Robot Programming by Human Demonstration

by

Nathan Joseph Delson

Submitted to the Department of Mechanical
Engineering in Partial Fulfillment
of the Requirements for the Degree of

Doctor of Philosophy

at the

Massachusetts Institute of Technology

May 1994

Signature of Author..[illegible]
Department of Mechanical Engineering

Certified by..[illegible]
Professor Harry West
Thesis Supervisor

Accepted by..
Professor Ain Sonin
Chairman, Graduate Thesis Committee

Robot Programming by Human Demonstration

by

Nathan Joseph Delson

Submitted to the Department of Mechanical Engineering
on May 5, 1994 in partial fulfillment of the
requirements for the Degree of Doctor of Philosophy in
Mechanical Engineering

Abstract

Robots can be programmed to perform different tasks, however, often the difficulty of the programming process limits more widespread use of robotic technology and discourages short term applications. To facilitate the use of robotics an intuitive method of programming is developed, Programming by Human Demonstration. The programmer demonstrates how the task is performed using a teaching gripper that measures the human's forces and positions. A robot program is generated from the demonstration data without using a model of the task geometry.

A direct approach to generating a robot program would be to simply duplicate the demonstrated trajectory. However, direct duplication may not succeed if there is variation in the environment. Furthermore, the human trajectory may contain unnecessary motion that would result in an unsatisfactory robot trajectory. To identify a manipulation strategy, multiple demonstrations of the task are performed and comparisons between demonstrations are used to distinguish between human adaptation and inconsistency. Human inconsistency is not interpreted as noise that should be eliminated from the analysis, but provides information about the accuracy requirements of the task for both position and force trajectories. Adaptation is identified by human motion that corresponds to detectable variations in the environment, and is incorporated into the robot program.

The analysis is applied to obstacle avoidance and simple assembly tasks consisting of 3D translation. A robot program is generated that can successfully perform the task provided that variations in the environment are not larger than that encountered during the demonstrations. A robot compliance controller is specified that will adapt to workpiece misalignment, and limit robot errors to the accuracy requirements defined by the demonstrator's inconsistency. In addition, the task is segmented into subtasks, and subtask termination conditions are identified that enable the robot to switch from one subtask to the next. Robot success is guaranteed by ensuring that the position accuracy is as high enough to avoid obstacles and reach the desired target position, the direction of robot force maintains the desired contact with the workpiece surfaces, and the magnitude of the robot force is less than the maximum demonstrated force to avoid damage to the parts.

Thesis Supervisor: Professor Harry West

Title: Associate Professor of Mechanical Engineering

Acknowledgments

A PhD thesis is an opportunity for one to explore an area of research which often leads in unanticipated directions. This experience can be significantly enriched if one is provided with encouragement as well as freedom in the pursuit of this objective. For this opportunity I would like to thank Professor Harry West, who throughout my graduate education has provided a stimulating environment, challenging objectives, and a sense of partnership in the pursuit of our research goals.

I would like to thank the members of my committee for their valuable contributions; Professor Haruhiko Asada, Professor Neville Hogan, and Dr. Daniel Whitney. I would additionally like to thank Professor. Asada for his commitment to graduate education, both in teaching and his support of student conferences. His efforts have increased the quality of my graduate education. I would also like to thank Professor James Munkres for his input in the area of topology.

Numerous individuals have contributed to the efforts in this research project. I am grateful for their contributions, and camaraderie in research. James Pinkney provided a significant contribution in his Master's thesis which involved the design and fabrication of the teaching gripper used in the experiments. The following contributions were made as Bachelor's theses: Chris Doehler incorporated the Logitech ultrasonic sensor into the system, Susannah Gardner built and evaluated a preliminary ultrasonic position sensor, and Pam Sirivedhin built a passive compliance device for the robot and designed a teaching interface for position tasks. Other contributions were: Cheng-Jung Chiu developed an automatic calibration algorithm between the robot and teaching gripper position sensor, Charlie Choi built and early version of the teaching gripper, Taro Harima set up the interface between the robot and the computer, Hyoseok Yang evaluated the Logitech sensor accuracy. In out trip to Japan, Soo Yong Lee helped in the implementation and control of the experimental robot at Daikin.

This work has been supported by Daikin Mechanical Engineering Laboratory, Osaka, Japan.

I would like to thank Rienk Bakker, Mohamed Khemira, and Ross Levinsky, for sharing the sometimes rocky, but always fun, experiences of graduate student life.

The person with whom I have shared most, the challenges and joy of performing (and completing) graduate research is my wife Mary Lee. You have kept my dreams alive throughout it.

Table of Contents

CHAPTER 1

Introduction

Robots can be programmed to perform different tasks, however, often the difficulty of the programming process limits more widespread use of robotic technology and discourages short term applications. Most current robot applications are in industrial settings, where large batch size justifies a significant programming effort. However an easier method of robot programming will facilitate the introduction of robotics into new areas such as: agile manufacturing, biotechnology research, and the office and service environments.

As more difficult tasks are automated, robots are required to adapt to variations in the environment. One approach to adapt to the environment has been to incorporate force sensors into robotic systems, and to control both the force and position of the robot. However, programming a robot to utilize force information requires even more sophisticated programming methods. Often the factor limiting automation is the ability to generate a program rather than the physical capabilities of the robot.

A person can often figure out how to perform a task much more easily than the efforts required to program a robot to perform the same task. Accordingly, human manual ability has been utilized to assist in robot programming. For example, in walk through programming, the programmer grasps the robot's end effector and physically moves the robot through the desired trajectory. The robot records the motion and then plays it back during task execution. This method is often applied to spray painting robots, yet is limited to use with robots that can be physically moved by a person.

Most robots, however, have gear reductions which prevent the robot to be backdriven, i.e. moved by an external force. With these robot's a popular method of programming is to use a teach pendent control box. Buttons on the teach pendent move the robot in different directions, and the robot trajectory can be recorded and then played back. A limitation with both the teach pendent approach and walk through programming, is that the robot can only play back the specified trajectory. The programming process does not provide the robot with the ability to adapt to the environment. Furthermore, any unnecessary motion imparted during the teaching process is repeated each time the robot executes the task.

Another approach to robot programming, is off-line model based programming. Here a computer program generates the robot trajectory from a model of the task geometry. With this approach variations in the environment can be considered in the programming process, and an adaptation strategy incorporated into the robot program. Model based methods have been successfully applied to specific tasks, yet an algorithm is not currently available for general tasks. A primary limitation of model based programming is the analytical difficulty associated with generating the robot program. Another factor that limits short term applications is the necessity of generating a computer model of the task. In addition, fine tuning of the robot program is often required to account for discrepancies and simplifications in the task model.

The objective of this thesis is to develop an easier method to program robots that includes the ability to adapt to the environment using force and position sensors.

1.1. Programming by Human Demonstration

Programming by Human Demonstration (PHD) is an intuitive method of robot programming. The programmer demonstrates how the task is performed using a teaching gripper that measures the human's forces and positions, and the data gathered from the human is used to generate the robot program, as shown in Figure 1.1.

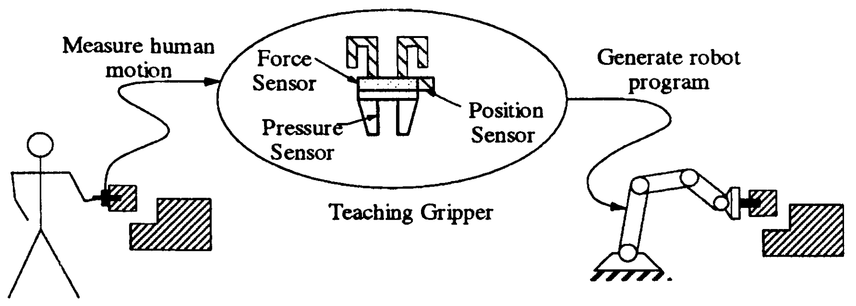


Figure 1.1: Programming by Human Demonstration
The human demonstrates the task with the teaching gripper, and the data is used to generate the robot program. The position sensor records the 6D position and orientation, the force sensor measures the 6D forces and torques, and the pressure sensor detects when the gripper is opened and closed.

