & a & -2a & a & a & -2a \\ a\sqrt{3} & -a\sqrt{3} & 0 & a\sqrt{3} & -a\sqrt{3} & 0 \\ b & b & b & b & b & b \\ \frac{-bd\sqrt{3}}{2} & \frac{bd\sqrt{3}}{2} & \frac{bR\sqrt{3}}{2} & \frac{b(R-d)\sqrt{3}}{2} & \frac{-b(R-d)\sqrt{3}}{2} & \frac{bR\sqrt{3}}{2} \\ -b\left[R-\frac{d}{2}\right] & -b\left[R-\frac{d}{2}\right] & b\left[\frac{R}{2}-d\right] & b\left[\frac{R+d}{2}\right] & b\left[\frac{R+d}{2}\right] & b\left[\frac{R}{2}-d\right] \\ aR\sqrt{3} & -aR\sqrt{3} & aR\sqrt{3} & -aR\sqrt{3} & aR\sqrt{3} & -aR\sqrt{3} \end{bmatrix}$$

..... A50

Appendix B

Force Measurement Principles In Sensor

This appendix deals with the force measurement in the sensor. The coupling matrix terms are to be evaluated, for the forces being applied. When a force is applied on the element, shown in Fig. B1, there is a change in resistance of the strain gauge, which is converted into voltage and amplified, this amplified signal is passed through an Analog to Digital conerter, demodulated and finally acquired by the computer. A software was designed to acquire and convert the signals into force display, on-line, at the École de technologie supérieure (ÉTS) laboratory [Croteau, 1995].

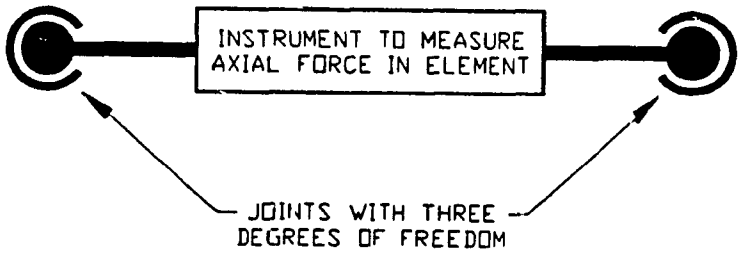


Fig. B1 Sensor element for measuring forces

B1 FORCE CHANGES

In a member the force is given by the general equation,

$$\text{Force, } F = E*A*\varepsilon \quad \ldots\ldots\ldots\ldots \quad B1$$

and the strain ε, is given as change in length divided by the original length.

$$\varepsilon = \Delta L / L \qquad \text{.....} \qquad B2$$

In the sensor elements for a particular force application, the only variable is the strain, as the Young's modulus and the area are constant. This change in strain is monitored and measured.

B2 THE GAUGES

The sensitivity of the gauge is given by, $S_A = (\Delta R / Rg) / \varepsilon$ B3

ΔR represents the variation in resistance of the gauge during any deformation. The value Rg is the resistance of the gauge and is specified by the manufacturer, varying from 120 ohms to 350 ohms depending on the type of the gauge. S_A ,the sensitivity depends on the material being used and the manufacturing method of the gauge.The gauges measure, both the longitudinal and the transverse deformations. The expression is given by,

$$(\Delta R / Rg) = S_L\varepsilon_L + S_T\varepsilon_T \qquad \text{.....} \qquad B4$$

This can be expressed in terms of the gauge factor as,

$$(\Delta R / Rg) = S_L(\varepsilon_L + K_T\varepsilon_T) \qquad \text{.....} \qquad B5$$

Where, $K_T = S_L / S_t$ is the transverse sensitivity of the gauge. The Gauge Factor is a standard used by manufacturers, is a measurement of the sensitivity of the strain gauge

and is defined as the resistance change per unit of initial resistance divided by the applied strain. The strain gauges used were of standard constantan (45 Ni, 55 Cu), with a gauge factor of 2.04. The expression for the ratio of change in resistance to the original resistance is given by,

$$(\Delta R / Rg) = GF\varepsilon_L \qquad \text{.....} \qquad B6$$

B3 MEASUREMENT

The principle of measurement was based on the Wheatstone bridge. The unknown resistance R_X in the circuit is that of the strain gauge, and is measured for a value when the galvanometer reads zero. For equiliibrium at that moment, the value of

$$R_X = (R_1 R_3) / R_2 \qquad \text{.....} \qquad B7$$

The gauges were positioned as shown in Fig. B2, so that the transverse and longitudinal components of the force are taken into consideration.

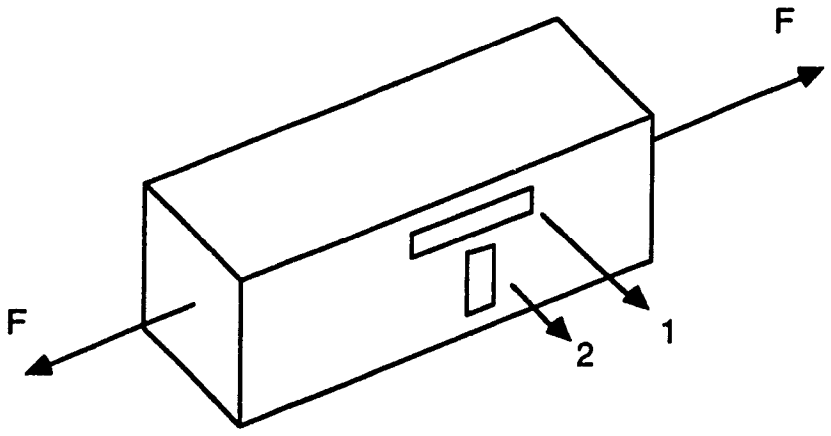


Fig. B2 Mounting of gauges on elements

The circuit in Fig. B3 shows that two gauges measure the deformation in the longitudinal direction and the other two in the transverse direction. The value of resistances 1 and 3 reduces on the application of the force in transverse direction and that of 2 and 4, increases due to application of the longitudinal force.

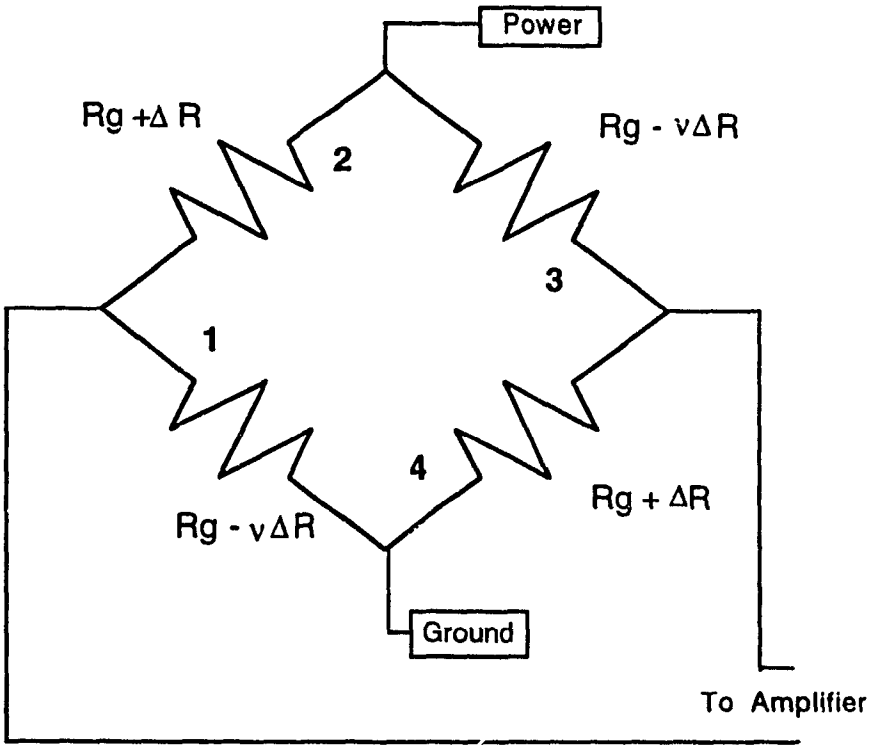


Fig. B3 Change of resistance on load application

The voltage therefore is given as,

$$V_S = \{(R_g + \Delta R) - (R_g - \nu \Delta R)\} * I / 2 \quad \ldots\ldots\ldots \text{B8}$$

On re-arranging, the expression is

$$V_S = \{\Delta R(1 + \nu)\} * I / 2 \quad \text{..... B9}$$

In terms of the Gauge factor the expression for V_S is given as,

$$V_S = \{R_g \varepsilon_L * GF(1 + \nu)\} * I / 2 \quad \text{..... B10}$$

In this equation all factors are constant, except ε_L .

B4 HARDWARE AND SOFTWARE

The hardware details for communication are not being discussed here, as they do not fall under the scope of this work. Details are avialable at ETS. The communication with the PC was through the RS-232 standard serial communication port. A 16 bit micro-controller card manufactured by Motorola was used as it was sufficient for the application.

The micro-controller main program is shown in Chart. II. After INITIALISATION, the processor enters the infinite loop to SAMPLE and VERIFY SERIAL PORT. The first step is to read the numeric signal for the six elements. When sampling is complete, Chart III, the program passes to the next block, VERIFY SERIAL PORT. This procedure is to verify if a character is recieved at the serial port. This character is sent to the micro-ordinator and the algorithm awaits the next character. During the communication if there is a delay, the values are stored in memory. Two characters of 8 bits are transmitted to the micro-ordinator. There are 2 memory buffers established for each of the six elements in

the hardware. Each sampling loop is begun on receipt of a character at the micro-controller.

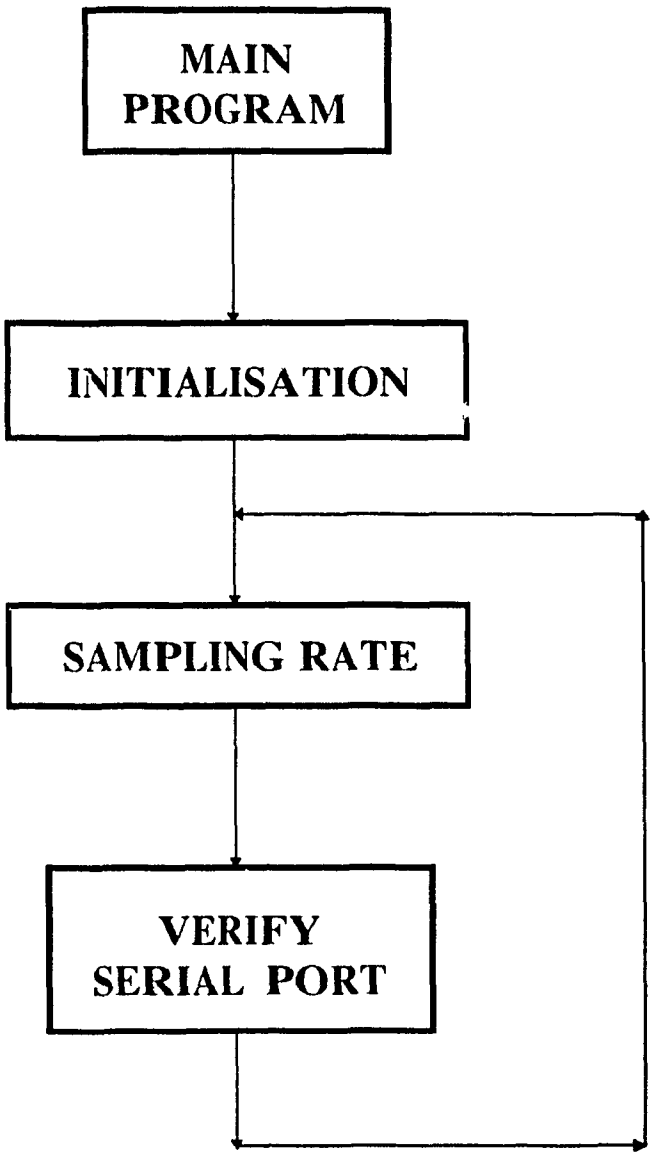


Chart II Micro-controller main program [Croteau 1995]

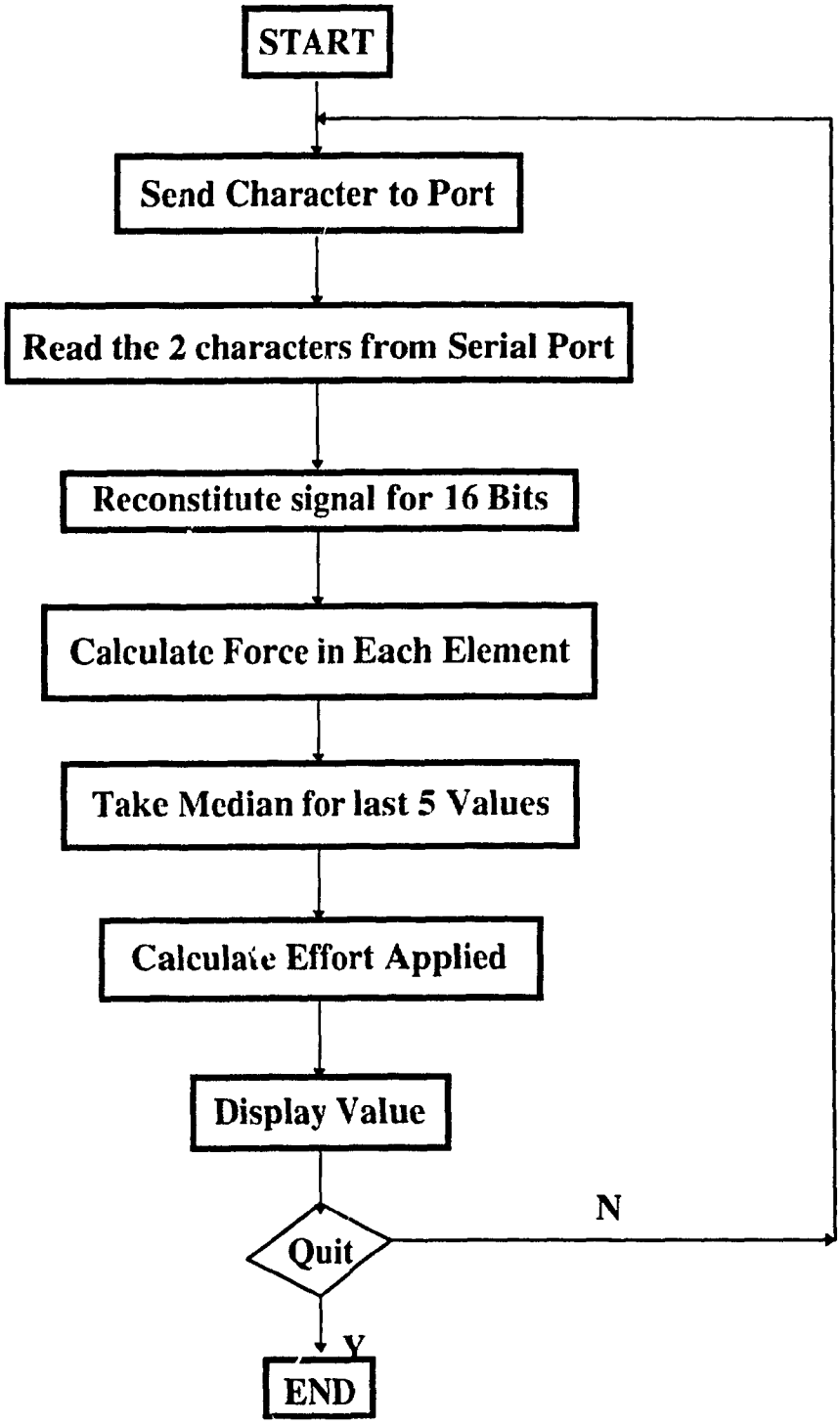


Chart III Algorithm for sampling procedure [Croteau 1995]

Appendix C

Design Details Of Passive Compliance Device

This appendix deals with the design details of the Passive Compliance Device (PCD). The PCD is fitted between the wrist sensor and the end effector of the robot and provides the flexibility during assembly process. The PCD developed is very simple in design and manufacture, and is modular, as discussed in detail in Chapter 4.

To maintain a linear relationship between the force and displacement during assembly, it was decided to have a design based on springs. The springs form the core of the PCD and are being discussed in the following sections. The purpose of this design is to easily transfer the technology to industry. Also, it was felt necessary that additional electronics or hardware should not be integrated into the system, to minimise cost, ensure easy handling and maintenance.

C1 THE FLOAT

The requirement of actuating the PCD evenly, both while a compressive load / tensile load acts on the system, lead to the incorporation of the float. The float, Fig. C1, is the component, that houses springs on top section of the PCD and the bottom section of PCD, and has a clearance of 2.5 mm on each side of the housing. This provides longitudinal and angular clearance for the float when the springs are in compression / tension. The float is connected by a set screw to the mounting pad of the end-effector, and

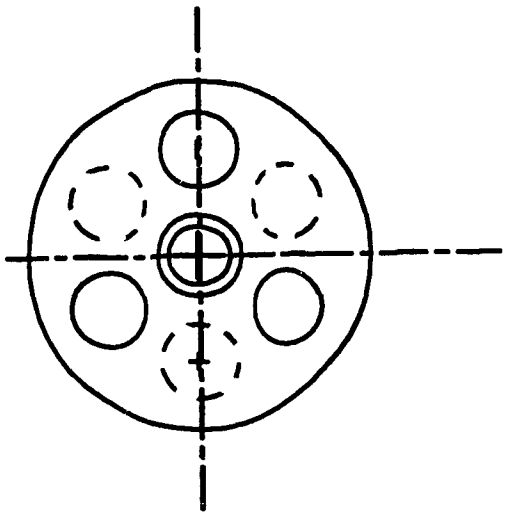


Fig. C1 Float for Passive Compliance device

transfers the misaligning forces / torques that are transmitted through this screw, onto the springs.

C2 SPRING CALCULATIONS

Six springs were selected and mounted in a way that they formed 2 sets of series springs, each set comprising of 3 springs in parallel. The springs were spaced at 120 degrees on either side of the float, causing an overall staggered effect at 60 degrees. Positioning the springs this way helped improve the cocking stiffness of the PCD. Standard springs were selected, based on the nearest design criterion, to help reduce time and cost in manufacturing special springs.

For a load of 6.6 lb the deflection per coil is 0.0839 inches from the standard machinery handbook, calculating the active coils results to 8.5 coils and the total number of coils is equal to 10.5. The nearest spring with these specification is available in trade for 11 coils. The load calculated for 1/10 inch is 5.6 lbs.