A direct approach to generating a robot program would be to simply duplicate the demonstrated trajectory. However, direct duplication will not always succeed if there is variation in the environment. Furthermore, the human trajectory may contain unnecessary motion that would result in an unsatisfactory robot trajectory.

To identify a robust manipulation strategy, multiple demonstrations of the task are performed. Comparisons between demonstrations are used to identify a strategy for adapting to the environment, as well as identifying unnecessary motion that need not be incorporated into the robot program.

The potential advantages of the PHD approach is that it incorporates the ease of the walk through programming, with the ability to adapt to the environment possible in model based programming. Furthermore, a geometric model of the task is not required, and the use of demonstration data should allow the program to be executed without fine tuning. PHD research can also provide insight into how humans perform tasks, and be used to improve performance of telerobotic systems where human control is combined with robotic operation.

1.2. Scope of Thesis

The category of tasks addressed in the thesis are pick and place tasks, and simple assembly operations. Pick and place tasks consist of moving a part to a desired location while avoiding obstacles in the way. The objective of an assembly task is typically to place a part in a desired location relative to another part. Both pick and place tasks, and assembly tasks are simplified by considering only tasks which require part translation, but no rotation. The simplified assembly operations are referred to as contact tasks, and an example is shown in Figure 1.2.

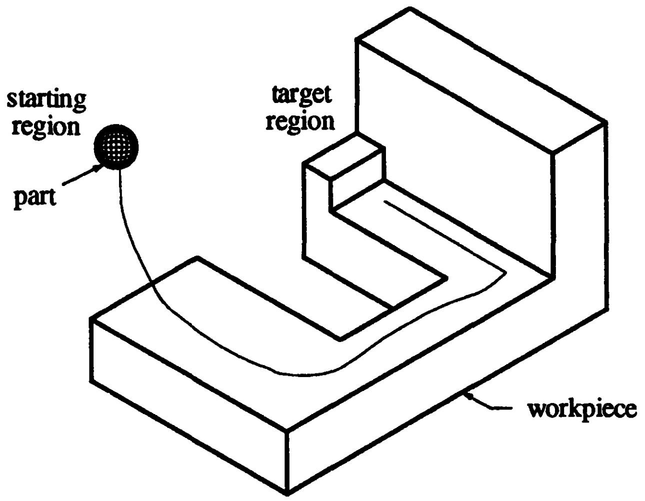


Figure 1.2: Example Contact Task Consisting of 3D Translation

The objective of the contact task is to place a part in a desired location relative to the workpiece. The part is shaped as a sphere so that part rotation does not effect task performance, and the workpiece is shaped as a polyhedral, i.e. its surfaces are flat. Misalignments in the workpiece position and orientation occur between demonstrations, that corresponding to variations that may occur on an assembly line. Thus the trajectory of the part consists of only translation, yet the workpiece may experience arbitrary misalignment including both rotation and translation. Successful implementation of a contact task requires that the part be placed in the desired position with sufficient accuracy, and that excessive contact forces do not damage the parts.

As the demonstrator performs a contact task, they modify their force and position trajectories to adapt to workpiece misalignment. The human may perform this level of adaptation subconsciously; however, this type of adaptation is explicitly required when programming a robot to perform a contact task. An objective of the thesis is to specify a robot program for which task success can be guaranteed, as long as workpiece misalignment is not larger than that encountered during the demonstrations.

1.3. Fundamental Assumptions

To generate a successful robot program using the PHD approach, it is necessary to capture how the human adapts to variations in the environment.

Accordingly, the programmer demonstrates the task a number of times, while small variations in workpiece location are introduced between the demonstrations. For ideal ease of programming, no restrictions would be made on how the demonstrator performs the task. However, to ensure that the demonstration data will be sufficient to generate a successful robot program the following two fundamental assumptions are made. A complete list of the assumptions used in the analysis is provided in Section 3.1.2 in Chapter 3.

- The demonstrator is restricted to using only sensory information that is measured by the teaching gripper. Otherwise, components of the human's adaptation strategy may not be captured by the demonstration data. Since the teaching gripper measures only position and force, the use of vision is prevented when demonstrating the task[1]. Of course the demonstrator can familiarize themselves with the task with their eyes open, and use the opportunity to develop a strategy that allows them to perform the task with their eyes closed. We wish to identify a similar strategy so that it may be implemented with a robot that uses only position and force sensors.
- The demonstrator uses the same manipulation strategy in all demonstrations. In the case of obstacle avoidance this means avoiding obstacles by passing them on the same side. In the case of contact tasks, this means that each demonstration is performed with the same sequence of contact states, i.e. the part contacts the same surfaces of the workpiece in the same order in each demonstration.

The category of tasks addressed in this thesis, along with the assumptions used, prevent immediate application of this research to many practical tasks. Indeed, most assembly operations do not involve spherical parts and therefore would require the analysis to incorporate part rotation as well as translation. Nevertheless an objective of this research is to lay a foundation which can be extended to a wider category of tasks. It is therefore desirable that the assumptions used in this thesis do not preclude implementation of

[1] Due to the inconvenience of demonstrating a task with one's eyes closed, the use of vision is allowed when it is not used to adapt to variations in the environment. One such case is where adaptation only occurs during the fine motion part of the task, and thus the use of vision is allowed during gross motion.

useful tasks, once the translation restriction is removed. Indeed, the assumption that precludes the use of vision is not a major limitation, as indicated by the fact that human's can perform many operations with their eyes closed. Furthermore, many useful tasks can be performed with a constant sequence of contact states, as shown by the examples in Appendix I.

1.4. Background

A review of robot programming methods is provided by Lozano-Pérez [1983]. Model based methods have been used to specify obstacle free robot trajectories, and to generate robot programs that use sensory information to adapt to the environment.

Due to the analytical difficulties of model based programming, approaches have been generated that use of human manipulation to assist in robot programming. Dessen and Balchen [1988] and Harima and West [1992] use demonstration data to play back position trajectories. These approaches effectively extend the advantage of walk through programming to robots that are not backdriveable, but do not incorporate adaptation into the robot program.

Other approaches have measured both human position and force. Hirzinger and Heindl [1983] incorporate both position and force information, yet use human input primarily to fine tune a robot program developed using knowledge of the task geometry. Kosuge, Fukuda, and Asada [1991] use human skill acquisition to identify robot controller parameters, yet also rely on a task model to interpret the demonstration data and generate the robot program.

Asada and Izumi [1987] use human demonstration data exclusively to identify both a robot trajectory and a hybrid force/position controller to perform contact tasks. A difficulty addressed by Asada and Izumi is that a single measurement of force and motion directions is not sufficient to identify the directions of admissible motion and force, which is required by a hybrid controller. Thus, to implement the controller it is assumed that the constraint surfaces are perpendicular to each other and that each contact state transition can only add a constraint, which limits possible workpiece geometries. In addition, while hybrid control compensates for workpiece translation by eliminating position errors in the force controlled directions, the effect of orientation misalignment of the workpiece is not explicitly addressed. Asada and Izumi [1987] originate the objective of generating both a robot trajectory and controller exclusively from demonstration data. The analysis in this thesis incorporates this objective and extends the analysis by allowing workpiece geometries with arbitrary surface orientations, and by allowing the number of constraints to increase or decrease at each contact state transition. In addition, here the analysis explicitly evaluates the effect of misalignment in workpiece translation and orientation, and quantifies robot performance in terms of position and force errors.

Liu and Asada [1992] use human demonstrations to automate a deburring operation. Both human position and force are measured, and the data is used to specify robot controller parameters that adapt to variations in burr size. It is recognized that human actions that do not correspond to detected changes in the environment present a problem in identifying an adaptation strategy. Liu and Asada present a method that distinguishes between human variation that is consistent with measured variations in the environment, and inconsistent variations which are attributed to adapting to sensory information that is not measured. The inconsistent human variations are removed from the analysis, and the robot strategy is based solely on the consistent data. This thesis presents an alternative approach for interpreting human demonstration data. Here all of the sensory information that the demonstrator uses is measured, and both human inconsistency and adaptation is used to specify the robot controller. The presence of human inconsistency is used advantageously to provide information regarding the task accuracy requirements.

Another approach of acquiring human demonstration data is through a vision system [Kuniyoshi, Inaba, and Inoue 1992; Ikeuchi, Kawade, and Suehiro 1993]. A limitation of this approach is that forces used by the human are not measured, which may contain a significant component of the manipulation strategy especially during the fine motion where parts contact each other. Accordingly, Ikeuchi et al. implement fine motion control by supplementing the demonstration data with model based knowledge.

The contribution of this thesis in relation to previous research is that both a robot trajectory and controller are generated exclusively from demonstration data, for the general case of a polyhedral workpiece and 3D translation. The demonstrator is restricted to using only sensory information that is measured; thereby ensuring that the demonstration data contains sufficient information to adapt to the environment. Accordingly, robot success can be guaranteed despite variations in the environment.

1.5. Overview of Thesis

An overview of the thesis is presented in this section by summarizing Chapters 2 through 5, and each of the following subsections corresponds to a chapter. First the analysis of pick and place tasks is presented in Chapter 2. In these tasks, there is no adaptation to the environment. However, unnecessary human motion is present in the demonstrations, and a method is presented to use human inconsistency to an advantage and improve the robot trajectory.

To implement contact tasks it is necessary to adapt to the environment. The PHD approach used to analyze contact tasks, segments the overall task into a sequence of subtasks. Chapter 3 presents a method of implementing individual subtasks, by extending the methods used in Chapter 2 for unconstrained motion to the case of constrained motion. Chapter 4 presents a method to segment the task into subtasks. Finally, Chapter 5 presents the method the robot uses to switch from one subtask to the next.

1.5.1. Obstacle Avoidance For Unconstrained Motion

The goal of a pick and place task is to move a part from a starting location to a target location without making contact with obstacles in the environment, as shown in Figure 1.3. The lack of contact between the part and the environment precludes any adaptation to the environment using force sensing. For pick and place tasks it is assumed that the obstacles remain stationary, and that the part is held without slip by a gripper. The objective of the analysis is to guarantee obstacle avoidance for both the part and the gripper, yet interference between the manipulator arm and the environment is not considered.

Obstacle avoidance is a well studied and often difficult problem in robotics [Latombe, 1991; and Lozano-Pérez, Jones, Mazer , and O'Donnell, 1992].

Typically a model of the environment is used and the robot trajectory is generated off-line. The number of potential trajectories can be very large, and selecting an appropriate robot path may require a computational expensive search. For tasks requiring low clearance between the robot and the obstacles, the level of detail necessary in the model increases along with the computational cost.

PHD addresses the difficulties encountered in off-line methods by utilizing the human's familiarity with manipulation and letting the demonstrator identify an obstacle free path. Here there are no variations in the environment, and direct duplication of the trajectory will be successful. However, in practice it is inefficient to exactly duplicate the human trajectory. Human motion often contains motion that is unnecessary to achieve the task, including vibrations and "wiggles." Exact duplication of the human trajectory would result in unnecessary robot motion.

One method to improve the robot trajectory is to smooth a demonstrated trajectory, yet by approximating the human motion obstacle avoidance cannot be guaranteed. Another method is presented by Ogata and Takahashi [1993], which supplements the demonstrated motion with a model of the obstacle locations. However, this method does not satisfy the objective of PHD which is to generate the robot program exclusively from the demonstration data.

The method presented in this thesis is to perform multiple demonstrations of the same task, to capture a range of obstacle free human motion, as shown in Figure 1.3. If all the demonstrations pass each obstacle on the same side, then the region between the demonstrations is obstacle free. A robot trajectory that stays within this region is guaranteed to be obstacle free. Using the multiple demonstration method, a trade off is not necessary between ensuring obstacle avoidance and efficient robot motion.

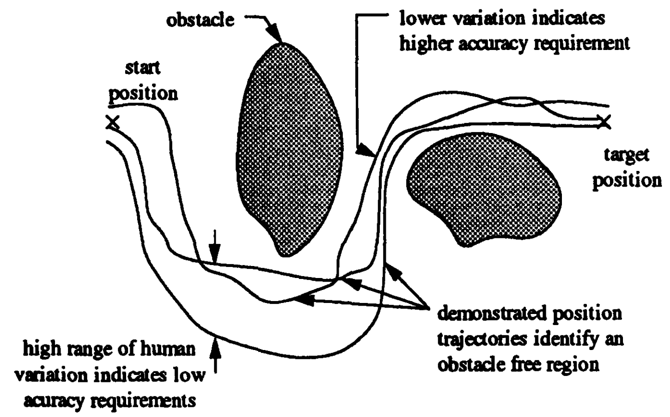

obstacle
lower variation indicates higher accuracy requirement
start position
target position
demonstrated position trajectories identify an obstacle free region
high range of human variation indicates low acuracy requirements

Figure 1.3: Multiple Demonstrations Approach

The human demonstrates the task multiple times, and the region between the demonstrations defines an obstacle free region, without the use of a model indicating obstacle locations. A robot trajectory within this region is guaranteed to be obstacle free

Since the environment is the same in all demonstrations, i.e. obstacle locations do not change, variations between demonstrations are unnecessary for task success and are thus interpreted as human inconsistency. Here inconsistency is not interpreted as noise that should be eliminated from the analysis, but provides information regarding the task requirements. The amount of human variation indicates the desired accuracy of the task, as shown by the wide and narrow regions in Figure 1.3. Motion that is necessary to achieve the task is present in all the demonstrations, such as the curved motion between the two obstacles in Figure 1.3, and is incorporated into the robot program. Human inconsistency identifies an obstacle free region between the demonstrations. Motion that does not effect task performance corresponds to variation within the obstacle free region, and need not be included in the robot trajectory.

In Chapter 2, a method is presented to identify the boundaries of the obstacle free region for 2D translation. In addition, an algorithm is presented that generates the shortest path robot trajectory within the obstacle free region, which is shorter than any of the demonstrations. This approach is extended to case of 3D translation, and an obstacle free robot trajectory is synthesized, although for 3D translation the trajectory is not the shortest path. The algorithm incorporates an optional buffer between the robot trajectory and the boundary of the obstacle free region, to allow for robot position errors. The buffer can guarantee obstacle avoidance even if there are errors in the robot position, as long as the buffer is larger than the maximum robot error.

The region of obstacle free motion defined from human inconsistency, identifies conditions *sufficient* to guarantee robot success. It is not *necessary* that the robot remain in the region bounded by the demonstrated trajectories. Indeed, if one had a model of the obstacle locations one could identify alternate obstacle free paths to the target that are outside the demonstrated region. However, the PHD approach uses only information from the demonstration data, and therefore identifies conditions *sufficient* to guarantee robot success.

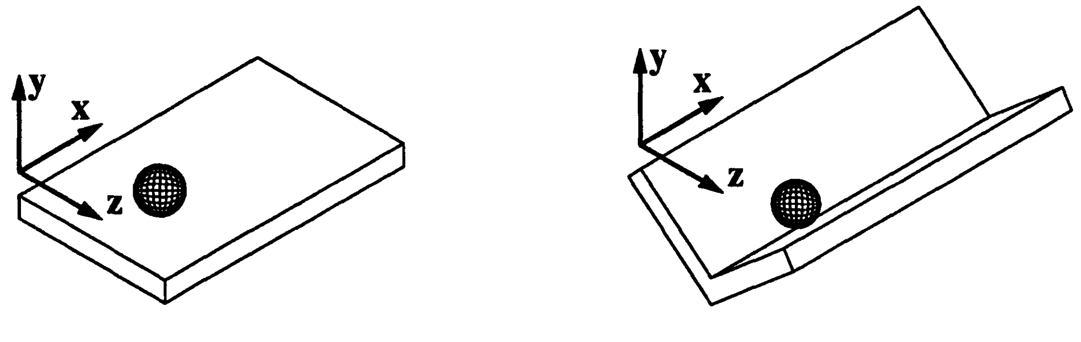
The obstacle free region identified from human inconsistency provides a number of advantages: obstacle avoidance can be guaranteed without knowledge of the obstacle locations, a buffer can used to avoid obstacles in the presence of robot error, unnecessary vibrations can be removed from the robot motion, the human does not feel pressured to perform the 'perfect' demonstration, and robot performance can be higher than any of the demonstrations in terms of the distance traveled.