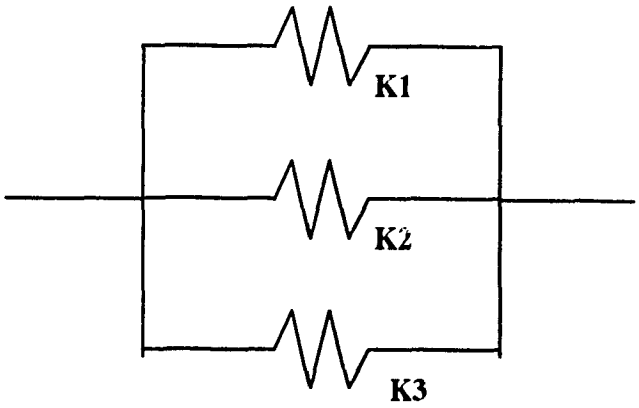


Fig. C2 Schematic of one set of springs in parallel, inside the housing

Stiffness per spring is = 56 lbs/ inch

For Springs in parallel, Equivalent stiffness $K_p = K1 + K2 + K3$ C1

Therefore, K = 3 X 56 = 168 lbs / inch

There are two sets of such springs in series,

Hence, Final stiffness, $1 / K_f = (1 / K_{p1} + 1 / K_{p2})$ C2

resulting in an overall stiffness of the PCD to be 84 lbs / inch.

For this stiffness and load, the coil diameter calculated was 0.065 inches.

In trade, the nearest spring available is with a rectangular cross section of 0.037 x 0.09075 inches, which was suitable for the PCD. Chrome Vanadium die springs readily available were purchased. This material behaves well under shock and impact loading and has good

fatigue strength and endurance limit. The specification of the springs have been included in appendix D.

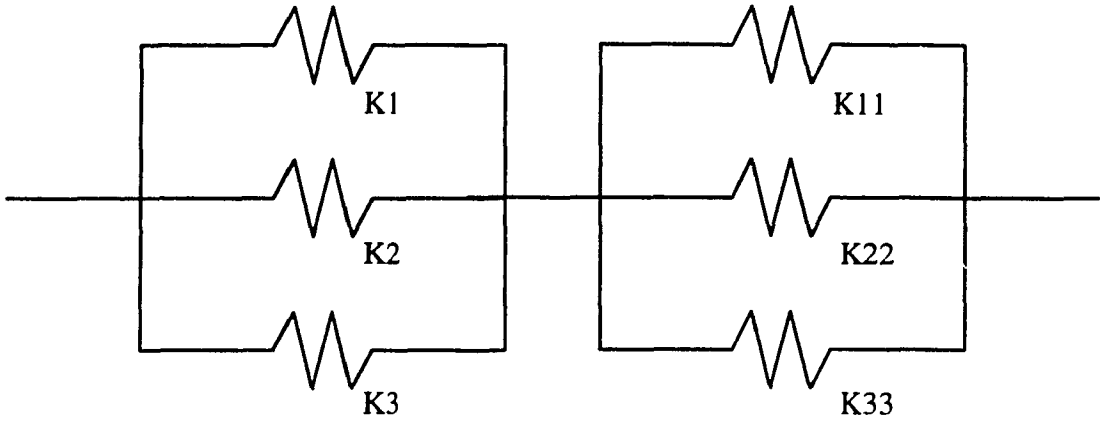


Fig. C3 Schematic of both sets of springs in series

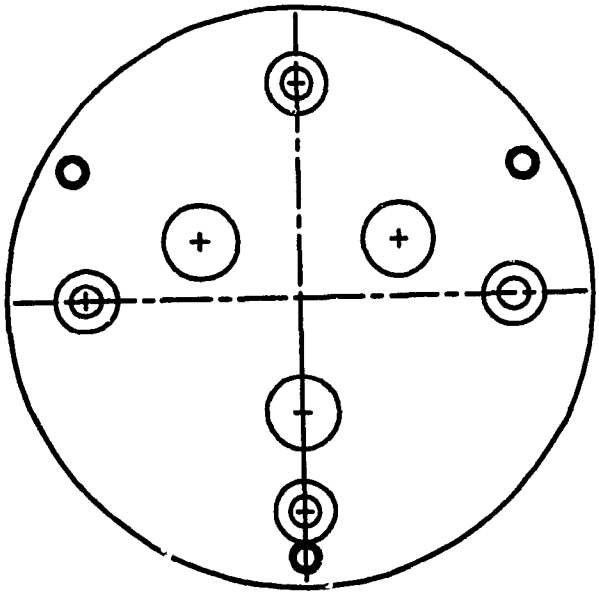


Fig. C4 Top plate for Passive Compliance device

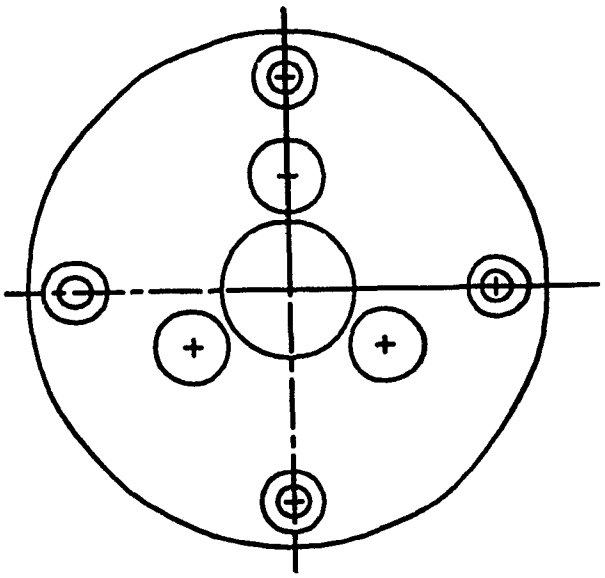


Fig. C5 Bottom plate for Passive Compliance device

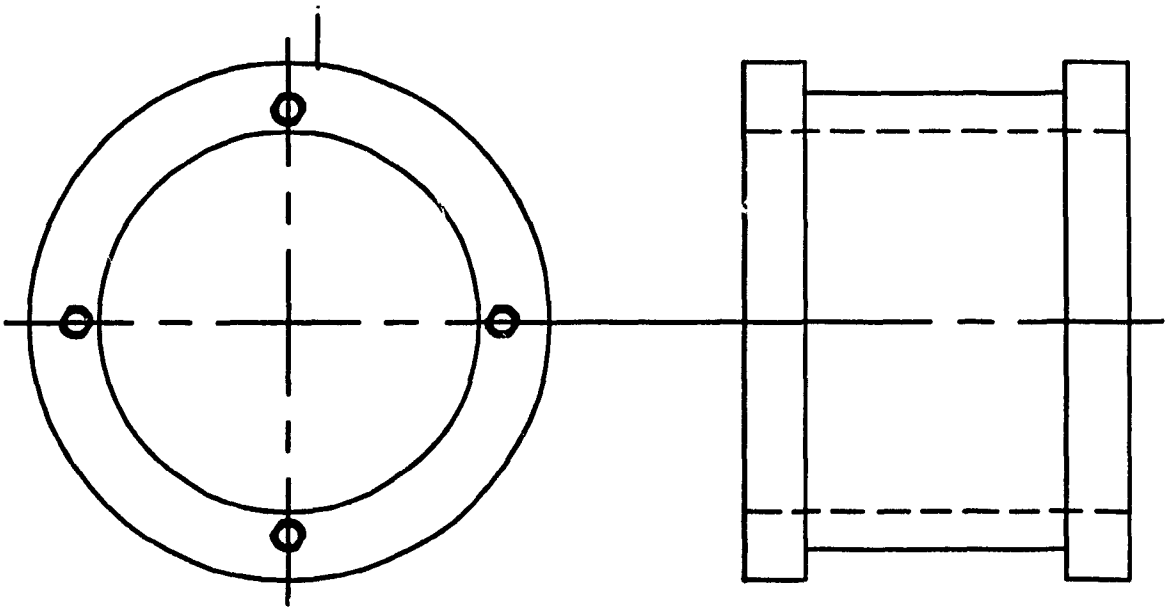


Fig. C6 Housing for Passive Compliance Device

C3 TOP AND BOTTOM PLATES

The top and bottom plates, Figs. C4 and C5, essentially hold and locate the springs and are also the top plate is an interface for the wrist sensor. A minimal thickness was decided for these plates based on the depth to which the springs were to be held, under no-load and pre-load conditions. These plates are fastened to the housing using 4 each M4 x 8 allen screws. Holes were drilled on these plates to match the wrist sensor and the end-effector. The thickness of these plates was 6 mm each, the screws were counter sunk upto 4 mm to flush them with the plates, so that there was no interference while mounting other components.

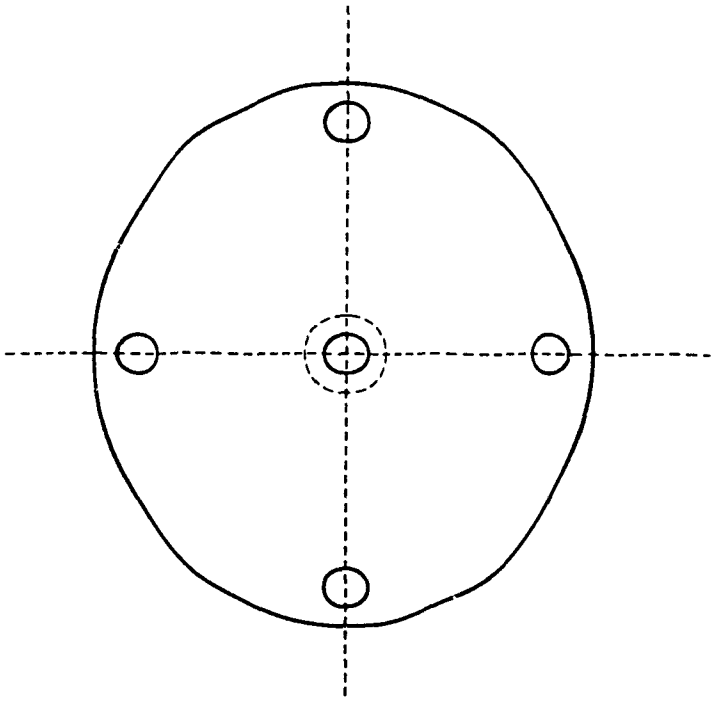


Fig. C7 Mounting Pad for Passive Compliance device

C4 THE HOUSING AND MOUNTING PAD

The housing, Fig. C6, encompasses the float, the springs and a part of the set screw. The mounting pad, Fig. C7, is fastened to the end of the set screw and is an interface for the end-effector.

Aluminum was used to manufacture the PCD as the weight had to be maintained to a minimum, so that the robot pay load was not reduced very much. Testing of PCD is discussed in Appendix E. Figs. C8 to C10 show the PCD being subjected to various loads, and its corresponding behaviour.

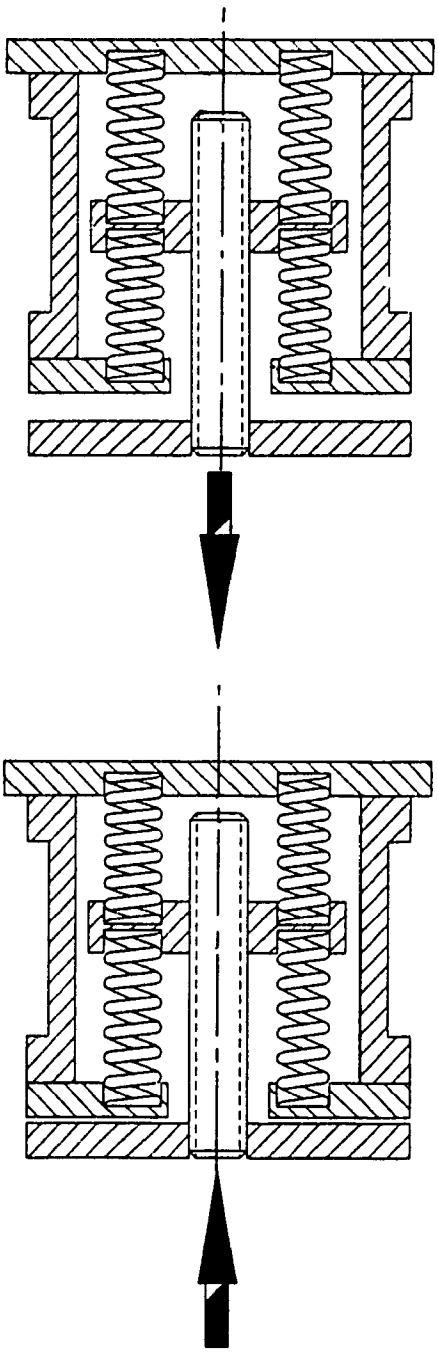


Fig. C8 PCD subjected to tensile and compressive loads

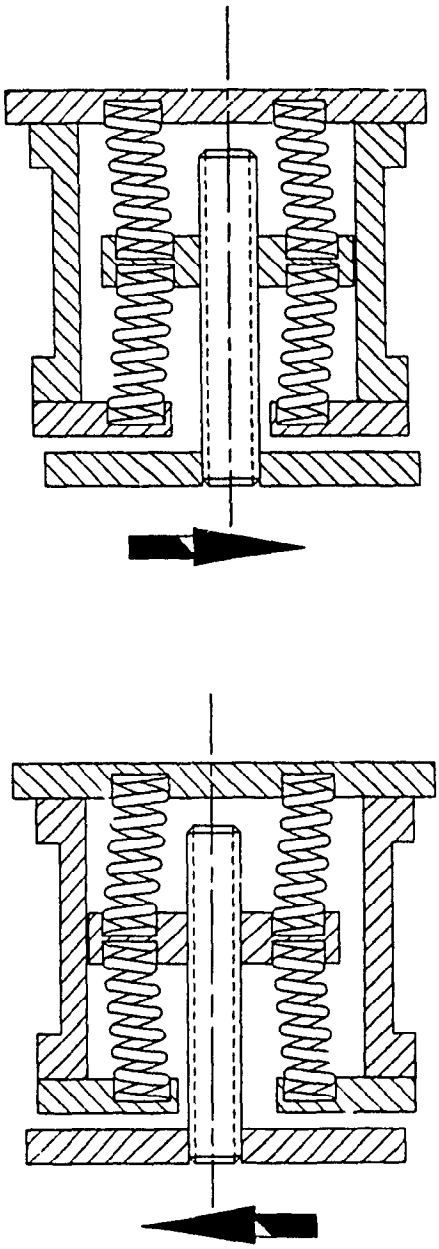


Fig C9 PCD subjected to lateral loads

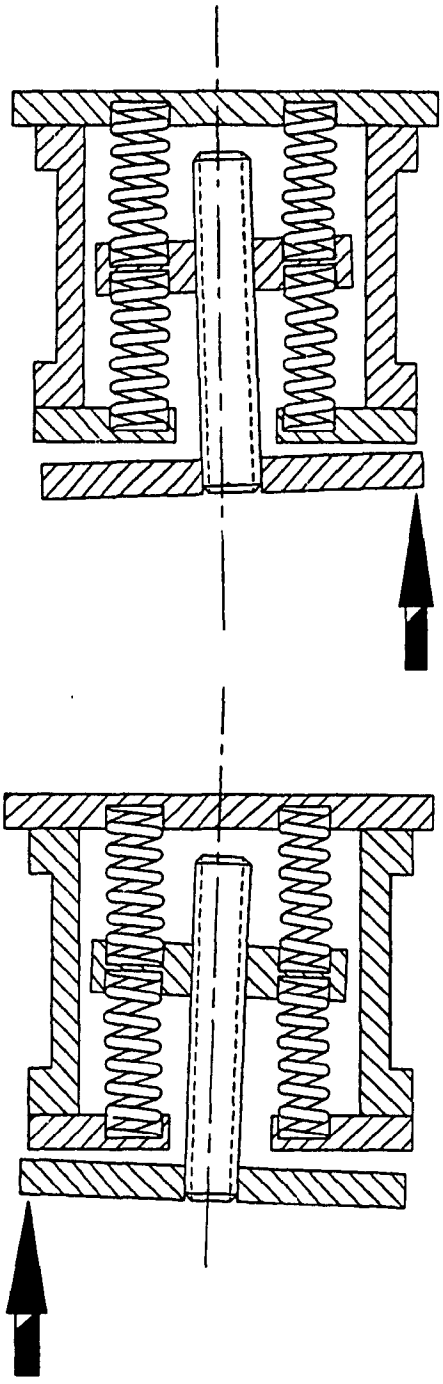


Fig. C10 PCD subjected to cocking loads

Appendix D

Sensor and PCD Specifications

D1.0 THE NEW SENSOR (Version 1)

Dimensions	Values	Units
Radius (R)	41	mm
Height (H)	99.4	mm
Offset (D)	7.5	mm
Length (L)	103	mm
Surface area (A)	11.52	mm^2
Rod Angle (θ)	74.8	degrees

Table I ... Dimensions of the Sensor (Version 1) [Croteau `95]

Characteritics	Axes		Value (Newtons)
Force measurable maximum for a (± 555 μ strain) per element limited by circuit (at centre of the sensor)	Elements		452
	X		340
	Y		320
	Z		2,620
Absolute uncertainity (for fine changes at 12 bits stabilised and ± 555 μ strain)	Elements	ΔF	0.22
	X	ΔFx	0.22
	Y	ΔFy	0.19
	Z	ΔFz	1.3
	X	ΔMx	0.030 N-m
	Y	ΔMy	0.032 N-m
	Z	ΔMz	0.012 N-m

Table II .... Specifications of the Sensor (Version 1) [Croteau `95]

D1.1 THE NEW SENSOR (Version 2)

Dimensions	Values	Units
Radius (R)	38	mm
Height (H)	34 . 6	mm
Offset (D)	9	mm
Length (L)	40	mm
Surface area (A)	6 . 3	mm^2
Rod Angle (θ)	60	degrees

Table IA ... Dimensions of the Sensor (Version 2)

Characteritics	Axes	Value (Newtons)
Force measurable maximum for a (± 555 μ strain) per element limited by circuit (at centre of the sensor)	Elements	247
	X	375
	Y	365
	Z	1,280
Maximum force for a (± 2,000 μ strain) per element for maximum resistance value of element (at centre of sensor)	Element	894
	X	1,400
	Y	1,400
	Z	4,700
Absolute uncertainity (for fine changes at 12 bits stabilised and ± 555 μ strain)	Elements ΔF	0.12
	X ΔFx	0.24
	Y ΔFy	0.21
	Z ΔFz	0.63
Force maximum in each element for worst case	F element	105
Critical charge for buckling	Factor of safety	190

Table IIA .... Specifications of the Sensor (Version 2)

Table III **COMPARISON OF THE DEVELOPED SENSOR (Version 2) with THE ATI Model 75 / 300 Sensor**

Parameter	Units	Developed Sensor	ATI Model 75 / 300
Fx	N	370	334
Fy	N	360	334
Fz	N	1280	334
Mx	N-m	21	33.9
My	N-m	24	33.9
Mz	N-m	24	33.9
ΔFx	N	0.24	0.28
ΔFy	N	0.21	0.28
ΔFz	N	0.63	0.56
ΔMx	N-m	0.014	0.014
ΔMy	N-m	0.014	0.014
ΔMz	N-m	0.012	0.014

Dimensions	Values	Units
Spring Diameter	3 / 8	inches
Spring Free Length	1.0	inches
Load at 50 percent deflection	28	lb
Height of the PCD (H)	2.80	inches
Diameter of the PCD (D)	2.75	inches
Weight of the PCD (W)	0.45	lb
Load Capacity :		
Tension	56	lb
Compression	84	lb
Lateral	4	lb
Cocking	18	lb-in
Stiffness :		
Lateral	42	lb / in
Cocking	1280	lb - in / rad
Misalignment Range :		
Lateral	0.1	in
Cocking	2.0	degrees
Torsion	5.0	degrees

Table IV ... Specifications of the Passive Compliance Device

Table V COMPARISON OF THE DEVELOPED PASSIVE COMPLIANCE DEVICE WITH ATI series 100 Device

Dimensions	Developed PCD	ATI Series 100	Units
Height of the PCD (H)	2.80	2.20	inches
Diameter of the PCD (D)	2.75	3.15	inches
Weight of the PCD (W)	0.45	0.50	lb
Load Capacity :			
Tension	56	18	lb
Compression	84	280	lb
Lateral	4	6	lb
Cocking	18	60	lb-in
Stiffness :			
Lateral	42	150	lb / in
Cocking	1280	4000	lb - in / rad
Misalignment Range :			
Lateral	0.1	0.085	in
Cocking	2.0	1.10	degrees
Torsion	5.0	5.0	degrees

Appendix E

Testing Of Sensor and PCD

After designing and manufacturing the hardware, testing was carried out to check the performance. The test results were a litmus test to compare the design with actual performance.

E1 TEST RIG

The test rig is shown in the schematic in Fig. E1. A hydraulic cylinder with a Linear Variable Differential Transducer was used to apply tensile and compressive forces. This served two purposes, one was to apply the load and the other advantage using the LVDT was it gave the precise distance the cylinder had moved during load application. This was translated into the spring deflection that had occurred. Using the LVDT gave accurate values of the spring compression and elongation.

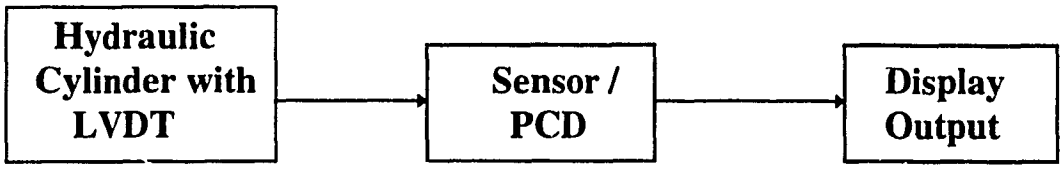


Fig. E1 Schematic of the Test set-up for Sensor & PCD

The sensor / PCD were held in proper fixtures and the loads were applied. The tests were also done manually for each element of the force sensor, as these elements were difficult to mount in the fixtures.

E2 SENSOR TESTING

For version 1 of the sensor, load range was 0 to 25 kilograms and for version 2, the load range was 0-20 kilograms, corresponding voltage output was measured and is tabulated below. The elements were held in a pin at one end of the bearing and loads were added at the other end, to check the corresponding voltage that was being obtained in each case, for any changes in load.

In the power testing of the sensor, mainly loads were applied to the assembled sensor and the corresponding voltages measured to check the output behaviour. When the sensor was assembled, each element in the sensor was under a different state and the voltage sign was positive for the elements that were in tension and negative for those in compression. The gains of each element amplifier were also adjusted to zero, before beginning the loading process.

From the values in the tables the curves were plotted using Matlab, the best fit-curve was used to get the trend of the element behaviour. The relationship in each case is pretty linear. At certain points some non-linearity is observed, this could be attributed to the un-

settling of amplifier, noise in the system and human errors. Overall the behaviour was linear for the elements.

Force Applied (in Kilograms)	Voltage output (in Volts) ELEMENT # 1	Voltage output (inVolts) ELEMENT # 2
1.261	0.0532	0.0554
3.524	0.1499	0.1510
5.904	0.2542	0.2477
8.098	0.3558	0.3418
10.406	0.4514	0.4359
12.669	0.5551	0.5371
13.529	0.5865	0.5734
15.792	0.6870	0.6698
18.027	0.7861	0.7649
20.436	0.8941	0.8672
22.671	0.9922	0.9630
24.971	1.0920	1.0624

Table VI Calibration Values for Elements 1 and 2 (Version 1)

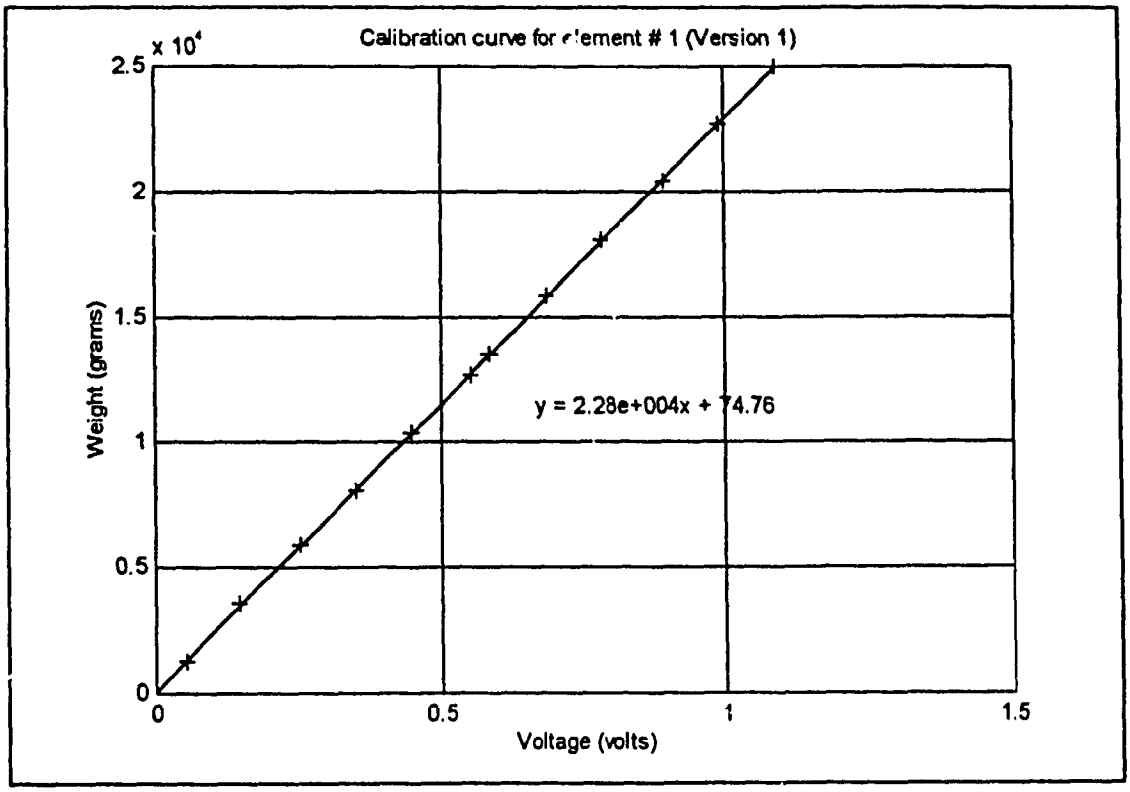


Fig.E2 Calibration curve for element #1 (Version 1)

Force Applied (in Kilograms)	Voltage output (in Volts) ELEMENT # 1	Voltage output (inVolts) ELEMENT # 2
1.147	0.1169	0.1133
2.300	0.2416	0.2423
3.447	0.3699	0.3723
4.574	0.4716	0.4874
5.720	0.5963	0.5959
6.874	0.7312	0.6953
8.020	0.8471	0.7984
8.958	0.9410	0.8735
11.258	1.1828	1.0741
15.902	1.4285	1.2867
18.150	1.6630	1.5033

Table VII Calibration Values for Elements 1 and 2 (Version 2)

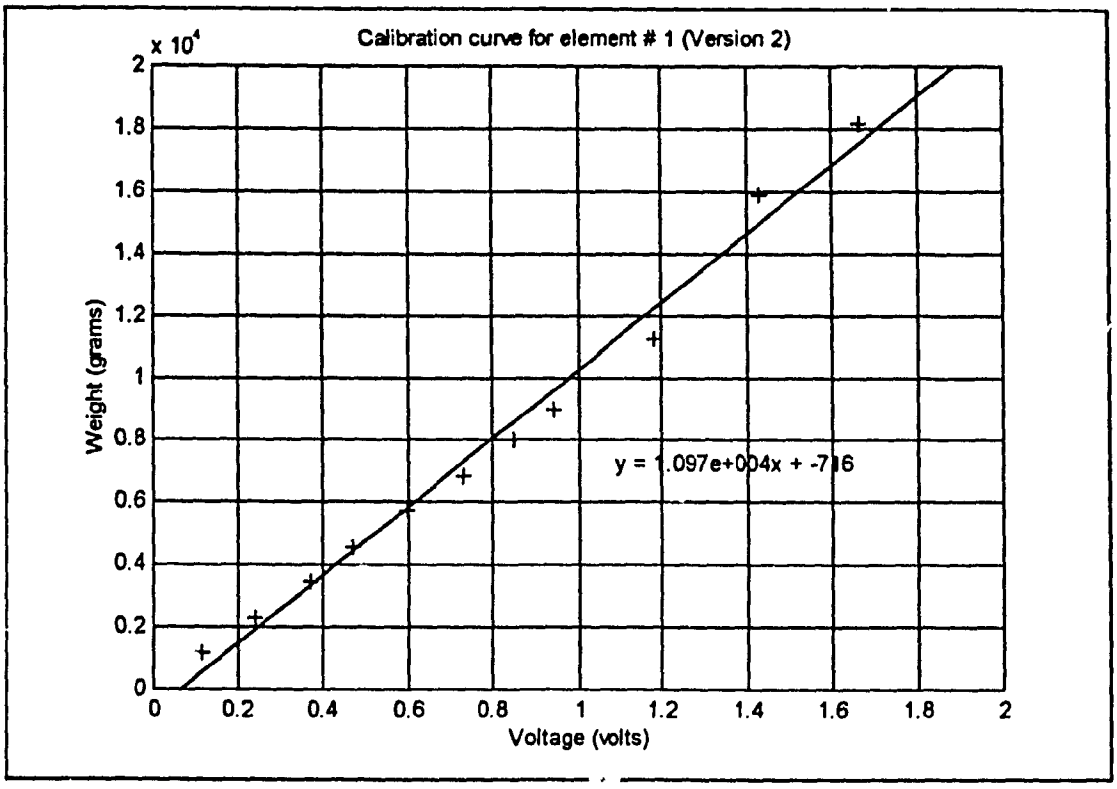


Fig.E3 Calibration curve for element #1 (Version 2)

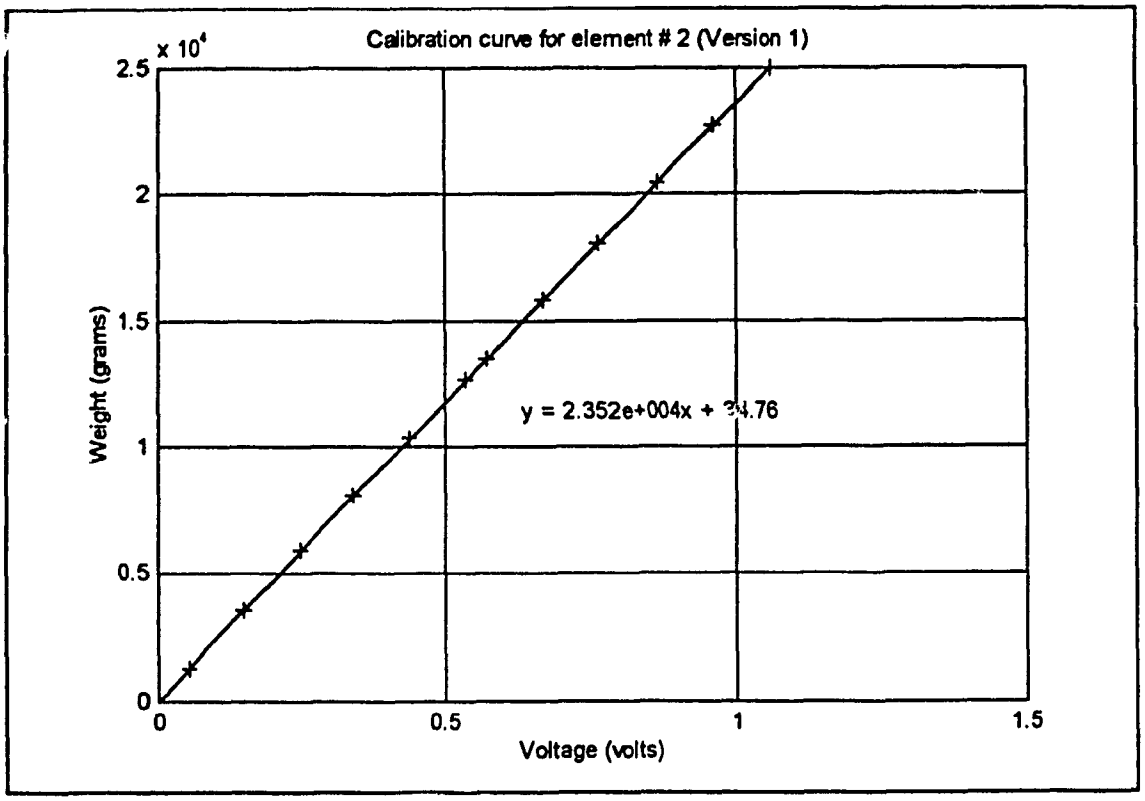


Fig.E4 Calibration curve for element #2 (Version 1)

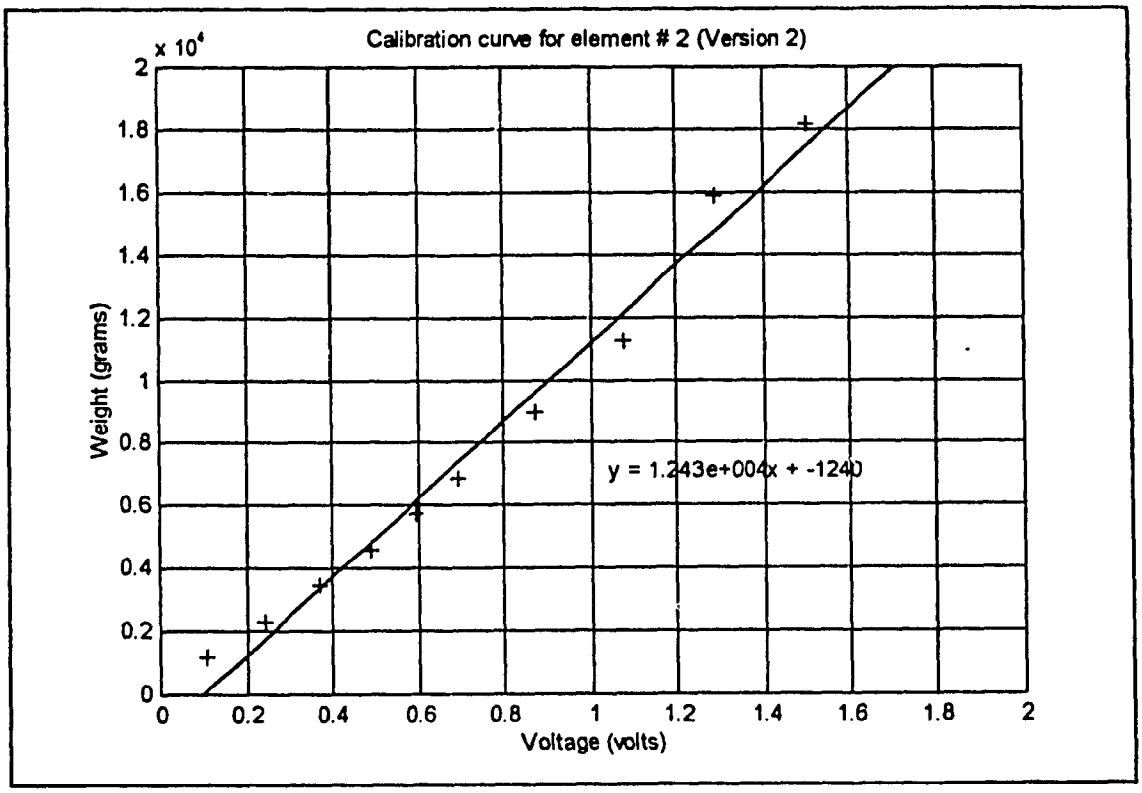


Fig.E5 Calibration curve for element #2 (Version 2)

Force Applied (in Kilograms)	Voltage output (in Volts) ELEMENT # 3	Voltage output (inVolts) ELEMENT # 4
1.261	0.0598	0.0593
3.524	0.1624	0.1666
5.904	0.2673	0.2734
8.098	0.3664	0.3802
10.406	0.4684	0.4890
12.669	0.5709	0.5956
13.529	0.6067	0.6360
15.792	0.7098	0.7422
18.027	0.8172	0.8503
20.436	0.9241	0.9601
22.671	1.0270	1.0649
24.971	1.1339	1.1722

Table VIII Calibration Values for Elements 3 and 4 (Version 1)