1.5.2. Constrained Motion and Adaptation To Workpiece Misalignment

In a contact task, the motion of the part becomes constrained by the workpiece surfaces. It is necessary for the manipulator (human or robot) to adapt to the environment. Otherwise, any misalignment of the workpiece could result in improper assembly and excessive contact forces that could damage the parts. The workpiece surfaces in contact with the part define the contact state, and to simplify the analysis each contact state is considered separately. In Chapter 3, a robot controller is specified for motion within a single contact state that can adapt to workpiece misalignment. To avoid excessive contact forces, a robot compliance controller is used, in which the robot reacts with the environment like a spring. If the robot stiffness is too high excessive contact forces could occur, yet if the stiffness is to low excessive position errors could occur due to disturbances from workpiece misalignment and friction. The analysis uses the demonstration data to identify a range of robot compliance appropriate for the contact state.

The approach presented for obstacle avoidance, which uses human inconsistency to identify a range of acceptable motion, is extended in Chapter 3 to constrained motion. Human inconsistency is used to define a range of acceptable constrained motion, force magnitudes, and force directions.

For the case of 3D translation there exists two possible configurations for constrained motion, as shown in Figure 1.4: a single direction of constraint and two directions of constraint. When the part is constrained in three directions, no translation is possible.



a. Single direction of constraint

b. Two directions of constraint

Figure 1.4: Two Possible Constraint Configurations for 3D Translation

When a part is constrained with a single constraint surface, there remains two direction of admissible motion tangential to the surface. Accordingly, the position trajectory on the constraint surface may vary between the demonstrations as shown in Figure 1.5. Here obstacles on the surface include any geometric location that would result in an undesirable change in the contact state. In Figure 1.5 the demonstrated trajectories circumvent a "hole" obstacle to avoid dropping off the surface. The demonstrated trajectories define a region of obstacle free motion on the workpiece surface similar to the one identified for unconstrained motion. However, here the obstacle free region is relative to the workpiece surface, whose location varies between demonstrations.

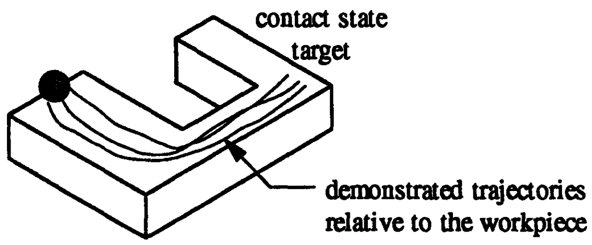


Figure 1.5: Demonstrated Trajectories On a Single Constraint
The demonstrations define an obstacle free region relative to the workpiece surface.

If the sphere traced its path along the workpiece, then the obstacle free region would be drawn on the workpiece as it is in Figure 1.5. However, the human motion is measured by the teaching gripper in an absolute coordinate system, and the actual workpiece location is not known since it varies between demonstrations. Accordingly, a method is presented in Chapter 3 to identify an obstacle free region relative to the workpiece surface.

In a contact state with two constraints there exists only a single degree of freedom in the direction of motion, as shown in Figure 1.4. Accordingly the direction of motion is completely defined by the workpiece. Since there is no error in the direction of motion, obstacle avoidance is not an issue. However, to successfully implement motion with two constraints it is necessary to maintain contact with both surfaces. Here there are two degrees of freedom in direction on the constraint force. As long as the force remains in the region of acceptable force directions, as shown in Figure 1.6, the desired contact state will be maintained.

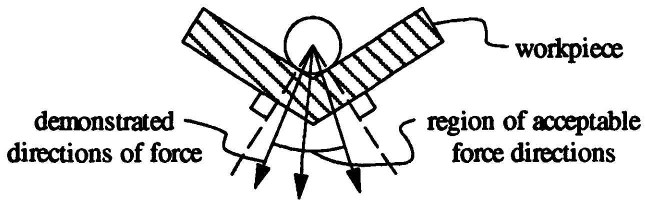

workpiece
demonstrated directions of force
region of acceptable force directions

Figure 1.6: Cross Section of Two Constraint Configuration
The demonstrated forces identify a region of acceptable force directions that ensure contact with both surfaces is maintained

As the demonstrator performs motion constrained by two surfaces, the direction of force can vary with motion along the surfaces and between demonstrations, as indicated by the force vectors in Figure 1.6. Nevertheless, the human successfully maintains contact with both surfaces for the range of applied forces, and thus human variation in force direction is not necessary for task performance and corresponds to human inconsistency. In Chapter 3, a region of acceptable force directions is identified from the demonstration data, and a robot controller is specified that maintains the constraint force to be within this region. A range of acceptable force magnitudes is also identified, and the maximum robot force is limited to be less than or equal to the maximum demonstrated force.

Variations between demonstrated trajectories in contact tasks do not all correspond to human inconsistency; rather some variation corresponds to adaptation to workpiece misalignment. While the human is adapting, their motion also contains some unnecessary motion. Accordingly, demonstrations of contact tasks contain both human adaptation and inconsistency. To implement PHD and generate a robot program, human adaptation is distinguished from inconsistency. Identifying human adaptation enables us to develop a robot controller that can also adapt to the environment. Furthermore, identifying human inconsistency provides information regarding the accuracy requirements for the task, as in the case of obstacle avoidance.

In the case of 3D translation, human adaptation and inconsistency can occur in the same contact state. An example is presented in Figure 1.7, where trajectories from two demonstrations are shown. If the demonstrator was "perfectly consistent", then only adaptation would occur as shown in Figure 1.7a and the shape of the two trajectories would be the same. With a "perfectly consistent" demonstrator the only modification between trajectories would correspond to misalignment in workpiece orientation to enable the trajectories to maintain contact the surface. In actuality inconsistency occurs together with adaptation, as shown in Figure 1.7b, and the shape of the trajectory varies in directions other than necessary to maintain contact.

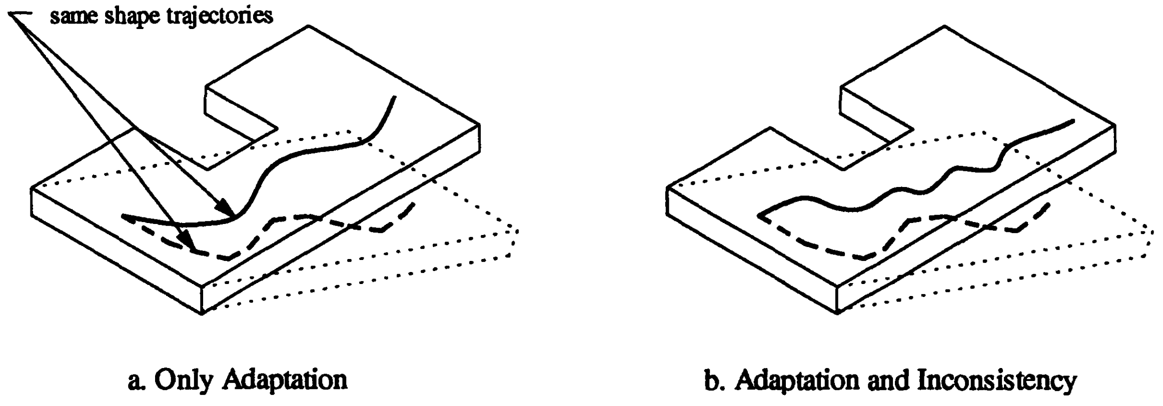


Figure 1.7: 3D Adaptation and Inconsistency
Trajectories from two demonstrations are shown, where the dashed lines correspond to the misaligned workpiece. If the demonstrator was "perfectly consistent", then only adaptation would occur as shown in 'a'. In actuality both adaptation and inconsistency occur as show in 'b'.

In Chapter 3 a method is presented to distinguish between human adaptation and inconsistency. It is not possible to directly measure human intent. However, the demonstrator can only adapt to workpiece misalignment that they can detect. Accordingly, the extent of workpiece misalignment that is detectable from force and position information is identified. Human motion that geometrically corresponds to detectable workpiece misalignment is interpreted as human adaptation; the remaining human variation is interpreted as inconsistency. For motion within a single contact state, adaptation is necessary to maintain contact with the workpiece surfaces, and is implemented by modifying the part trajectory in response to surface orientation misalignments.