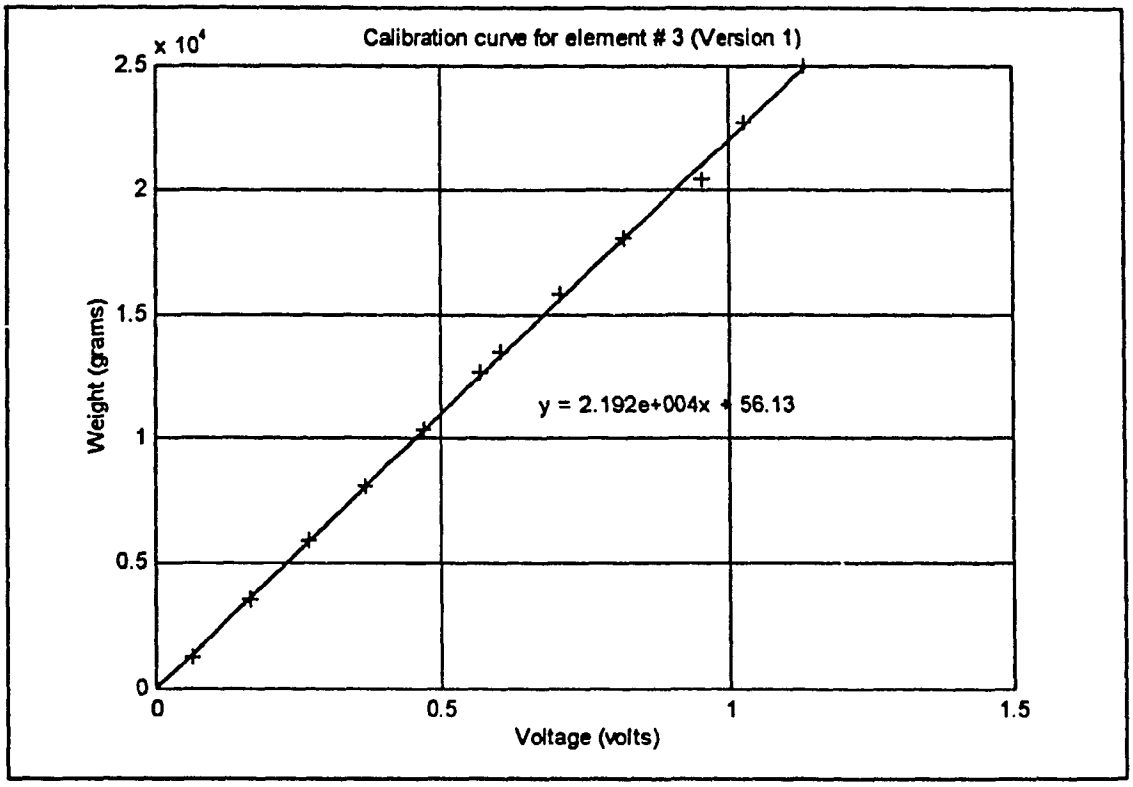


Fig. E6 Calibration curve for element #3 (Version 1)

Force Applied (in Kilograms)	Voltage output (in Volts) ELEMENT # 3	Voltage output (in Volts) ELEMENT # 4
1.147	0.1165	0.1012
2.300	0.2277	0.2048
3.447	0.3188	0.2985
4.574	0.4347	0.4520
5.720	0.5287	0.5148
6.874	0.6422	0.6499
8.020	0.7633	0.7881
8.958	0.8372	0.9067
11.258	0.9998	1.1375
15.902	1.2420	1.3010
18.150	1.4317	1.4715

Table IX Calibration Values for Elements 3 and 4 (Version 2)

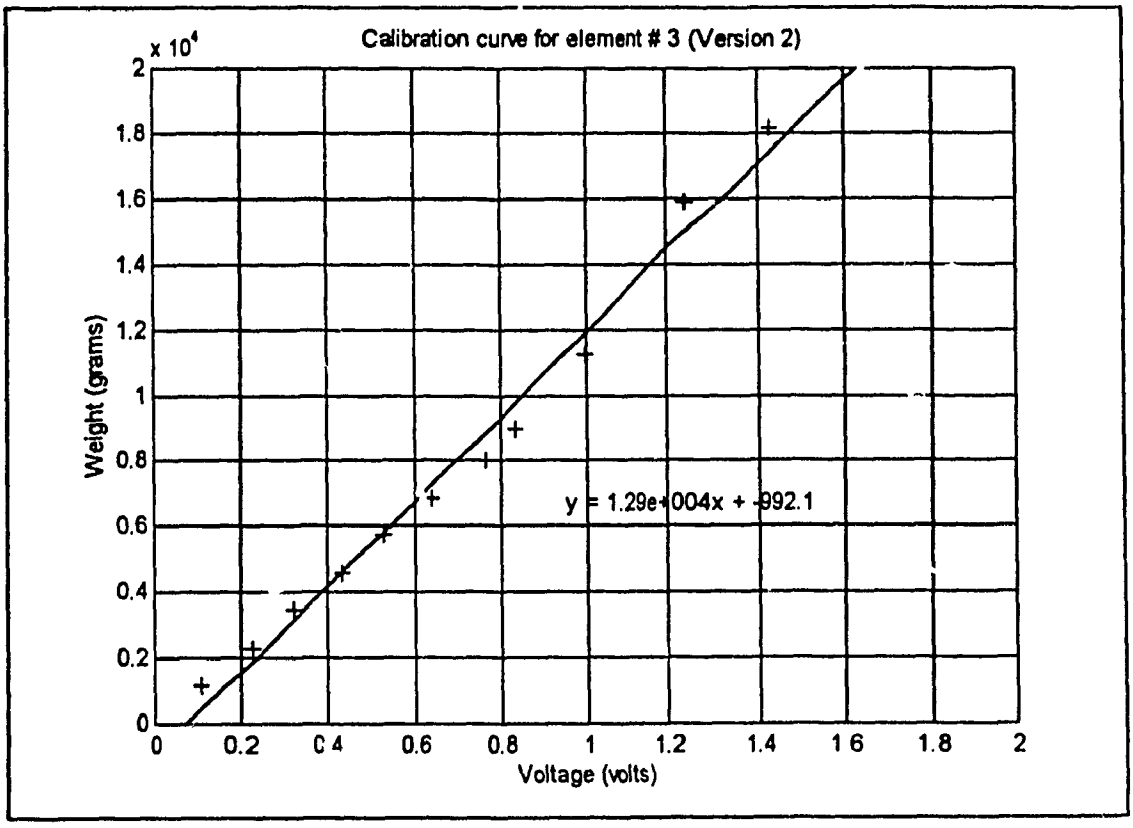


Fig.E7 Calibration curve for element #3 (Version 2)

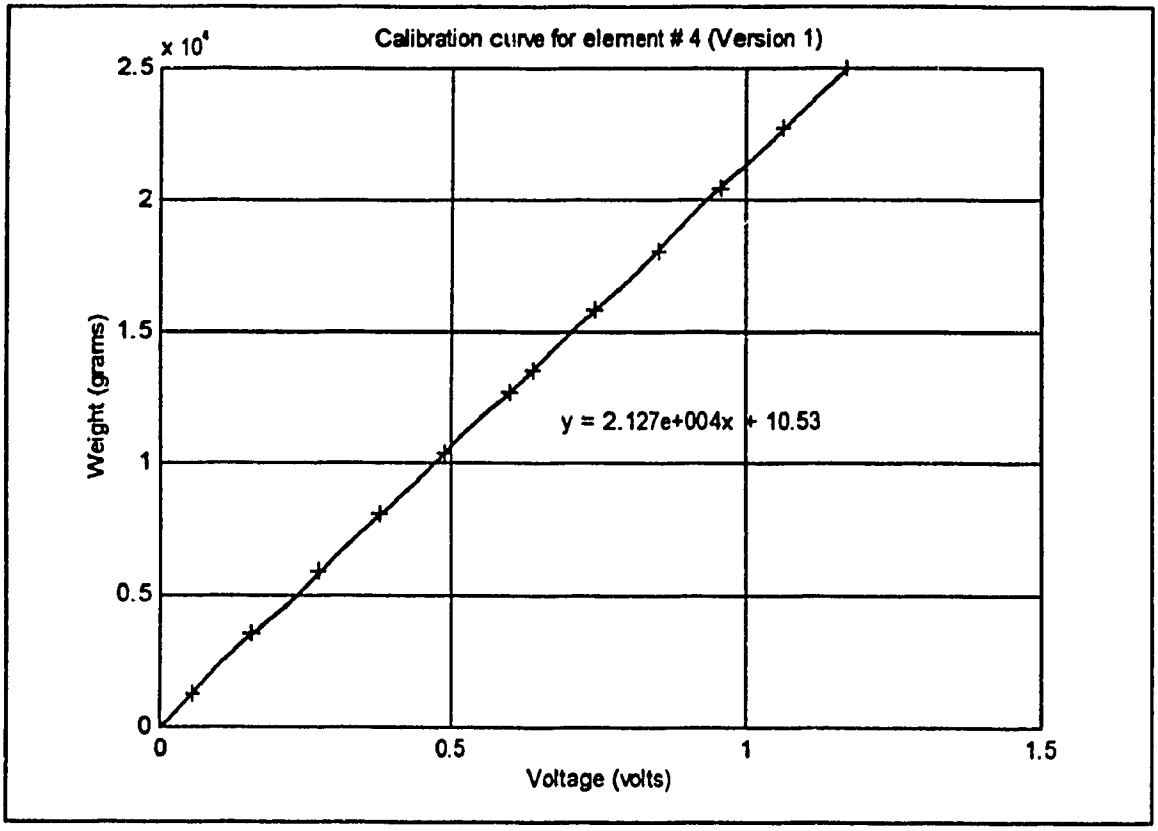


Fig.E8 Calibration curve for element #4 (Version 1)

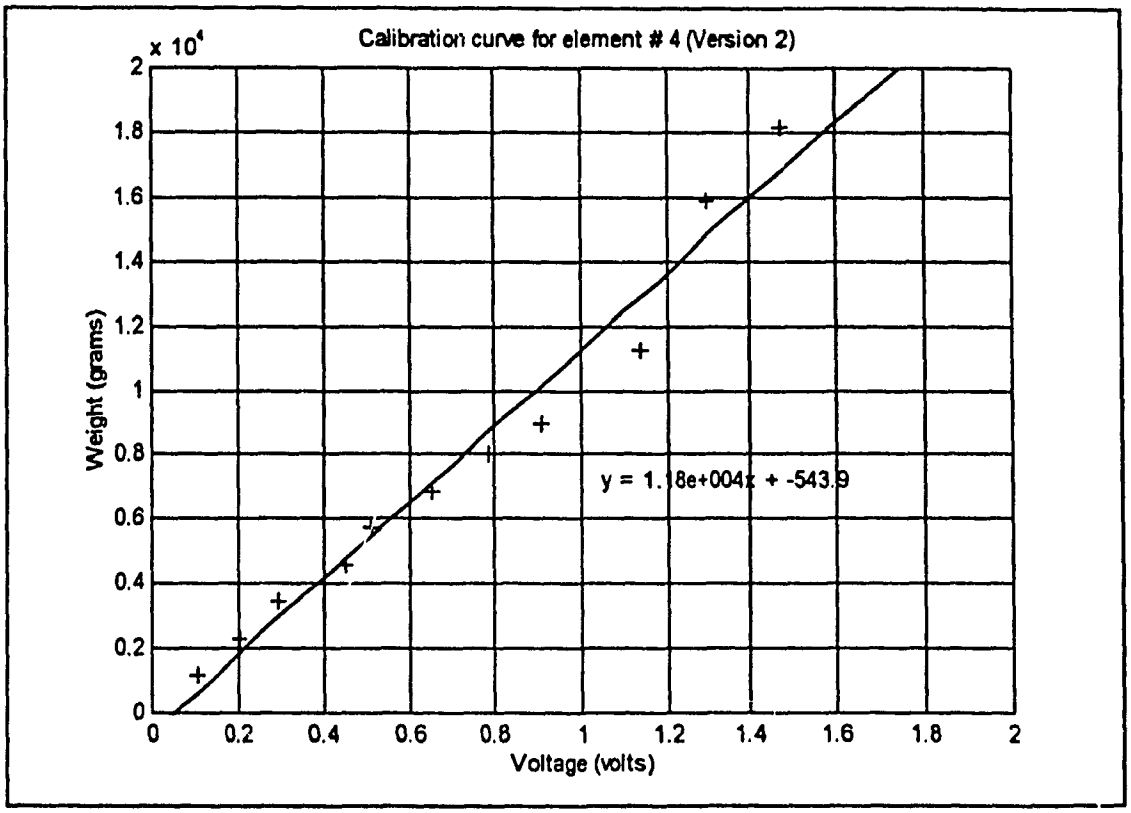


Fig.E9 Calibrartion curve for element #4 (Version 2)

Force Applied (in Kilograms)	Voltage output (in Volts) ELEMENT # 5	Voltage output (inVolts) ELEMENT # 6
1.261	0.0603	0.0578
3.524	0.1714	0.1575
5.904	0.2863	0.2567
8.098	0.4002	0.3552
10.406	0.5166	0.4566
12.669	0.6304	0.5594
13.529	0.6723	0.5956
15.792	0.7873	0.6994
18.027	0.8967	0.7926
20.436	1.0179	0.8997
22.671	1.1330	0.9856
24.971	1.2463	1.01918

Table X Calibration Values for Elements 5 and 6 (Version 1)

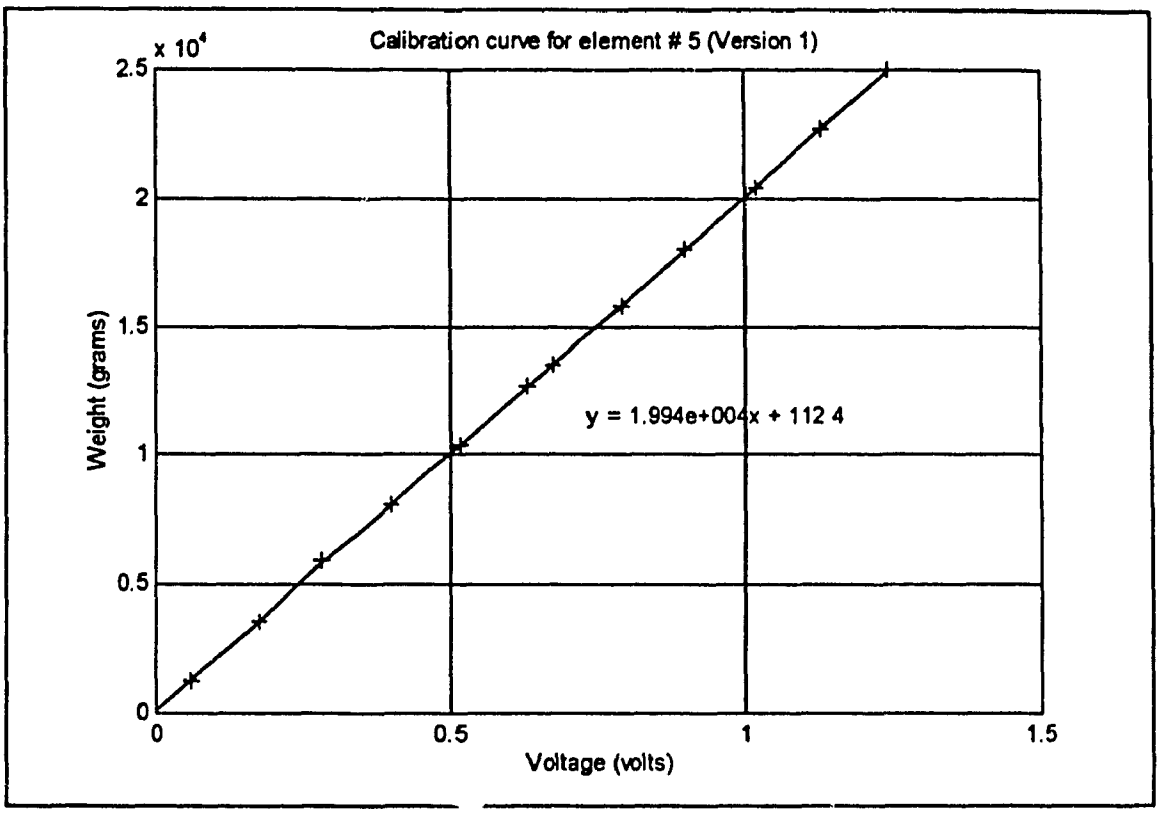


Fig.E10 Calibration curve for element #5 (Version 1)

Force Applied (in Kilograms)	Voltage output (in Volts) ELEMENT # 5	Voltage output (in Volts) ELEMENT # 6
1.147	0.1208	0.1189
2.300	0.2332	0.2438
3.447	0.3271	0.3899
4.574	0.4486	0.4590
5.720	0.6226	0.6061
6.874	0.7703	0.7068
8.020	0.8496	0.8051
8.958	1.0083	0.9094
11.258	1.2915	1.1420
15.902	1.5771	1.4400
18.150	1.7480	1.63333

Table XI Calibration Values for Elements 5 and 6 (Version 2)

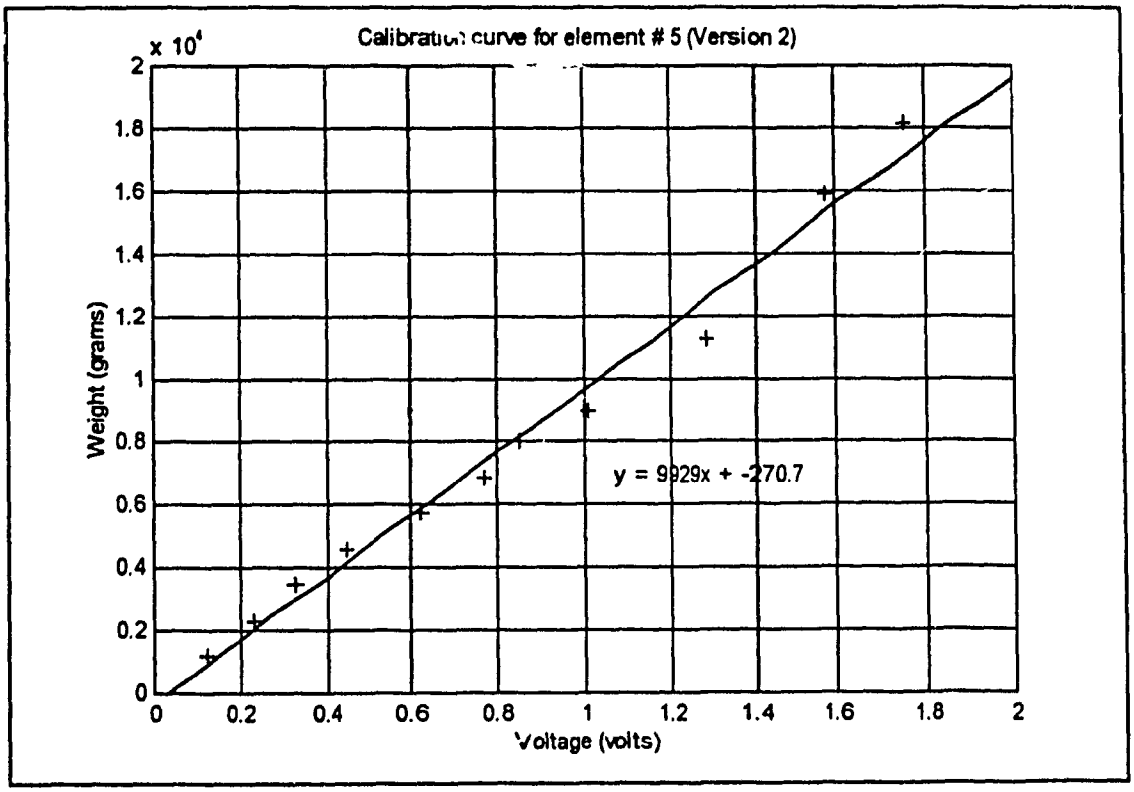


Fig.E11 Calibration curve for element #5 (Version 2)