The analysis of human adaptation and inconsistency is combined to specify a robot compliance controller for each contact state. The robot performance is quantifies in terms of position and force errors, caused by workpiece misalignment and friction. The robot controller is specified so that it can adapt as well as the human, while its position and force trajectories remain within the region of acceptable motion and force identified from human inconsistency. The complete robot controller, including both trajectory and compliance, is specified from the demonstration data without using a geometric model of the task.

1.5.3. Segmentation of Task Into Subtasks

Segmenting a task into a sequence of subtasks allows the task to divided into simpler components, and each subtask can then be implemented with a simpler controller. Since a model of the task is not available in PHD, it is necessary to identify the subtasks from the demonstration trajectories. Segmenting the demonstration data into meaningful subtasks is an essential component of the analysis. The preceding method of identifying human inconsistency and adaptation is based on comparing demonstrations at corresponding contact states. If the segmentation algorithm does not segment each demonstration at corresponding points, then the comparisons between the demonstrations are invalid.

The criterion for valid segmentation, is that a robot controller can be specified that can perform each subtask. The analysis in Chapter 3 identifies a robot controller for a section of motion in which the part is constrained by the same workpiece surfaces, i.e. motion within a single contact state. Accordingly, a natural approach is to segment the task at each contact state transition. A segmentation algorithm is presented in Chapter 4 which identifies contact state transitions when possible.

A difficulty with the contact state segmentation approach is that not all contact state transitions can be detected from the demonstration data. However, it is shown in Chapter 4 that undetectable contact state transitions do not need to be detected. Any segment of motion in which the sensory measurements provide no new information from the environment can be implemented as a single subtask. In most cases, information regarding workpiece location can be detected upon contact with a new workpiece surface. However, in all undetectable contact state transitions, no new information can be acquired by contact with the second contact state. Accordingly, the robot controller identified for a single contact state will be able to perform the motion for both contact states, when the transition is undetectable.

Other research approaches have been used to segmented human demonstration data. These methods have relied upon assumptions of how the human performs the task; for example that the demonstrator pauses between subtasks. However, the demonstrator may pause at different points in the task for no apparent reason, and thus segmenting at each pause will not yield consistent results. An advantage of the segmentation algorithm presented in Chapter 4 is that it does not rely on when the human pauses or on a model of the internal method the human uses to control their motion. Instead subtasks are defined at points where new information can be acquired from the force and position measurements.

1.5.4. Subtask Termination Conditions

Subtask termination conditions are sensor measurements that indicate to the robot when to switch from one subtask to the next, and are an important component of the adaptation strategy. In model based robot programming, it is necessary to predict the appropriate termination conditions. However, with the PHD approach the sensor signals are directly available. Accordingly, sensor measurements that indicate completion of a subtask can be extracted from the demonstration data. In Chapter 5 a method is presented for identifying the termination conditions, by correlating the completion of subtasks with sensor measurements.

For example, consider the contact state transition shown in Figure 1.8. When the part moves from contact with a single surface to motion constrained by both surfaces, an increase in the force in the negative 'x' direction occurs.

This change in force occurs in all demonstrations, and therefore can be identified as a subtask termination condition. The robot uses this signal to detect the change in contact, and begin motion in the 'y' direction.

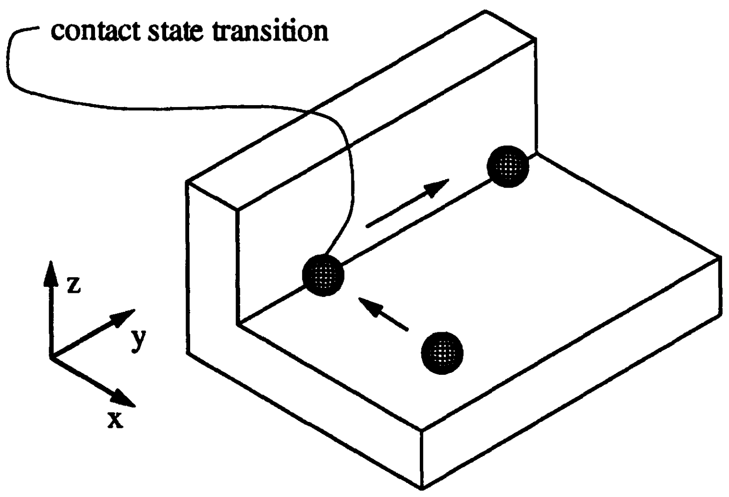


Figure 1.8: Contact State Transition

The robot trajectory for a new subtask is generated relative to the completion point of the prior subtask, which is identified by the termination condition. In this fashion, the robot adapts to workpiece translation misalignment. Translation of a workpiece surface changes the position at which a contact state transition occurs, and by detecting this change the robot trajectory is modified accordingly. However, workpiece orientation misalignment is not detected from the position of the part at the beginning of a subtask. Accordingly, the robot compliance controller identified in Chapter 3, is used to adapt to orientation misalignments.

There are two types of subtask termination conditions. The first type corresponds to a detected change in contact state, and the second type corresponds to reaching a desired target position within a contact state. In Chapter 5, potential termination conditions of the first type are defined in terms of changes in the direction of motion or force. For a potential termination condition to be a valid termination condition, its value should reach a threshold level at the completion of the subtask, but not beforehand. To evaluate potential termination conditions, a delectability criterion is identified, and the termination condition with the highest delectability is selected. If none of the termination conditions of the first type are valid, then the termination condition is of the second type. In this case, the desired target position within the contact state is defined from the average position from all the demonstrations. Both types of termination condition are present in the example shown in Section 1.6.

An additional issue associated with identifying termination conditions from human demonstration data, is that the potential exists for confusion between cause and effect. In the contact state transition shown in Figure 1.8, the proper termination condition corresponds to an increase of the force in the 'x' direction as the part contacts a new surface. After the demonstrator detects the transition, they begin to move along the next contact state, and thus generate both motion and force (to overcome friction) in the 'y' direction. The force in the 'x' direction is the proper termination condition, which *causes* the demonstrator to switch to the next subtask. On the other hand, the force and motion in the 'y' direction correspond to the beginning of motion in the following subtask, and are the *effect* of the demonstrator switching subtasks. Sensor measurements that correspond to both cause and effect occur within a short period of time at the completion of a subtask, and it is necessary to ensure that the termination conditions used by the robot corresponds to the cause of the transition.

The approach presented in Chapter 5 to ensure proper causality, is to identify the point in time when the demonstrator changes their action to begin the next subtask. The proper termination condition will always occur prior to the change in human action. A period of time is defined as the termination region, which begins just prior to the completion of a subtask and ends at the time at which the human changes their action in response to subtask completion. Figure 1.9 shows the force in the 'x' and 'y' direction, at a contact state transition corresponding to the one shown in Figure 1.8. The change in f_x occurs within the termination region and thus can be identified as a valid termination condition. However, the change of f_y, which corresponds to the beginning of motion in the following subtask, occurs outside of the termination region, and thus will not be identified as a termination condition.