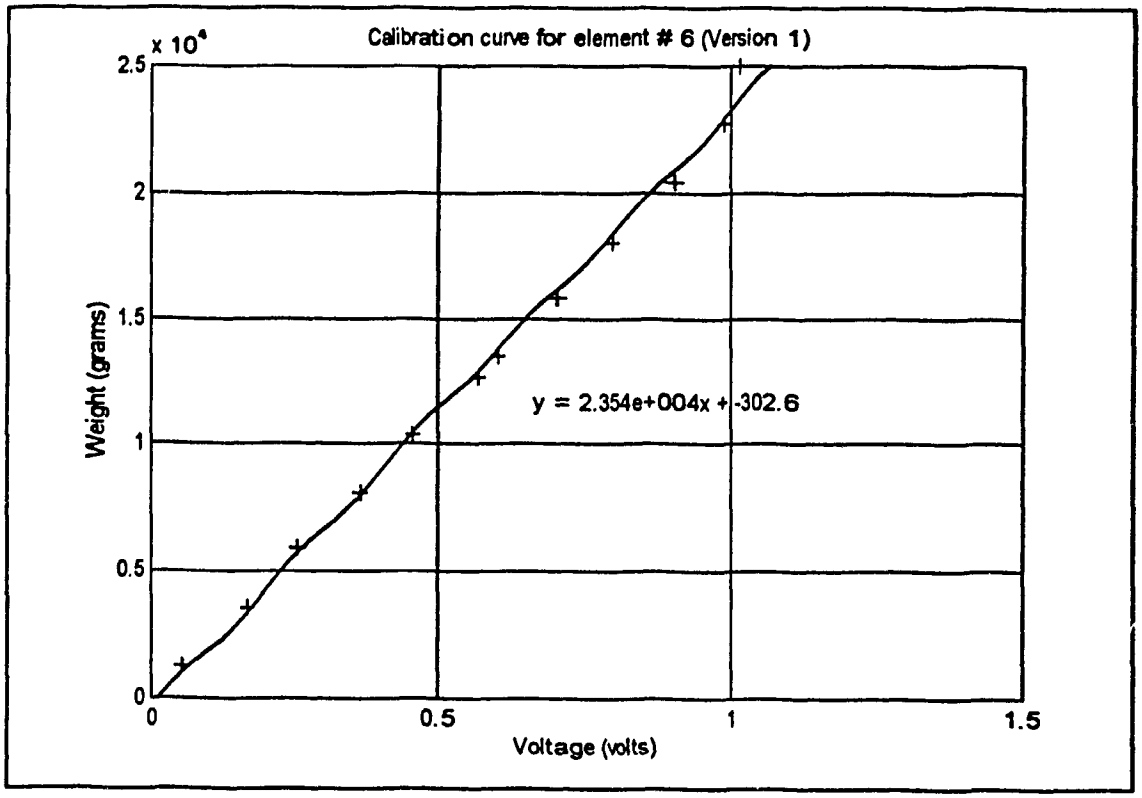


Fig.E12 Calibration curve for element #6 (Version 1)

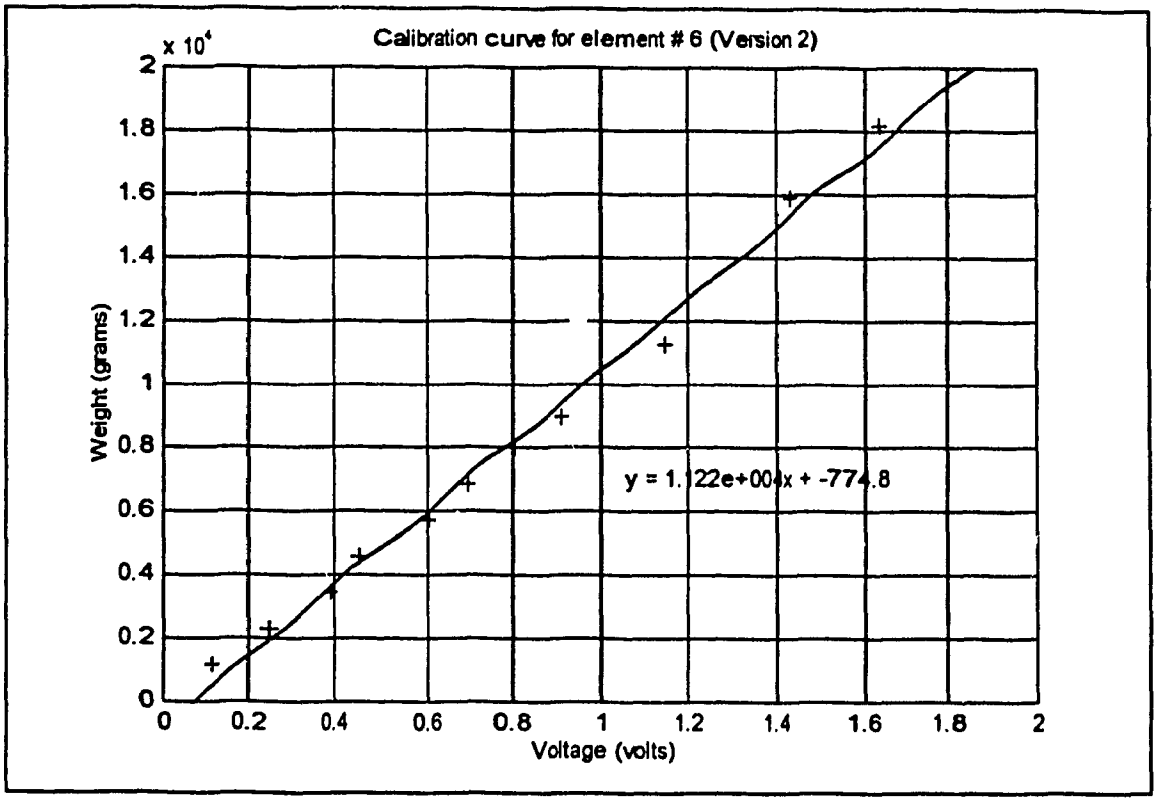


Fig.E13 Calibration curve for element #6 (Version 2)

E3 PCD TESTING

The PCD was tested in a similar manner to that of the sensor. The PCD was set-up in the fixture and power testing was carried out. Load was applied in steps, using the hydraulic cylinder and the corresponding compression was measured, from the LVDT readings. The same procedure was adopted for the tensile testing as well. Results were studied and the following features were of significance.

Load at 0. 10 inches (compression)	=	8 lbs
Load at 50 percent deflection (compression)	=	44 lbs
Load at 0.10 inches (tension)	=	5.4 lbs
Load at 50 percent deflection (tension)	=	26 lbs

At the end of these tests, the sensor and the PCD were mounted on the robot and various tests for the assembly algorithm were conducted, which have been discussed in Chapter 7.

Appendix F

Test Set-up Hardware Details

F1 THE ROBOT

A SCARA robot manufactured by Adept Technologies Inc., was used for this research. This robot has four degrees of freedom, with 3 rotations of the three main axes and in addition to translation in the third axis. This robot, Fig. F1, has a very high accuracy and is used for assembly operations in various industries. This robot does not have the tilt angle capability in the end effector, which turned out to be an advantage in performing this work, as non-linearities in one axes of the robot were absent.

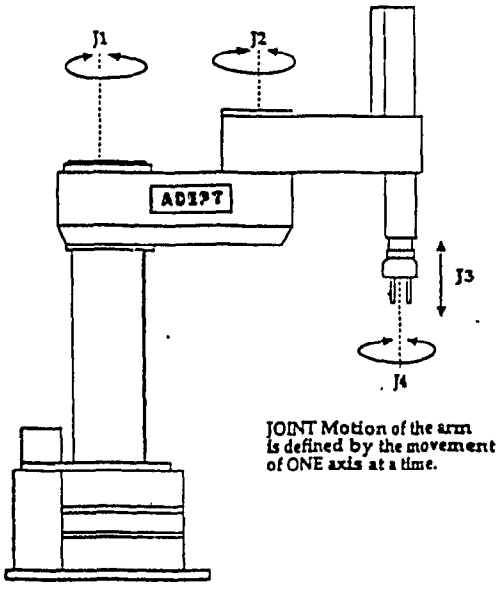


Fig. F1 Schematic of the Adept Robot [Adept Inc.1988]

The specifications of the robot are given below.

Specification	Value with units
Pay Load	20 lb
Downward force	80 lb
Joint 4 - nominal inertia	96 lb-in^2
Joint 4 - maximum inertia	1000 lb-in^2
Resolution (X, Y)	0.00036 in
Repeatability (X, Y)	0.001 in
Accuracy	± 0.003 in
Joint 1 rotation	250 degrees
Joint 2 rotation	104 degrees
Joint 3 stroke	8.0 in
Joint 4 rotation	554 degrees
Maximum reach	21.5 in
Minimum reach	9.8 in
Robot self weight	260 lb
Environmental Limitations	5 to 50 degrees C

Table XII Specifications of Robot [Adept Inc.1988]

F2 ROBOT CONTROLLER

The standard Adept CC Controller, Fig. F2 was used. For experimentation purposes, the controller was used as a path to send commands and receive them. The actual processing was carried out on the computer. The Specification of the controller are given in Table. XIII.

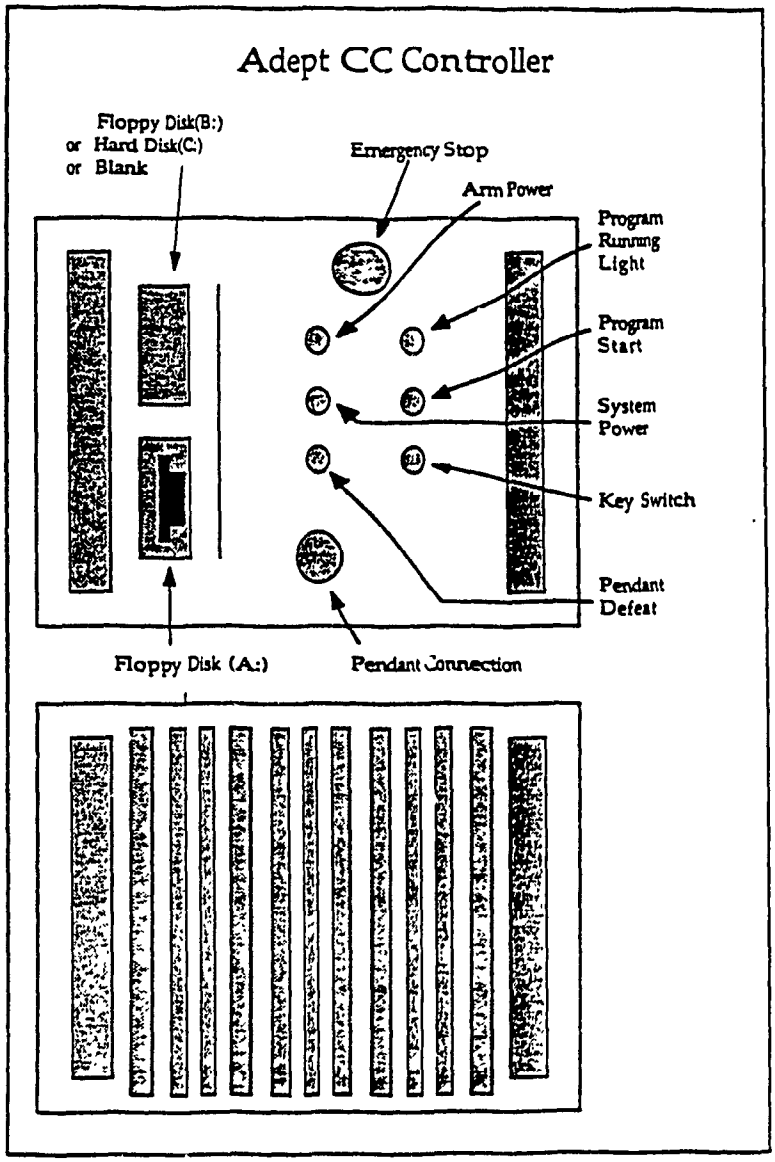


Fig. F2 Schematic of Adept CC Controller [Adept Inc. 1988]

Parameter	Specifications
Programming language (OS)	Standard : Adept V Optional : Adept V+
Microprocessors	Standard : Three Motorola 68000 (10 MHz) Optional : Companion 68020 (17 MHz)
Random Access Memory	1024 Kbytes
Mass storage medium	One 3.5 inches disk drive
Communication standards	Standard : Binary I/O; 16 channels serial; six RS 232 ports Optional : Manual control pendant
Mounting Standard	Two 19 inches rack mountable units
Weight	Control unit : 90 lbs Amplifier : 165 lbs
Electrical Requirements	208 / 220 / 240 VAC, 50 / 60 Hz, Single phase

Table XIII. Specifications of the Controller [Adept Inc.1988]

The Adept robot control system has been designed to allow control of the manipulator from either the optional programmer's terminal or the optional manual control pendant. The figure describes various control functions that this unit can perform. The 'soft 'buttons' are used during the execution of application programs. The 'Mode control buttons' alter the mode of operation of the robot. The 'USER LED is lit when an application program is accessing the pendant. Using the 'Function' buttons, any program

can be selected in a particular mode for priming. After a program has been primed it can be executed using the 'RUN / HOLD' button.

F3 AMPLIFIER FOR SENSOR

The signal from the sensor is needed to be amplified and then processed for acquiring. An

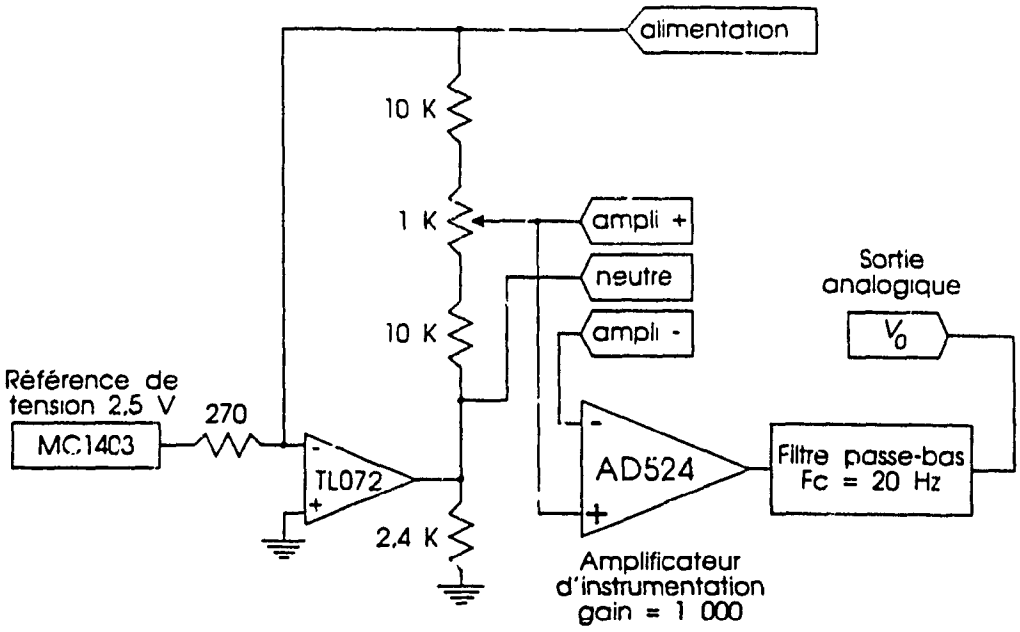


Fig. F3 Amplification circuit for one element [Croteau 1995]

amplifier with the following configuration was used. Detailed illustrations on selection criteria are available in the laboratory and do not fall in the scope of this work.

F4 INTERFACING COMPUTER & DATA ACQUISITION CARDS

A standard 486 system was used for processing information. The principal characteristics of the data acquisition card in brief are below.

i. - Eight Analog inputs for the micro controller MC68HC705C8.

ii - 16 bit Analog to Digital converter, with 16 micro-seconds conversion.

iii - Bi-polar signal for conversion

iv - Serial link RS 232 (maximum 115200 bauds)

v - Micro controller of Motorola, MC68HC705C8, three external ports

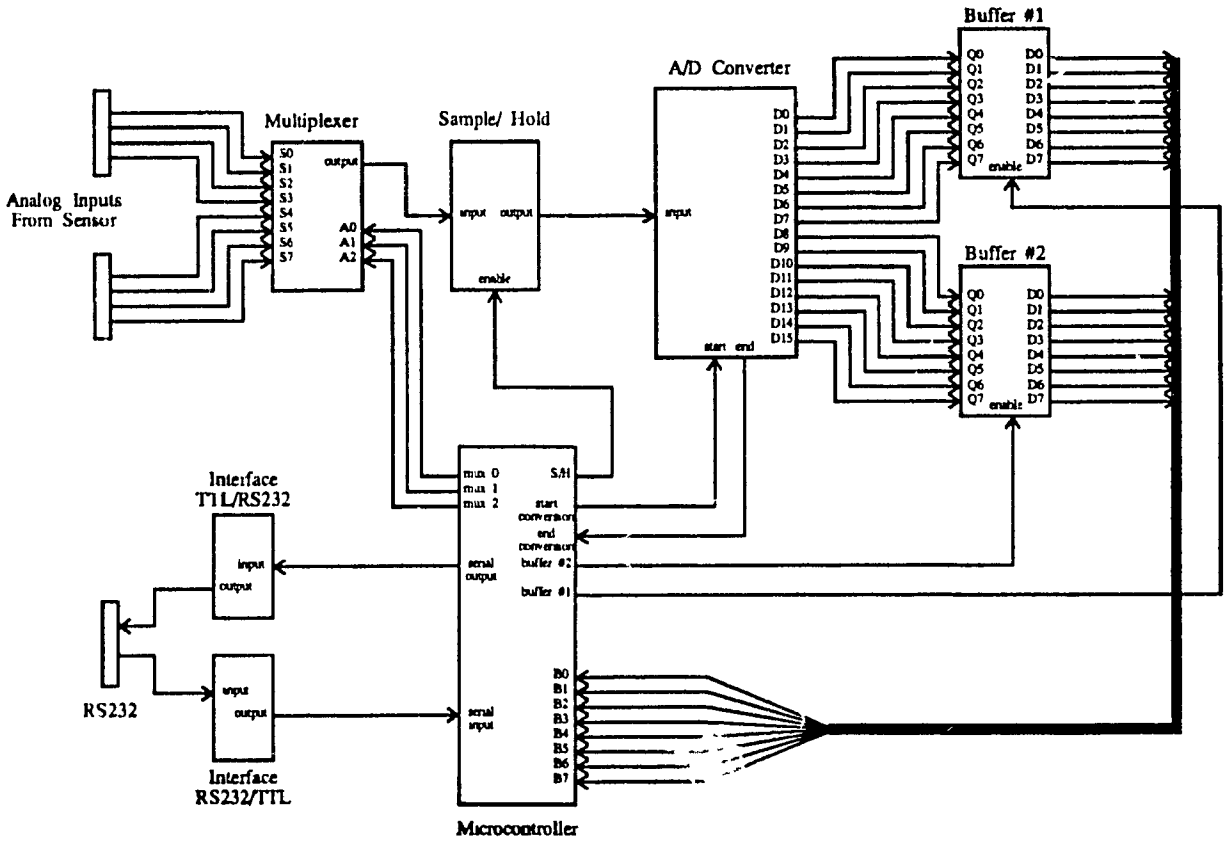


Fig. F4 Electronic circuit for Acquisition card [Richard 1995]

Appendix G

Codes for Simulation and Algorithm

G1 COMMUNICATION

```
// ********************** Début de ARUN.CPP **********************
//
//
#include <stdlib.h>
#include <stdio.h>
#include <conio.h>
#include <dos.h>
#include <math.h>
#include <ctype.h>
#include <string.h>
#include "crs232.h"
#include "pc8250.h"

extern void WritePort(char c);
extern char ReadPort();
extern void Move ( float X , float Y , float Z , float T ) ;
extern    void Where ( float *X , float *Y , float *Z , float *T ) ;
extern    void Close () ,
extern    void Open () ;


int main()
{

char c;

  for ( ; ; )
  {
    if ( kbhit() )
    {
      c = getch();
      if ( c == 27) return 0;
                          if ( c != 27) WritePort( c );
    }

    if ( ( c = ReadPort() ) > 0 )  cout << hex << (char) c;
    cout.flush(); // Microsoft
  }

  return 0;
}
// ********************** FIN DE ARUN.CPP **********************
```

G2 SIMULATION ROUTINES

```
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
#include <time.h>
#include <dos.h>
#include <graphics.h>

/*# define divide(a, b) = a / b;*/

/*# define Fxx  = (delFx / delx)  ratio of incremental Force in x */
                                    /*to an incremental change in x */
/*# define Fthx = (delFth / delx) /* ratio of incremental Force in theta */
                                    /* to an incremental change in x */
/*#define Fzz  = (delFz / delz)      ratio of incremental Force in z */
                                    /* to an incremental change in z */
# define MOVE_OK  1                 /* both gears are on same plane */
# define MOVE_FURTHER  0            /* both gears not in same plane */
# define C 50                       /* center distance of both gears */
# define GEARS_MESHED  1            /* both gears meshed */
# define GEARS_NOT_MESHED  0        /* gears stuck */

void step1 ();
int step2 ();
void draw( float x, float theta );

void main (void)

{
int driver, mode, i;
int alfa, count[23];

        for(i=0;i<23;i++) count[i]=0;
/*
        driver = DETECT;
        mode = 0;
        initgraph( &driver, &mode, "");
*/
        for (i=0; i<100; i++) {
                sleep(2);
                step1 ();
//      cleardevice();
                alfa = step2 ();
                if (alfa<23) count[alfa]++;
        }

        for(i=0;i<20;i++)
                printf("theta=%d  count= %d  \n", i, count[i]);

        getch();
//      restorecrtmode();
}
```

```
void step1 ()
{

        int flag;
        float z;
        int delz, Fzz;

        for ( z= 0.0; z <= 9.5; z+= 0.5)
        printf ("z = %f\n", z);

        flag = MOVE_FURTHER;
        Fzz = 0;
        z = 9.5;
        while (!flag)                   {
                z += 0.1;
                Fzz++;
                if (Fzz <= 5) {
                printf ("z = %f   Fzz = %d\n", z, Fzz);
                }
                if ((Fzz >= 5) || (z == 10)) {
                        flag = MOVE_OK;
                        printf ("Fzz = %d\nMOVE_OK \n", Fzz);
                }
        }
}

int step2 ()

{
        int flag;
        float x, theta, L1, L2;
        int Fxx, Fxth, statum;

        randomize();
        L1 = 0.05*random(60) - 1.5 + 30.0;// distance between gears
        L2 = 6.50;                                      // max movement for meshing

        statum = rand_n(100);
        if (statum > 5) statum = 1; // 1 - not mesh
        else statum = 0;                // 0 - meshed

        x=C+L1+L2;
        flag = GEARS_NOT_MESHED;
        Fxx = 0;
        theta = 0.0;

        x -= 0.1;
        while (!flag)                   {
//              draw( x, theta );
//              getch();
//              sleep( 1 );


                if (x > C+L2 && !Fxx) {
```

```
                        x-= 0.1;
//                      printf("Move Closer; x= %5.2f\n", x);
                        continue;
                }
                else {
                        if (statum) {
                                Fxx++;
//                              printf("Contact Made! Fxx= %d\n", Fxx);
                        }
                }
                if (Fxx) {

                        do { x+= 0.1;
                        } while ( x <= C+L2+5);
                        Fxx = 0;

                        theta += 1.0;
                        statum = random(100);
                        if (statum > 30) statum = 1;            // 1 - not mesh
                        else statum = 0;                // 0 - meshed
                        continue;
                }
                else x-= 0.1;
                if (x <= C) {
                        flag= GEARS_MESHED;
//                      printf("Mesh OK!");
                }
                else {
                        flag= GEARS_NOT_MESHED;
  //                    printf("Try Again; x = %5.2f, Fxx = %2d, theta= %5.2f\n ", x, Fxx, theta);
                }
        }
        return theta;
}

void draw( float x, float theta )
{
        cleardevice();
        circle( 450, 200, 60 );
        circle( 450 - 2*x, 200, 40 );
        line(450 - 2*x, 200, 450 - 2*x +40, 200+ theta*2);
}


//*****************************************************************************//
//*****************************************************************************//



/********************************* DIFFERENT PROBS*****************************/
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
```

```
#include <time.h>
#include <dos.h>
#include <conio.h>
#include <graphics.h>

/*# define divide(a, b) = a / b;*/

/*# define Fxx  = (delFx / delx)  ratio of incremental Force in x */
                                    /*to an incremental change in x */
/*# define Fthx = (delFth / delx) /* ratio of incremental Force in theta */
                                   /* to an incremental change in x */
/*#define Fzz  = (delFz / delz)      ratio of incremental Force in z */
                                   /* to an incremental change in z */
# define MOVE_OK  1                /* both gears are on same plane */
# define MOVE_FURTHER  0           /* both gears not in same plane */
# define C 80                      /* center distance of both gears */
# define Rg 50                     /* radius of driving gear */
# define Rp 30                     /* radius of driven gear */
# define GEARS_MESHED  1           /* both gears meshed */
# define GEARS_NOT_MESHED  0       /* gears stuck */

void step1 ();
void step2 ();
void draw( float x, float theta );

void main (void)

{
int driver, mode;

        driver = DETECT;
        mode = 0;
        initgraph( &driver, &mode, "c:\\bc\\bgi");

        step1 ();
        cleardevice();
        step2 ();
        getch();
        restorecrtmode();
}

void step1 ()
{

        int flag;
        float z;
        int delz, Fzz;

        for ( z= 0.0; z <= 9.5; z+= 0.5)
        printf ("z = %f\n", z);

        flag = MOVE_FURTHER;
        Fzz = 0;
        z = 9.5;
```

```
        while (!flag)                   {
                z += 0.1;
                Fzz++;
                if (Fzz <= 5) {
                printf ("z = %f   Fzz = %d\n", z, Fzz),
                }
                if ((Fzz >= 5) || (z == 10)) {
                        flag = MOVE_OK,
                        printf ("Fzz = %d\nMOVE_OK \n", Fzz);
                }
        }
}

void step2 ()

{
        int flag;
        float x, theta, L1, L2,
        int Fxx, Fxth, statum;

        randomize();
        L1 = 0.05*random(60) - 1.5 + 30.0;// distance between gears
        L2 = 6.50;                                      // max movement for meshing

        statum = random(100);
        if (statum > 5) statum = 1; // 1 - not mesh
        else statum = 0;                // 0 - meshed

        x=C+L1+L2;
        flag = GEARS_NOT_MESHED;
        Fxx = 0;
        theta = 0.0,

        x -= 0.1,
        while (!flag)                   {
                draw( x, theta );
//              getch(),
//              sleep( 1 );


                if (x > C+L2 && !Fxx) {
                        x-= 0.1;
//                      printf("Move Closer; x= %5.2f\n", x);
                        continue;
                }
                else {
                        if (statum) {
                                Fxx++;
//                              printf("Contact Made! Fxx= %d\n", Fxx);
                        }
                }
                if (Fxx) {

                        do { x+= 0.1;
```

```
                        } while ( x <= C+L2+5);
                        Fxx = 0;
                        theta += 1.0;
                        statum = random(100);
                        if (statum > 40) statum = 1;          // 1 - not mesh
                        else statum = 0;              // 0 - meshed
                        continue;
                }
                else x-= 0.1;
                if (x <= C) {
                        flag= GEARS_MESHED;
//                      printf("Mesh OK!");
                }
                else {
                        flag= GEARS_NOT_MESHED;
//                      printf("Try Again; x = %5.2f, Fxx = %2d, theta= %5.2f\n ", x, Fxx, theta),
                }
        }
}

void draw( float x, float theta )
{
        cleardevice();
        circle( 400, 200, Rg*2 );
        circle( 400 - 2*x, 200, Rp*2 );
        line(400 - 2*x, 200, 400 - 2*x +Rp*2, 200+ theta*2);
}



//*************************************************************************//
//*************************************************************************//



/*****************************uniform PROBABILITY ****************************/

#include <stdio.h>
#include <stdlib.h>
#include <math.h>
#include <time.h>
#include <dos.h>
#include <conio.h>
#include <graphics.h>

/*# define divide(a, b) = a / b;*/

/*# define Fxx  = (delFx / delx)  ratio of incremental Force in x */
                                  /*to an incremental change in x */
/*# define Fthx = (delFth / delx) /* ratio of incremental Force in theta */
                                  /* to an incremental change in x */
/*#define Fzz  = (delFz / delz)       ratio of incremental Force in z */
                                  /* to an incremental change in z */
```

```
# define MOVE_OK  1              /* both gears are on same plane */
# define MOVE_FURTHER  0         /* both gears not in same plane */
# define C 50                    /* center distance of both gears */
# define GEARS_MESHED  1         /* both gears meshed */
# define GEARS_NOT_MESHED  0     /* gears stuck */

void step1 ();
int step2 ();
void draw( float x, float theta );

void main (void)

{
int driver, mode, i;
int alfa, count[23];

        for(i=0;i<23;i++) count[i]=0;
/*
        driver = DETECT;
        mode = 0;
        initgraph( &driver, &mode, "");
*/
        randomize();
        for (i=0; i<1000; i++) {
                step1 ();
//      cleardevice();
                alfa = step2 ();
                if (alfa<23) count[alfa]++;
        }

        for(i=0;i<20;i++)
                printf("theta=%d  count= %d   \n", i, count[i]);

        getch();
//      restorecrtmode();
}

void step1 ()
{

        int flag;
        float z;
        int delz, Fzz;

        for ( z= 0.0; z <= 9.5; z+= 0.5)
        printf ("z = %f\n", z);

        flag = MOVE_FURTHER;
        Fzz = 0;
        z = 9.5;
        while (!flag)                   {
                z += 0.1;
                Fzz++;
                if (Fzz <= 5) {
```

```
                printf ("z = %f   Fzz = %d\n", z, Fzz);
                }
                if ((Fzz >= 5) || (z == 10)) {
                        flag = MOVE_OK;
                        printf ("Fzz = %d\nMOVE_OK \n", Fzz);
                }
        }
}

int step2 ()

{
        int flag;
        float x, theta, L1, L2;
        int Fxx, Fxth, statum;


        L1 = 0.05*random(60) - 1.5 + 30.0;// distance between gears
        L2 = 6.50;                                      // max movement for meshing

        statum = random(20);
//      if (statum > 50) statum = 1;            // 1 - not mesh
//      else statum = 0;                // 0 - meshed

        x=C+L1+L2;
        flag = GEARS_NOT_MESHED;
        Fxx = 0;
        theta = 0.0;

        x -= 0.1;
        while (!flag)                   {
//              draw( x, theta );
//              getch();
//              sleep( 1 );


                if (x > C+L2 && !Fxx) {
                        x-= 0.1;
//                      printf("Move Closer; x= %5.2f\n", x);
                        continue;
                }
                else {
                        if (statum) {
                                statum--;
                                Fxx++;
//                              printf("Contact Made! Fxx= %d\n", Fxx);
                        }
                }
                if (Fxx) {

                        do { x+= 0.1;
                        } while ( x <= C+L2+5);
                        Fxx = 0;
```

```
                        theta += 1.0;
//                      statum = random(100);
//                      if (statum > 30) statum = 1;          // 1 - not mesh
//                      else statum = 0;              // 0 - meshed
                        continue;
                }
                else x-= 0.1;
                if (x <= C) {
                        flag= GEARS_MESHED;
//                      printf("Mesh OK!");
                }
                else {
                        flag= GEARS_NOT_MESHED;
  //                    printf("Try Again; x = %5.2f, Fxx = %2d, theta= %5.2f\n ", x, Fxx, theta);
                }
        }
        return theta;
}

void draw( float x, float theta )
{
        cleardevice();
        circle( 450, 200, 60 );
        circle( 450 - 2*x, 200, 40 );
        line(450 - 2*x, 200, 450 - 2*x +40, 200+ theta*2);
}



/*****************************************************************************/
/*****************************************************************************/




/*-----------------------------------------------------------*/
/*                                                           */
/*      Display images                                       */
/*                                                           */
/*-----------------------------------------------------------*/
#include <graphics.h>       /* For graphics library functions */
#include <stdlib.h>         /* For exit() */
#include <stdio.h>
#include <malloc.h>
#include <conio.h>
#include <math.h>
#include <dos.h>


#define hole_x          250
#define hole_y  250
#define hole_rad 40
```

```
#define shaft_x              150
#define shaft_y              150
#define shaft_rad     40
#define key_width_ang     15
#define key_thick8
#define step_x        1
#define x_number       100
#define step_y        1
#define y_number       100


void draw(int c_x, int c_y, int rad, float width_ang, int dep, float theta,int color, int pattern,int bk);
void delay_time(int a);

int set_graph(void)
{
int graphdriver = DETECT, graphmode, error_code;


initgraph(&graphdriver, &graphmode, "");

error_code = graphresult();
if (error_code != grOk)
return(-1);
if ((graphdriver != VGA) && (graphdriver != EGA))
  {
  closegraph();
  return(0);
  }
return(1);
}

main()
{
int i,j,k,l;
float x1,y1,x2,y2,current_x,current_y;
float theta_shaft=40, theta_hole=0;
if (set_graph() != 1)
    {
    printf("This program requires VGA graphics\n");
    exit(0);
    }
setgraphmode(VGA);

draw(hole_x, hole_y, hole_rad, key_width_ang, key_thick, theta_hole,7,1,10);
draw(shaft_x, shaft_y, shaft_rad, key_width_ang, key_thick, theta_shaft,11, 1,4),

for(i=1;i<=x_number;i++)
          {
          clrscr();
          current_x=shaft_x+i*step_x;
          draw(hole_x, hole_y, hole_rad, key_width_ang, key_thick, theta_hole,7,1,10),
          draw(current_x, shaft_y, shaft_rad, key_width_ang, key_thick, theta_shaft,11,1,4),
          delay_time(2000);
```

```
        }
for(i=1;i<=y_number;i++)
        {
        clrscr();
        current_y=shaft_y+i*step_y;
        draw(hole_x, hole_y, hole_rad, key_width_ang, key_thick, theta_hole,7,1,10);
        draw(current_x, current_y, shaft_rad, key_width_ang, key_thick, theta_shaft,11,1,4);
        delay_time(2000),
        }
whiie((theta_shaft-theta_hole)>0.5)
{
clrscr();
theta_shaft-=0.2;
draw(hole _x, hole_y, hole_rad, key_width_ang, key_thick, theta_hole,7,1,10),
draw(current_x, current_y, shaft_rad, key_width_ang, key_thick, theta_shaft,11,1,4);
delay_time(1000);
}
closegraph();
}


void draw(int c_x, int c_y, int rad, float width_ang, int dep, float theta,int color, int pattern,int bk)
{
int x1,y1,x2,y2,x3,y3,x4,y4;
setcolor(bk);
circle(c_x, c_y, rad);
arc(c_x,c_y,theta+width_ang,theta-width_ang,0);
width_ang/=(180/3.14159);
theta/=(180/3.14159);
x1=c_x+rad*cos(width_ang+theta);
y1=c_y-rad*sin(width_ang+theta),
x2=x1+dep*cos(theta);
y2=y1-dep*sin(theta);
x4=c_x+rad*cos(theta-width_ang);
y4=c_y-rad*sin(theta-width_ang);
x3=x4+dep*cos(theta);
y3=y4-dep*sin(theta);
lineto(x1,y1);
setcolor(bk);
lineto(x2,y2);
lineto(x3, y3);
lineto(x4,y4);
setfillstyle(pattern,color);
floodfill(c_x,c_y,bk);
return;
}

void delay_time(int a)
{
int i;
double b=1.0,c;
for(i=0;i<a;i++)
{
do{
```

```
        c=log(b);
        b+=0.5;
}while(b<11.0);
}
return;
}

/*****************************************************************************/
/*****************************************************************************/
```