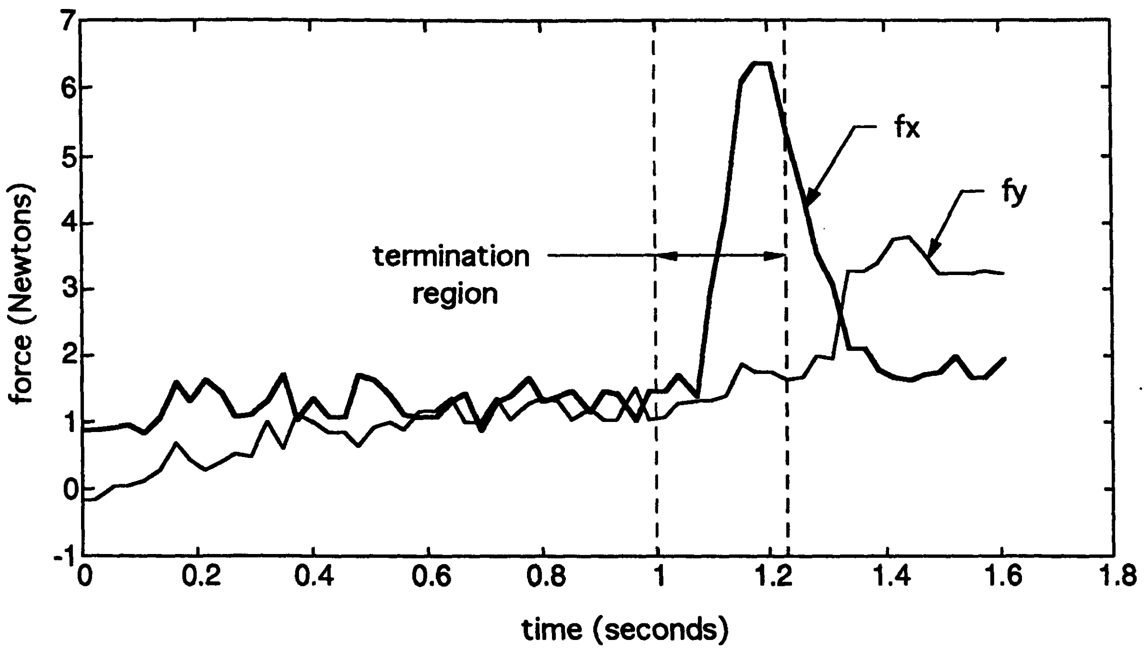


Figure 1.9: Termination Conditions

1.6. Experimental Application

An example contact task, shown in Figure 1.10, is used to illustrate the analysis in Chapters 3, 4, and 5. In each of these chapters, experimental results corresponding to the analysis are presented.

The contact state sequence the demonstrator used to implement the task, is shown by the numbered contact state transitions in Figure 1.10. This sequence of contact states allows the demonstrator to successfully implement the task with their eyes closed, despite significant workpiece misalignment. The demonstrator can reach the target position, as long as workpiece misalignment does not cause the demonstrator to miss the initial contact state, and the orientation misalignment is not excessive.

The example task contains a number of features useful for illustrating the analysis. Previous research in using human demonstrations for contact tasks [Asada and Izumi 1987], restricted workpiece geometry to contain perpendicular surfaces, and allowed only contact state transitions that would increase the number of constraints. In the example shown, a workpiece surface is not perpendicular to the other ones, and as the task progresses the number of constraints increases and decreases.

In addition successful robot implementation requires specific levels of position and force accuracy. In the fifth contact state, the position accuracy must be high enough to avoid the triangular holes on the constraint surface. In the third contact state, the part is constrained by two surfaces with an obtuse angle between them, which requires a higher level of accuracy in force direction that in the case of perpendicular constraint surfaces.

Another aspect of the example is that it contains both types of termination conditions. Contact states from one to three are of the first type, where the completion of the subtask is detected by contacting a new surface. However, the fourth termination condition is of the second type. Through familiarity with the workpiece, the demonstrator knows that to continue in the fourth subtask too long will result in the part falling into the triangular hole in the surface. Accordingly, the subtask is completed when the motion has progressed a specified distance within the contact state.

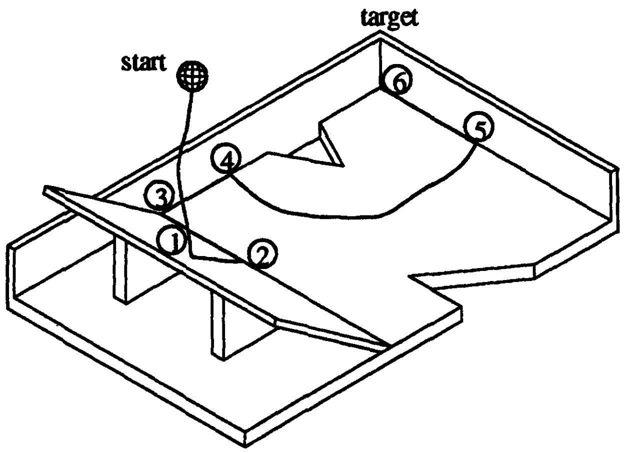


Figure 1.10: Workpiece Used in the Experiments

1.7. Summary

The method of Programming by Human Demonstration provides an alternative to traditional programming methods. The advantage of PHD is its easy of use. Issues that are difficult in model based programming methods such as obstacle avoidance and selecting an appropriate contact state sequence, are solved intuitively by the human. In addition, a model of the task geometry is not required.

One of the contributions of this thesis is that a new approach is presented for interpreting human data. Inconsistent human motion and forces are not treated as an undesirable phenomena, but are used to an advantage. Robot accuracy requirements are identified from the range of human inconsistency. The advantageous use of human inconsistency has the additional benefit that it further increases the ease of programming, since the demonstrator does not have to worry if the quality of their demonstration is not perfect.

Another contribution is that human demonstrations are used to generate robot programs that can adapt to variations in the environment. The strategy that the human uses to adapt to the environment cannot be directly measured. However, the demonstrator can only adapt to workpiece misalignment that they can detect. Accordingly, the extent of workpiece misalignment that is detectable from force and position information is quantified, and human action that corresponds to this information is identified as human adaptation. Robot compliance controllers are specified that adapt to orientation misalignments within a contact state, and thereby maintain contact with the workpiece surfaces. Subtask termination conditions are identified that detect contact state transitions, and enable the robot to adapt to translation misalignment of the workpiece.

The following chapters present the details of the analysis, which are applied to different components of the task. However, the three following guidelines are present throughout the analysis:

- Variation between human demonstrations that does not correspond to detectable variation in the environment, is interpreted as human inconsistency. In a section of the task with a high level of human inconsistency, high robot accuracy is not required.
- Components of the demonstrations that are consistently present in all demonstrations, are interpreted as significant for task success, and are incorporated into the robot program.
- Adaptation can only occur in response to detectable variation in the environment.

The objective of the analysis in the following chapters is to identify a complete robot program, including the details necessary for force and position control. Accordingly, a robot program including trajectory, compliance, and termination conditions are generated from demonstration data. The robot program can successfully perform the task as long as variations in the environment are not larger than those encountered during the demonstrations.

CHAPTER 2

Unconstrained Motion

And

Obstacle Avoidance

2.1. Introduction

This chapter addresses pick and place tasks in which a part is moved from a starting position to a target position without contacting other objects in the environment, as shown in Figure 2.2. Objects in the environment are referred to as obstacles, and the motion is referred to as unconstrained since there is no contact with the environment that would restrict the motion. In this chapter it is assumed that the obstacle location remains fixed, and thus no adaptation to the environment is necessary. Regardless, the lack of contact between the part and the environment precludes any adaptation to the environment using force sensing.

Obstacle avoidance is a well studied and often difficult problem in robotics [Latombe, 1991; and Lozano-Pérez, Jones, Mazer , and O'Donnell, 1992]. When a model of the environment is used, the robot trajectory is generated off-line. The number of potential trajectories can be very large, and selecting an appropriate robot path may require a computational expensive search. For tasks requiring low clearance between the robot and the obstacles, the level of detail necessary in the model increases along with the computational cost.

The PHD approach addresses the obstacle avoidance problem by letting the human identify an obstacle free path, and then generating the robot trajectory from the demonstration data. The demonstration identifies an obstacle free path for the part and the gripper, yet does not specify a trajectory for the robot arm. Analysis of interference between the robot arm and the environment depends on the robot configuration. However, as indicated by Lozano-Pérez et. al. [1992], most potential interference occurs with the part and gripper which by necessity are close to the other parts in the task. Accordingly, this thesis does not consider obstacle avoidance for the robot arm.

The most direct approach to generating a robot trajectory is to duplicate the demonstrated trajectory. As long as there are no variations in the environment, direct duplication of the trajectory will be successful. However, in practice it is inefficient to duplicate the human trajectory exactly. Human motion often contains motion that is unnecessary to achieve the task, including vibrations and "wiggles." Exact duplication of the human trajectory would result in unnecessary and robot motion.

One approach to improve the robot trajectory, is to approximate the demonstrated trajectory. Indeed the first approach investigated in this research was a piecewise linear approximation, of which an example is shown in Figure 2.1. Appendix II presents the details of this method which relies on a statistical hypothesis test to determine when to begin a new straight line segment. However, regardless of the approximation method used, there is a tradeoff between robot performance and the level of approximation. For example a given trajectory can be duplicated to a high level of accuracy with a large number of piecewise linear segments, yet the human "wiggles" will also be duplicated. If the approximation is less accurate, then one runs the risk of the robot trajectory hitting an obstacle. The only information available from a single demonstration is that the given trajectory is obstacle free. Therefore with any level of approximation, it is not possible to guarantee that the robot trajectory will be obstacle free. Furthermore, for different task and different demonstrators, the desirable level of approximation will vary. Accordingly, the approximation method is not satisfactory.

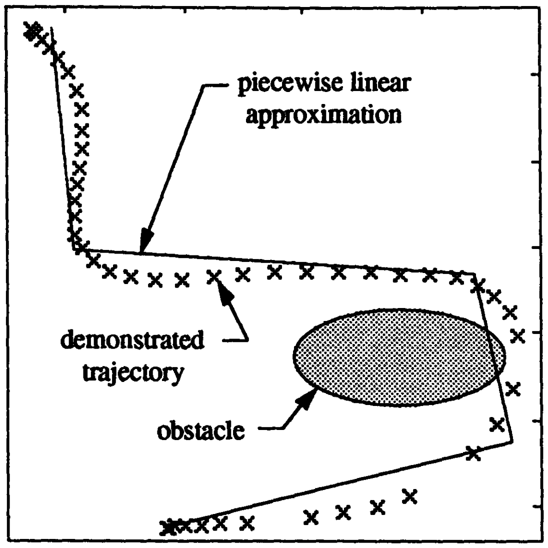


Figure 2.1. Piecewise Linear Approximation
Approximating the demonstrated trajectory can remove unnecessary motion, yet could also cause the robot to hit an obstacle.

An alternative method of generating robot trajectories from human demonstration data is presented by Ogata and Takahashi [1993]. Here a model of the environment is used, and the workspace is segmented into convex obstacle free regions. The human demonstration is used to identify a sequence of regions from which the robot trajectory is generated. Since our objective is to generate the robot program exclusively from the demonstration data, such model based methods in which the obstacle location is known are not appropriate.

2.1.1. Multiple Demonstrations Approach

The method proposed in this chapter is to perform multiple demonstrations of the same task to capture a range of obstacle free human motion. An example of an obstacle avoidance task is shown in Figure 2.2. If all the demonstrations pass each obstacle on the same side, then the region between the demonstrations is obstacle free. A robot trajectory that stays within this region is guaranteed to be obstacle free. Using the multiple demonstration method, a trade off is not necessary between ensuring obstacle avoidance and efficient robot motion.

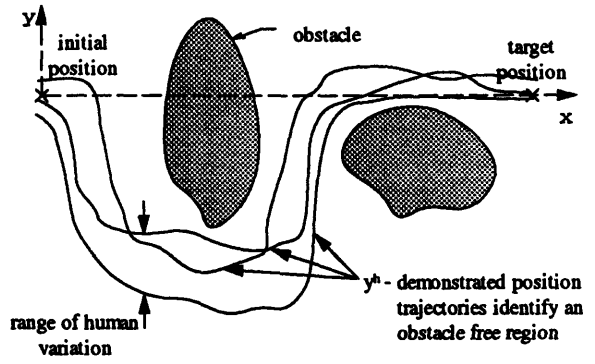

y
initial position
obstacle
target position
x
y^h - demonstrated position trajectories identify an obstacle free region
range of human variation

Figure 2.2: Multiple demonstrations. Simple case $y = f(x)$
The human demonstrates the task multiple times, and the region between the demonstrations defines an obstacle free region, without the use of a model indicating obstacle locations. A robot trajectory within this region is guaranteed to be obstacle free

Since the environment is the same in all demonstrations, i.e. obstacle locations do not change, variations between demonstrations are interpreted as human inconsistency and correspond to unnecessary motion. Here inconsistency is not interpreted as noise that should be eliminated from the analysis, but provides information regarding the task requirements. The amount of human variation indicates the desired accuracy of the task, as shown by the wide and narrow regions in Figure 2.2. Motion that is necessary to achieve that task is present in all the demonstrations, such as the curved motion between the two obstacles in Figure 2.2, and is incorporated into the robot program. Human inconsistency identifies an obstacle free region between the demonstrations. Motion that does not effect task performance corresponds to variation within the obstacle free region, and need not be included in the robot trajectory.

The example shown in Figure 2.2 is referred to as a simple case, and it is straightforward to identify an obstacle free robot trajectory. A coordinate system is defined with the "x" axis passing through the initial and target positions. This example is simple since all the trajectories are a function of the "x" axis, and the location on the "y" axis defines whether a point is above or below an obstacle. The demonstrator is requested to perform the task a number of times, while avoiding the obstacles in a consistent fashion, meaning they should always pass above or always pass below each obstacle. The demonstrated trajectory is given by $y_i(x)$, where the subscript "i" indicates the demonstration number. For each point along the "x" axis, the maximum and minimum of the human trajectory identify an obstacle free range. A robot trajectory, $y^r(x)$, is guaranteed to avoid obstacles as long as it stays within this range, which is indicated by:

$$\min_{\forall i}\{y_i(x)\} \leq y^r(x) \leq \max_{\forall i}\{y_i(x)\} \qquad (2.1)$$

A specific robot trajectory is not specified at this point. Instead the analysis is extended to include more general motion, and then a robot path is defined. It should be noted that no attempt is made to transfer the velocity profile of the demonstrator to the robot, since obstacle avoidance does not rely on a specific velocity of motion.

2.1.2. General Translation Overview

The example in Figure 2.2 is simple both because the demonstrated trajectories are a function of 'x', and because the motion is restricted to a plane. The general case of translation includes 3D motion, and looping in the trajectory prevents the trajectory from being a function of 'x' (in any coordinate system) as shown in Figure 2.3. The approach for identifying an obstacle free robot trajectory is extended to general translation in two stages.

In the following section general 2D translation is addressed. The field of topology is used to define the boundaries of the obstacle free region, since the boundaries cannot simply be defined by maximum and minimum values along the 'y' axis, as in Equation 2.1. In addition, the requirement that the demonstrator pass an obstacle on the same side in each demonstration, requires clarification. In general 2D translation, passing on the same side cannot be defined as simply above or below an obstacle. Once, the obstacle free region has been defined, a robot trajectory is selected that is the shortest path from the starting position to the target within the obstacle free region.

The analysis of 3D translation is presented in Appendix III. In 3D the boundary of a region is defined by a 2D surface, and thus cannot be composed of segments of 1D demonstrated trajectories. Accordingly, an alternative method is presented in which an obstacle free robot trajectory is identified without specifying the boundaries of the obstacle free region. The resulting robot trajectory is satisfactory but not necessarily the shortest.

2.2. 2D Translation

From the demonstrator's point of view the general case of 2D translation is very similar to the simple case shown in Figure 2.2. It is not difficult for a person to understand what is meant by passing an obstacle on the same side even in the general case examples shown in Figures 2.3 and 2.6. However, from a mathematical point of view, a more rigorously definition is required. Accordingly, this section presents a topological definition of the obstacle free region, and a set of criteria for passing an obstacle on the same side. Five assumptions are presented regarding the demonstrated trajectories and how they avoid the obstacles. In the following chapters, referring to passing an obstacle on the same side in each demonstration, implies that the assumptions presented in this section are met.

2.2.1. Jordan Curve Theorem

Our objective is to define the interior of an obstacle free region from a sequence of demonstrated trajectories. The fields of topology address the definition of interior and exterior of a region with the Jordan curve theorem [Munkres 1975]. The theorem and subsequent lemmas can be stated as follows:

Theorem 1: A Jordan curve is a simple closed curve in R^2 (a curve homeomorphic to the unit circle). The Jordan curve divides the space R^2 into two components, an interior and an exterior, and is the common boundary between them.

Definition 1: An arc is a space homeomorphic to the unit interval [0, 1], and thus cannot intersect itself.

Lemma 1: A Jordan curve can be defined by the union of two arcs that have precisely two points in common.

Lemma 2: A line switches from exterior to interior and vice versa every time it intersects the Jordan curve, provided that at the points of intersection, the line is not tangent to the Jordan curve or at a vertex of the curve.