G3 ASSEMBLY TRIALS

```
/****************************************** ASSEMBLY*****************************/

#include <stdio.h>
#include <math.h>
#include <float.h>


/* ------------------- CONSTANTS-----------------------------*/

#define PEGS            5       /* Max. number of pegs          */
#define PEG_HEIGHT      5.0     /* Height of peg        */
#define MINPEG_DIA      2.0     /* Minimum diameter of peg      */
#define DIA_DIFF        0.1     /* Diameter difference peg-hole */
#define FORCE_Z             10.0/* Force in Z-axis on assembly  */


/* ------------------- VARIABLES ----------------------------*/

FRAME position [3];
int     peg_count [5] = {0,0,0,0,0};
REAL    peg_diameter [PEGS];
REAL    depth;
REAL    fZ;
/* ----------------- MAIN ----------------------------------*/

()

{
        int     pegs;           /* Number of pegs to assemble   */
        int     startpos; /* Carry peg from conveyor1     */
        int     endpos;         /* Carry assembly to conveyor2  */

        printf ("\n");
        printf (" `ROBOTIC ASSEMBLY' by `Arun  Jaura'\n");
        printf ("\n");

        initialisation();

        do      {
                enter_pegs(&pegs);
```

```
                if (fZ  != 10.0)    {

                  depth = 5.0 + (PEGS -

                  enter_startpos(&startpos);
                  enter_endpos(&endpos, startpos);

                  peg_count[startpos-1] = pegs;
                  carry(pegs, startpos, endpos);

                  }
        }

        printf("\n");
        printf("   Program  ASSEMBLY  finished. \n");
}

/* ------------------------ initialisation  -------------------*/

initialisation ()
{
        int i;

        for (i=0; i<pegs;  i++)
                peg_diameter[i] = MINPEG_DIA + (i-1) * DIA_DIFF.

        for (i=0; i<5; i++)             {
                makerotation(&position[i].rot, &yaxis, 90.0);
                position[i].transl.x = 50.0 - ABS(i-1) *2.0,
                position[i].transl.y = 0 + (i-1)*13.8,
                position[i].transl.z = PEG_HEIGHT.

           }
           speedfactor = 0.5;
}

/* ------------------------ enter_startpos----------------------*/

enter_startpos(startpos)
int *startpos;
{
        do  {

           printf("Enter start position  1...5: ");
           scanf("%d", startpos);

           if (startpos < 1  ||  *startpos >5)  {
             printf("Start position not in range \n");
             *startpos = 0;
           }
         }  while (!*startpos);
}

/* ---------------------------- enter_endpos -------------------*/
```

```
enter_endpos(endpos, startpos)
int *endpos, startpos;
{
     do {
          printf("Enter final position 1 .. 5. ");
          scanf("%d", endpos);

          if (*endpos < 1  || *endpos > 5) {
            printf("End position not in range\n");
            *endpos = 0;
          }
          if (*endpos == startpos) {
            printf("Assembly Successful\n");
            *endpos = 0;
          }
    } while (!endpos);
}



//*****************************************************************************//
//*****************************************************************************//




/****************************** INCREMENTAL STEPS FOR MOVE*****************/

#include <stdlib.h>
#include <stdio.h>
#include <conio.h>
#include <dos.h>
#include <math.h>
#include <ctype.h>
#include <string.h>
#include "crs232.h"
#include "pc8250.h"

extern void WritePort(char c);
extern char ReadPort();
extern void Move ( float X , float Y , float Z , float T ) ;
extern    void Where ( float *X , float *Y , float *Z , float *T ) ;
extern    void Close () ;
extern    void Open () ;


#define goal_x 100
#define goal_y 100
#define goal_z 100
#define fzmax  50
#define origin_x 0
```

```
#define origin_y 0
#define origin_z 0


void main()

{
        int i,j,k;
        float xx, yy, zz, tt, fxx, fyy, fzz, delfz,angle,
        float path_x[20],path_y[20],path_z[20];
        float force_x[360],force_y[360],force_z[360],delfz1[20];
        int casenum,
        for(i=0;i<10;i++)
                {

                Move(xx,yy,zz,tt);
                Where(&xx,&yy,&zz,&tt);
/*              path_x[i]=robot1.x;
                path_y[i]=robot1.y;
                path_z[i]=robot1.z;
                Measure(fxx,fyy,fzz,delfz);
                force_x[i]=fxx;
                force_y[i]=fyy;
                force_z[i]=fzz,
                if((robot1.x==goal_x)&(robot1.y==goal_y)&(fzz==0))
                  break;
*/
                }

}


/*****************************************************************************/
/*****************************************************************************/



/********************************** PRELIMINARY RUNS ***********************/

#include <dos.h>
#include <string.h>
#include <math.h>
#include <stdio.h>
#include "prelim.h"
#include "crs232.h"
#include "pc8250.h"


PC8250 port(COM2,9600);
char c;

PRELIM::PRELIM()
```

```
{
        x=0;
        y=0;
        z=0;
        fx=0;
        fy=0;
        fz=0,
}

//void Move (float x, float y, float z, float t)
int PRELIM::MOVE(float xx,float yy, float zz, float tt)

{
char s[80];
int a,i;
                {
                        a= strlen(s);
                        for(i=0,i<a;i++)
                                port.EcrirePort (s[i]),
                }

                if (( c= port.LirePort () ) > 0 ) cout << hex << (char)  c;



//      delay (1000);
//      Move(xx,yy,zz,tt);
        x+=xx;
        y+=yy;
        z+=zz;
        return 0;
}

int PRELIM::Rotate(float angle)
{
return 0;
}

int PRELIM::Measure(float fxx,float fyy, float fzz, float delfz)
{
return 0;
}
int PRELIM::GETPOS(float xx,float yy, float zz, float tt)
{
return 0;
}
int PRELIM::SETPOS(float xx,float yy, float zz, float tt)
{
return 0;
}
int PRELIM::SEEPOS(float xx,float yy, float zz, float tt)
{
return 0;
}
```

```
// ********************** Début de ARUNrob.CPP **********************
//
//
#include <stdlib.h>
#include <stdio.h>
#include <conio.h>
#include <dos.h>
#include <math.h>
#include <ctype.h>
#include <string.h>
#include "crs232.h"
#include "pc8250.h"

PC8250 port( COM2, 9600);
char c;
float X,Y,Z,T;

void Move ( float X , float Y , float Z , float T ) ;
void Where ( float *X , float *Y , float *Z , float *T ) ;
void Close () ;
void Open () ;
void WritePort(char c);
char ReadPort();

extern long LireTemps( void );
void delai( long milliseconds );

void Put_Str ( char *Str ) ;
void Get_Str ( char *Str ) ;


void delai( long milliseconds )
{
   long temps_debut;

   temps_debut = LireTemps();
   if ( ( LireTemps() - temps_debut ) >= milliseconds ) return;

}


void WritePort( char c)
{
        port.EcrirePort( c );
```

```
}


char ReadPort()
{

 return (port.LirePort() );
}

void Move ( float X , float Y , float Z , float T )
{
   char X_str[80] , Y_str[80] , Z_str[80] , T_str[80] ;

   sprintf ( X_str , "%f" , X ) ;
   sprintf ( Y_str , "%f" , Y ) ;
   sprintf ( Z_str , "%f" , Z ) ;
   sprintf ( T_str , "%f" , T ) ;
   port.EcrirePort ( 2 ) ;
   Put_Str ( X_str ) ;
   Put_Str ( Y_str ) ;
   Put_Str ( Z_str ) ;
   Put_Str ( T_str ) ;
         delai (1000) ;
}

void Put_Str ( char *Str )
{
   int Index ;

   Index = 0 ;

   while ( Str[Index] != NULL )
   {
      port.EcrirePort ( (unsigned char)Str[Index] ) ;
      Index++ ;
   }
   port.EcrirePort ( (unsigned char)13 ) ; //CR
   port.EcrirePort ( (unsigned char)10 ) ; //LF
}

void Where ( float *X , float *Y , float *Z , float *T )
{
   char X_Str[80] , Y_Str[80] , Z_Str[80] , T_Str[80] ;

   port.EcrirePort ( 1 ) ;
   Get_Str ( X_Str ) ;
   Get_Str ( Y_Str ) ;
   Get_Str ( Z_Str ) ;
   Get_Str ( T_Str ) ;
   *X = (float) atof ( X_Str ) ;
   *Z = (float) atof ( Z_Str ) ;
   *Y = (float) atof ( Y_Str ) ;
   *T = (float) atof ( T_Str ) ;
         delai (1000) ;
```

```
}

void Get_Str ( char *Str )
{
  char Index , Tampon ;
  Index = 0 ;

  while ( (port.LirePort ( Tampon )) == -1 ) ;
  while ( Tampon != 13 )
  {
    Str[Index] = Tampon ;
    Index ++ ;
    while ( (port.LirePort ( Tampon )) == -1 ) ;
  }
  while ( (port.LirePort ( Tampon )) == -1 ) ; //enleve le LF
  Str[Index] = NULL ;
}

void Close ()
{
  port.EcrirePort ( 3 ) ;
        delai (1000) ;
}

void Open ()
{
  port.EcrirePort ( 4 ) ;
        delai (1000) ;
}
```

```
// FIRST.CPP == THIS PROGRAM IS TO INPUT STANDARD INPUTS FROM THE
// DO-LOOP FOR THE ROBOT TO MOVE TO DIFFERENT LOCATIONS.


#include "robot.h"
#include <iostream.h>

class Process {
private:
        double X,Y,Z,T;
public:
        void MoveIT (int START, int END);
                };

void Process::MoveIT (int START, int END)
```

```
{
int Index;
Robot *Rp;

        float x1,y1,z1,t1;
        Index = 0;

        Rp = new Robot (2);

        x1 = 100; y1 = 20; z1 = 0; t1 = 0;
        Rp->Move(x1,y1,z1,t1),

        do {
         x1+=15; y1 += 10; z1 +=15; t1 +=30;
         Rp->M ove(x1,y1,z1,t1);
         Index++;
         Rp->Where(X,Y,Z,T);
         cout << "step" << Index;
         cout << "X, Y, Z, T:  " << X << ", " << Y << ", "
            << Z << ", " << T << "\n";
         Rp->Close();
        } while (Index < 5);

}


/*****************************************************************************/
/*****************************************************************************/



// ALTFIRST.CPP == THIS PROGRAM IS TO INPUT RANDOM POINTS FOR THE
// ROBOT TO MOVE TO DIFFERENT LOCATIONS.

#include "robot.h"
#include <iostream.h>

class Process {
private:
        double X,Y,Z,T;
public.
        void MoveIT (int START, int END);
                };

void Process::MoveIT (int START, int END)
{
int Index;
Robot *Rp;

        float x1,y1,z1,t1;
        Index = 0;

        Rp = new Robot (2);
```

```
        x1 = 100; y1 = 20; z1 = 0; t1 = 0,
        Rp->Move(x1,y1,z1,t1);
        Rp->Where(X,Y,Z,T);
        cout << "X, Y, Z, T:  " << X << ", " << Y << ", "    // May be this
            << Z << ", " << T << "\n";   // cout command is not allowed
                                         // check it up when you run it

        x1 = 150; y1 = 60; z1 = 30; t1 = 30;
        Rp->Move(x1,y1,z1,t1);
        Rp->Where(X,Y,Z,T);
        cout << "X, Y, Z, T:  " << X << ", " << Y << ", "    // May be this
            << Z << ", " << T << "\n";   // cout command is not allowed
                                         // check it up when you run it

        x1 = 200; y1 = 80; z1 = 60; t1 = 60;
        Rp->Move(x1,y1,z1,t1);
        Rp->Where(X,Y,Z,T);
        cout << "X, Y, Z, T.  " << X << ", " << Y << ", "    // May be this
            << Z << ", " << T << "\n";   // cout command is not allowed
                                         // check it up when you run it


//      do {
//        x1+=15; y1 += 10; z1 +=15; t1 +=30;
//        Rp->Move(x1,y1,z1,t1);
//        Index++;
//        Rp->Where(X,Y,Z,T);
//        cout << "step" << Index;
//        cout << "X, Y, Z, T:  " << X << ", " << Y << ", "
//            << Z << ", " << T << "\n";
//        Rp->Close();
//      } while (Index < 5);

// }


/*****************************************************************************/
/*****************************************************************************/


/********************************** TESTING COMMUNICATION************/


.PROGRAM tst_com()
        serial = 12
        terminal = 4
        ATTACH (serial)
        ATTACH (terminal)
        fin = 1
        sortie = 13

        WHILE fin DO
          input = GETC(serial)
          WRITE (terminal) "caractere :", input
```

```
            WRITE (serial) sortie
;           READ (terminal) touche
;           IF touche <> 5 THEN
;           fin = 0
;       END

        END

/*************************************************************************/
/*************************************************************************/


PROGRAM tst_move()
        serial = 12
        terminal = 4
        ATTACH (serial)
        ATTACH (terminal)
        WRITE (terminal) "Wait for a command ... "
        input = GETC(serial)
        WHILE (input <> 53) DO
          CASE input OF
           VALUE 49:
                MOVES p1
           VALUE 50:
                MOVES p2
           VALUE 51:
                MOVES p3
           ANY
                WRITE (terminal) "Invalid command"
          END
          input = GETC(serial)
        END


/*************************************************************************//
//*************************************************************************//




/*********************************ALGORITHM RUNS ****************************/

#include<stdio.h>
#include<conio.h>
#include<stdlib.h>

#include"u8250.h"


/*******************************************************************/


double get_posi(void)
```

```
{
char recu, string[24];
double result;
int i;

do
{
        recu = getc_uart();
}while(recu != 0x20);

i=0;
do
{
        if(recu != 0x00)
        {
                recu = getc_uart();
                string[i] = recu;
        }
        else
        {
                string[i] = NULL;
        }
        i++;
}while(recu != 0x00);

result = atof(string);
return(result);
}


/*****************************************************************/
double *get_xyzypr(void)
{
char request;
int i;
double *pos;

request = 0x01;
putc_uart(request);

for(i=0; i<6; i++)
{
        pos[i] = get_posi();
}

return(pos);
}

/*****************************************************************/
void put_str(char* Str)
{
int index;

index = 0;
```

```
while(Str[index] != NULL)
{
        putc_uart(Str[index]);
        index++;
}

putc_uart(0x0D);            // CR
putc_uart(0x0A);        // LF

return;
}


/*****************************************************************/
void send_xyzypr(double *posi)
{
char X_str[80], Y_str[80], Z_str[80], y_str[80], p_str[80], r_str[80];
char request;

printf("\n\nThe new position is :\n");
printf("X : %5.4f\t Y: %5.4f\t Z : %5.4f\n", posi[0], posi[1], posi[2]);
printf("y : %5.4f\t p: %5.4f\t r : %5.4f\n", posi[3], posi[4], posi[5]);
printf("\n\nPress a key to continue");
getch();


sprintf(X_str, "%f", posi[0]);
sprintf(Y_str, "%f", posi[1]);
sprintf(Z_str, "%f", posi[2]);
sprintf(y_str, "%f", posi[3]);
sprintf(p_str, "%f", posi[4]);
sprintf(r_str, "%f", posi[5]);

request = 0x02;
putc_uart(request);

put_str(X_str);
put_str(Y_str);
put_str(Z_str);
put_str(y_str);
put_str(p_str);
put_str(r_str);

return;
}


/*****************************************************************/


void main(void)

{
```

```
char caract;
double *test;
int fin,
FILE *fp;


init_uart8250(2,0x03,0x0C);

fin = 0;
fp = fopen("c:\\denis\\test.dat",""w+");


do
{
        clrscr();
        printf("Enter a character (g: get position; s:send new position; c:close hand; "),
        printf("\t\to:open hand, h:go to home; q:quit)");
        caract = getch();
//      putc_uart(caract);

        switch(caract)
        {
                case 'g':
                        test = get_xyzypr();
                        printf("\n\nX = %5.4f\t Y = %5.4f\t Z = %5.4f\n",test[0], test[1], test[2]),
                        printf("y = %5.4f\t p = %5.4f\t r = %5.4f\n",test[3], test[4], test[5]),
                        printf("\nPress a key to continue");
                        fprintf(fp,"X = %3.3f\t Y = %3.3f\t Z = %3.3f\t",test[0], test[1], test[2]),
                        fprintf(fp,"y = %1.1f\t p = %3.1f\t r = %3.3f\n",test[3], test[4], test[5]);

                        getch();
                        break;

                case 's':
                        test[0] = test[0] + 10;
                        test[1] = test[1] - 0;
                        test[2] = test[2] - 0;
                        test[5] = test[5] - 0;
                        send_xyzypr(test);
                        break;

                case 'c':
                        printf("\n\nClose Hand");
                        putc_uart(0x03);
                        printf("\nPress a key to continue");
                        getch();
                        break;

                case 'o :
                        printf("\n\nOpen Hand");
                        putc_uart(0x04);
                        printf("\nPress a key to continue");
                        getch();
                        break;
```

```
                case 'h':
                        printf("\n\nGo to Home position");
                        putc_uart(0x05);
                        printf("\nPress a key to continue");
                        getch();
                        break;

                case 'q':
                        putc_uart(caract);
                        fin = 1;
                        break;
        }

}while(!fin);

fclose(fp);

return;

}



/*****************************************************************************/
/*****************************************************************************/




/************************* ALG O1 *******************************/

#include<stdio.h>
#include<conio.h>
#include<stdlib.h>

#include"u8250.h"

void FORMOM(void);

double *posi;
double fxx,fyy,fzz,delfz,angle,Mx,My,Mz;

/*************** FORMOM sub-routine ***********************************/
void FORMOM(void)
{

/***** TO BE DONE ****/
fxx = 0;
fyy = 0;
fzz = 0;
```

```
delfz = 0;
angle = 0;
Mx = 0;
My = 0;
Mz = 0;

return;

}

/********** GET sub-routine **********/

double get_posi(void)

{
char recu, string[24];
double result;
int i;

do
{
        recu = getc_uart();
}while(recu != 0x20);

i=0;
do
{
        if(recu != 0x00)
        {
                recu = getc_uart();
                string[i] = recu;
        }
        else
        {
                string[i] = NULL;
        }
        i++;
}while(recu != 0x00);

result = atof(string);
return(result);
}


/********** GET routine **********/

void get_xyzypr(void)
{
char request;
int i;


request = 0x01;
putc_uart(request);
```

```
for(i=0; i<6; i++)
{
        posi[i] = get_posi();
}

return;
}

/************************SEND sub-routine ********************************/

void put_str(char* Str)
{
int index;

index = 0;
while(Str[index] != NULL)
{
        putc_uart(Str[index]);
        index++;
}

putc_uart(0x0D);            // CR
putc_uart(0x0A);        // LF

return;
}



/******************** SEND routine ************************************/

void send_xyzypr(void)
{
char X_str[15], Y_str[15], Z_str[15], y_str[15], p_str[15], r_str[15];
char request;

printf("\n\nThe new position is :\n");
printf("X : %5.4f\t Y: %5.4f\t Z: %5.4f\n ", posi[0], posi[1], posi[2]);
printf("y : %5.4f\t p: %5.4f\t r: %5.4f\n ", posi[3], posi[4], posi[5]);
printf("\n\nPress a key to continue");
getch();


sprintf(X_str, "%f", posi[0]);
sprintf(Y_str, "%f", posi[1]);
sprintf(Z_str, "%f", posi[2]);
sprintf(y_str, "%f", posi[3]);
sprintf(p_str, "%f", posi[4]);
sprintf(r_str, "%f", posi[5]);

request = 0x02;
putc_uart(request);
```

```
put_str(X_str);
put_str(Y_str);
put_str(Z_str);
put_str(y_str);
put_str(p_str);
put_str(r_str);
return;
}

/******************* OTHER FUNCTIONS ********************************/

void GET(void)

                {

                FILE *fp;

                fp = fopen("c:\\denis\\test.dat","w+"),


                        get_xyzypr();
                        printf("\n\nX = %5.4f\t Y = %5.4f\t Z = %5.4f\n ",posi[0], posi[1], posi[2]),
                        printf("\n y = %5.4f\t  p = %5.4f\t r = %5.4f\n",posi[3], posi[4], posi[5]);
                        printf("\nPress a key to continue");
                        fprintf(fp,"X = %3.3f\t Y = %3.3f\t Z = %3.3f\t", posi[0], posi[1], posi[2]);
                        fprintf(fp,"y = %1.1f\t p = %3.1f\t r = %3.3f\n",posi[3], posi[4], posi[5]);

                        fclose(fp);
                        return;
                }

                void SEND(double A,double B, double C, double D, double E,double F)
                {
                        posi[0] = posi[0] + A;
                        posi[1] = posi[1] + B;
                        posi[2] = posi[2] + C;
                        posi[3] = posi[3] + D;
                        posi[4] = posi[4] + E;
                        posi[5] = posi[5] + F;
                        send_xyzypr();
                        return;
                }

                void CLOSE(void)
                {
                        printf("\n\nClose Hand");
                        putc_uart(0x03);
                        printf("\nPress a key to continue");
                        return;
                }

                void OPEN(void)
                {
                        printf("\n\nOpen Hand");
```

```
                putc_uart(0x04);
                printf("\nPress a key to continue");
                return;
        }

        void HOME(void)
        {
                printf("\n\nGo to Home position");
                putc_uart(0x05);
                printf("\nPress a key to continue");
                return;
        }

        void QUIT(void)
        {
                putc_uart(0x113);
                return;
        }

/****************** PROGRAM ALG 01 ********************************/

void main(void)

{


int LOOP;
int ENDLOOP;
int ASSEMBLY;
int FLAG;
int END_ASS, ASS_ABORT, CYCLE;

init_uart8250(2,0x03,0x0C);


ENDLOOP = 0;
ASSEMBLY = 0,
FLAG = 1;

posi[0] = 100;          // to send initial position to robot in xyzypr
posi[1] = 100;
posi[2] =  50;
posi[3] =   0;
posi[4] =   0;
posi[5] =  30;
send_xyzypr();   // robot moves to pre-defined position to begin assy.


do
{
        clrscr();

        GET();
        FORMOM();
```

```
if (fzz == 0)
{
        ASSEMBLY = 1;
        ENDLOOP = 1;
}
else
{
        LOOP = 0;
        do
        {
                SEND(0,0,0,0,0,1);          // rotate cw 1 degree steps
                GET();
                FORMOM();
                if(fzz == 0)
                {
                        LOOP = 10;                  // maximum rotation 10 deg cw
                        ASSEMBLY = 1;
                        ENDLOOP = 1;
                }
                else
                {
                        LOOP++;
                }
        } while (LOOP < 10);

        if (ASSEMBLY != 1)
        {
                SEND(0,0,0,0,0,-10);        // return to original -10 deg ccw
                LOOP = 0;
                do
                {
                        SEND(0,0,0,0,0,-1);         // rotate -1 deg ccw upto 350 deg
                        GET();
                        FORMOM();
                        if (fzz == 0)
                        {
                                LOOP = 350;
                                ASSEMBLY = 1;
                                ENDLOOP = 1;
                        }
                        else
                        {
                                if (Mx == 0)
                                {
                                        if (FLAG == 1)
                                        {
                                                SEND(0,0,-5,0,0,0); // retract z
                                                                    //upwards
                                                GET();
                                                SEND(0,5,0,0,0,0);          // move
// in y=1/2(dia of shaft)
                                                SEND(0,0,5,0,0,0);          // move
                                                                    //z downwards
```

```
                                                FLAG = FLAG + 1;
                                                LOOP = 350;
                                        }
                                        else
                                        {
                                                LOOP = 350;
                                                ASSEMBLY = 0;
                                                ENDLOOP = 1;
                                        }
                                }
                                else
                                {
                                        LOOP = LOOP + 1;
                                }
                        }
                } while (LOOP < 350);
        }
    }
}while (!ENDLOOP);

if(ASSEMBLY)
{
        CYCLE = 0;
        LOOP = 0;
        END_ASS = 0;
        ASS_ABORT = 0;
        do
        {               // send 1 mm in + z and record
                GET();
                SEND(0,0,1,0,0,0);      // until assembly complete
                                                // and fzz = fzmax and z = goal_z
                FORMOM();
                if(fzz != 0)
                {
                        if(LOOP < 20)
                        {
                                CYCLE++;
                                if(CYCLE < 2)
                                {
                                        SEND(0,0,-CYCLE,0,0,0);
                                }
                                else
                                {
                                        ASS_ABORT = 1;
                                        END_ASS = 1;
                                }
                        }
                        else
                        {
                                ASS_ABORT = 0;
                                END_ASS = 1;
                        }
                }
                else
```

```
                {
                        LOOP++;
                        if(LOOP == 20)
                        {
                                ASS_ABORT = 0;
                                END_ASS = 1;
                        }
                }
        }while(!END_ASS);

        if(!ASS_ABORT)
        {
                OPEN;
        }
        SEND(0,0,-(CYCLE+10),0,0,0);
        HOME;

}

return;
}
```

Appendix H

Set-up / Hardware Photographs

This appendix contains pictures of the hardware and the set-up. Although, a video has been prepared for the demonstration of tasks and display of hardware, the pictures supplement the text for readers.

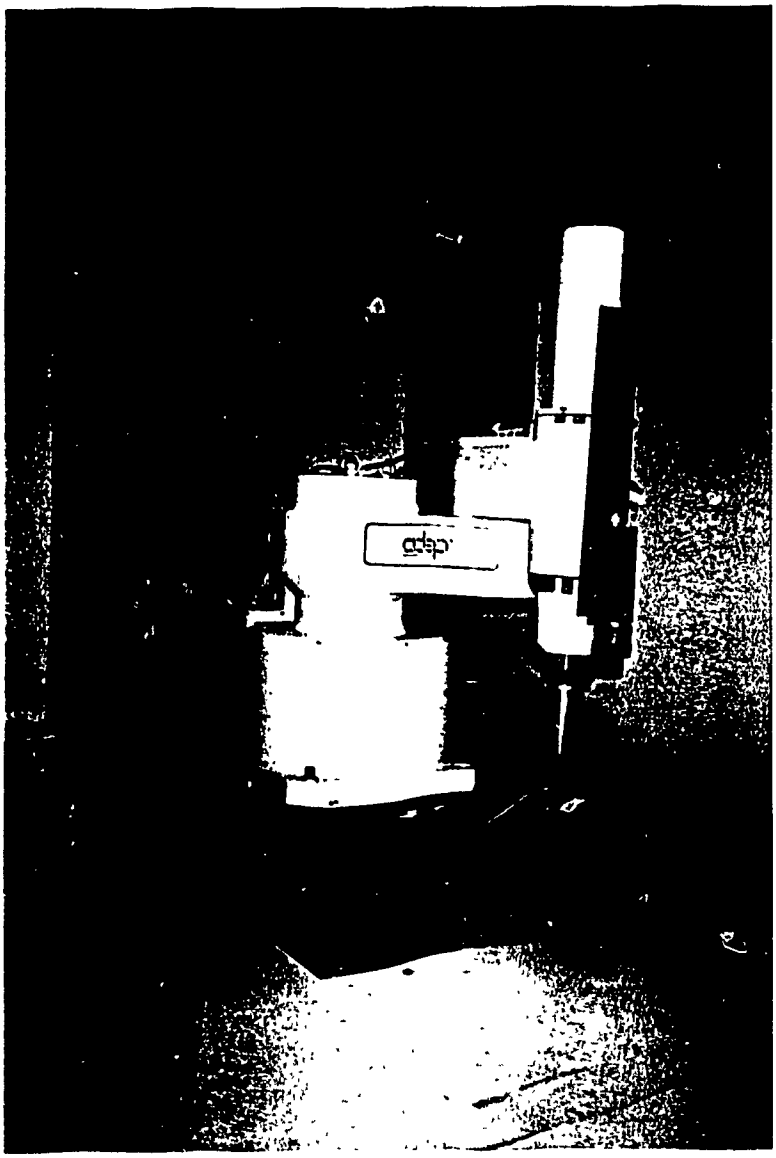


Fig. H1 The Adept Robot

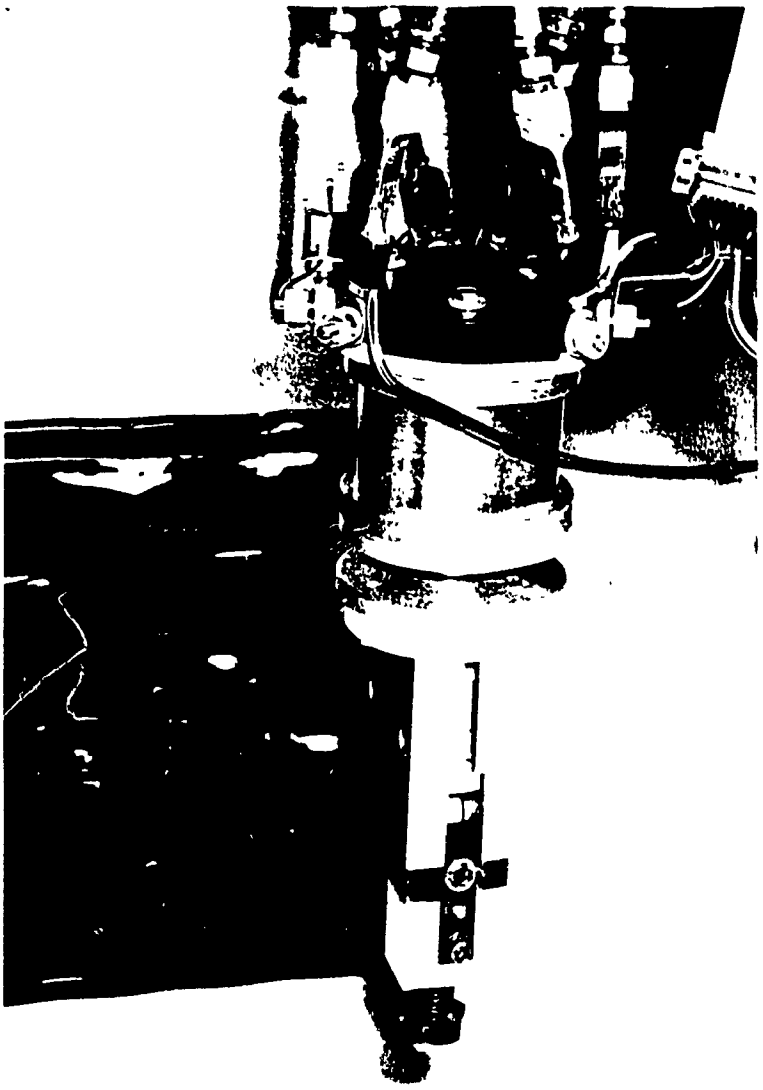


Fig. H2 The Passive compliance device fixed to the end-effector

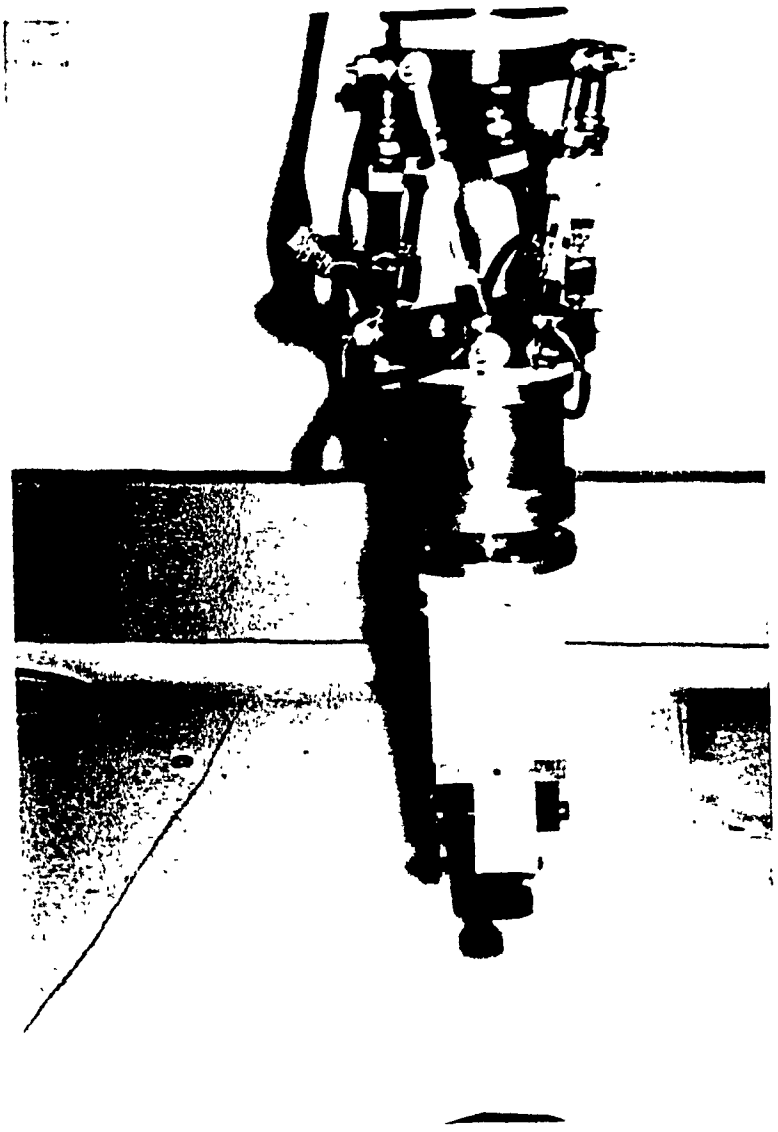


Fig. H3 The Sensor, PCD, and end-effector (top to bottom)

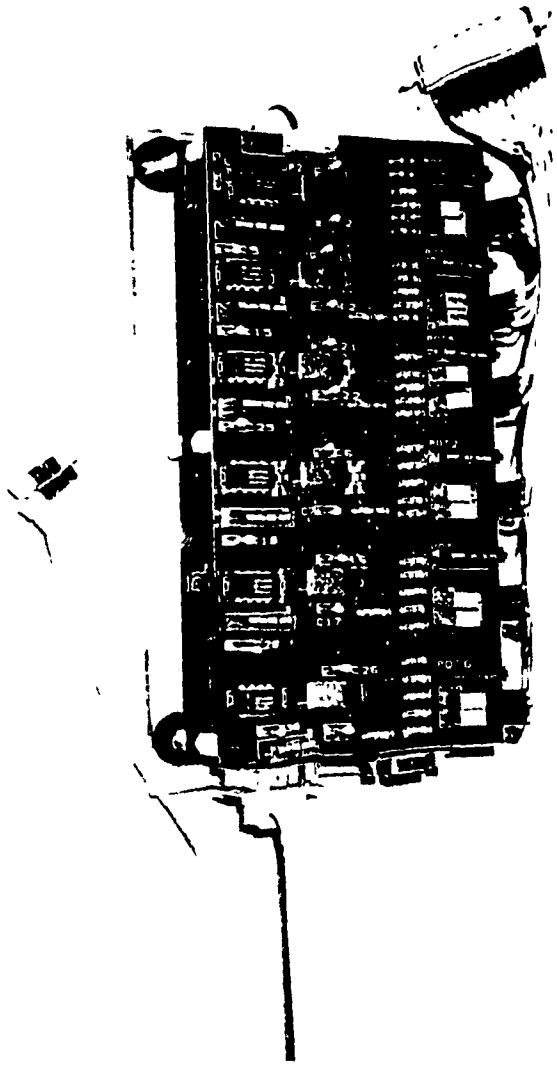


Fig. H4 The Amplifier card for sensor

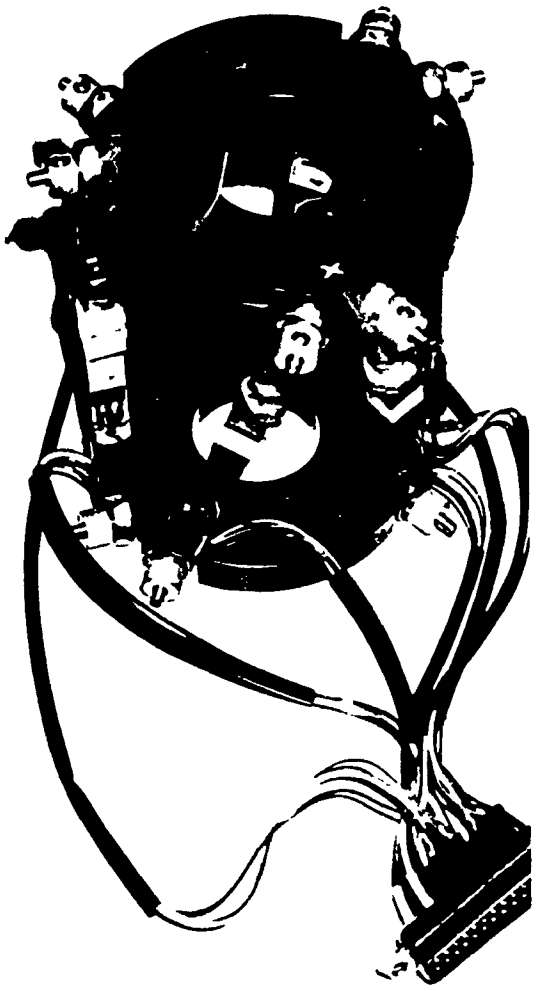


Fig. H5 A closer look at the Sensor

Fig. H6 A closer look at the PCD