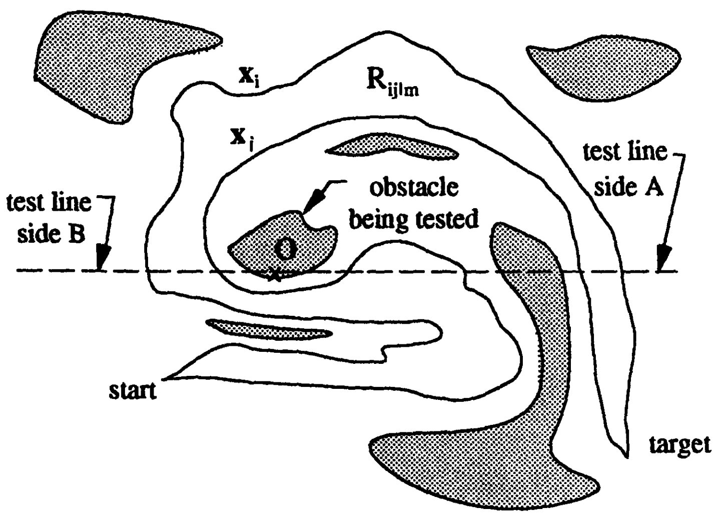


Figure 2.3 : Jordan curve
Segments from two demonstrated trajectories compose the Jordan curve, J_{ijlm}, and define the region R_{ijlm}. The test line shows that obstacle O is on the exterior R_{ijlm}. Both trajectories x_i and x_j have an odd number of intersections with side A of the test line indicating the same strategy of avoiding obstacle O.

An example of a region defined by a Jordan curve is shown in Figure 2.3, where the arcs $\mathbf{x}_i$ and $\mathbf{x}_j$ are segments of two demonstrated trajectories. For two trajectories to compose a Jordan curve, they must start and end at the same location. Furthermore, for a trajectory to be a valid arc it cannot intersect itself. To satisfy these conditions a number of assumptions are made.

Assumptions:

1. Each demonstrations is performed successfully by moving a part from the starting region to the target region while avoiding contact with obstacles.
2. The demonstrated trajectory is sampled fast enough so that linear interpolations between sample points are obstacle free. Thus a continuous trajectory is generated for the ith demonstration, $\mathbf{x}_i(s)$, and is a function of the distance traveled, "s."
3. To define a common starting and target position, straight line segments are appended to both ends of each trajectory from the average starting and ending positions. It is assumed that these added segments are also obstacle free. This assumption is always satisfied if the demonstrations begin and end in convex obstacle free regions.

For a segment of demonstrated trajectory to be an arc it cannot intersect itself. One would not expect a demonstrated trajectory to frequently intersect itself, since it would be inefficient for the human to return to a previous position in an obstacle avoidance task. Nevertheless, if looping does occur, the trajectory is modified by removing the loop. The modified trajectory, $\mathbf{x}'(s')$, does not intersect itself and is given by:

$$\mathbf{x}'_i(s') = \mathbf{x}_i(s) - \{\mathbf{x}_i(a<s<b)\ \forall\ \mathbf{x}_i(s=a) = \mathbf{x}_i(s=b),\ a<b\} \tag{2.2}$$

A pair of trajectories, $\mathbf{x}'_i$ and $\mathbf{x}'_j$, can be combined to define a Jordan curve or a series of Jordan curves. The trajectories always intersect at the start and target positions, and additional intersections generate additional Jordan curves. The intersection locations are designated by the series s_{ilm} and s_{jlm}, where $\mathbf{x}_i(s_{ilm})=\mathbf{x}_j(s_{jlm})$ and $m=1,2,\ldots M$. The series s_{ilm} is ordered according to the increasing magnitude of "s" in the ith trajectory.

<u>Assumption 4:</u> The intersections between and two trajectories "i" and "j" are in the same sequence. Specifically, one trajectory does not intersect another trajectory at a location prior to a previous intersection, which can be stated by:

$$\mathbf{x}'_i(s<s_{ilm}) \neq \mathbf{x}'_j(s>s_{jlm}) \qquad \text{for } m=1,2,\ldots M \tag{2.3}$$

Accordingly, the segments from trajectories $\mathbf{x}'_i$ and $\mathbf{x}'_j$ between intersections "m" and "m+1" designate two unique arcs which are combined to define the Jordan curve, J_{ijlm}:

$$J_{ijlm} = \mathbf{x}'_i(s_{ilm}<s<s_{ilm+1}) \cup \mathbf{x}'_j(s_{jlm}<s<s_{jlm+1}) \tag{2.4}$$

The region interior to the Jordan curve J_{ijlm} is designated by R_{ijlm}.

2.2.2. Passing an Obstacle on the Same Side

To ensure that the interior region of R_{ijlm} is obstacle free, it is necessary that the demonstrator uses the same strategy of avoiding obstacles in all the demonstrations. In the example shown in Figure 2.2 where $y=f(x)$, a consistent strategy is defined as always passing above or always passing below an obstacle. For the general case of 2D translation, a corresponding definition is required, which is developed with the use of a test line.

For each obstacle a test line can be constructed, as shown in Figure 2.3, to determine if any point on the obstacle is inside the region defined by the Jordan curve. The test line is a straight line of infinite length that passes through any point on the obstacle in question and is selected so that it does not intersect the Jordan curve at a vertex at a tangent. The test line is divided into sides A and B from a point on the obstacle, depicted by the cross mark in Figure 2.3. The number of times a trajectory $x'_i(s)$ intersects side A and side B of the test line is designated by T_{Ai} and T_{Bi}. A consistent strategy of avoiding obstacles is defined by assumption 5.

Assumption 5: If the value of T_{Ai} is odd for one demonstration then it is odd for all demonstrations, and if it is even for one demonstration then it is even for all demonstrations. Similarly for side B and T_{Bi}.

The proof that the interior of R_{ijlm} is obstacle free follows from lemma 2. At an infinite distance from the obstacle, side A of the test line is on the exterior of R_{ijlm}. As the test line approaches the obstacle and intersects the Jordan curve, it switches to the interior of R_{ijlm}, and then back to the exterior on the next intersection. If the obstacle is on the exterior of R_{ijlm}, then there will be an *even* number of intersections between the Jordan curve and side A of the test line. The number of intersection between J_{ijlm} and side A of the test line is designated by T_{Aij} and is the sum of the intersections T_{Ai} and T_{Aj}. As shown in the following equation, the value of T_{Aij} will always be *even* as long as assumption 5 is valid.

$$T_{Aij} = T_{Ai} + T_{Aj} = \left\{ \begin{matrix} \text{odd} + \text{odd} \\ \text{or} \\ \text{even} + \text{even} \end{matrix} \right\} = \text{even} \qquad (2.5)$$

Any point on the obstacle that is also on the test line is guaranteed to be on the exterior of the R_{ijlm}. It can also be shown that all other points on the same continuous obstacle are also exterior to R_{ijlm}. Given that point 1 on the obstacle is known to be exterior to R_{ijlm}, we assume that point 2 on the obstacle is interior to R_{ijlm}, and then show that this violates previous assumptions. In a continuous obstacle, points 1 and 2 can be connected with a line that stays within the obstacle, which is designated by the line O_{12}. Since the line O_{12} starts on the exterior and ends on the interior of R_{ijlm}, it will intersect the Jordan curve J_{ijlm} indicating that at the point of intersection J_{ijlm} is in the obstacle. However, according to assumptions 2 and 3, the trajectories that compose J_{ijlm} are obstacle free. Thus, line O_{12} cannot exist and point 2 along with all other points on the obstacle are exterior to R_{ijlm}.

The test line is an analytical tool used to prove that the region R_{ijlm} is obstacle free. However, in practice the demonstrator is not required to evaluate their trajectories relative to test lines or be familiar with the details of the preceding proof. Assumption 5 is indeed quite similar to the intuitive assumption used for the simple case shown in Figure 2.2, of always passing above or always passing below a given obstacle. Typically it is sufficient to request of the demonstrator to "to use the same strategy of avoiding obstacles in all demonstrations" These instructions are intuitive, and in our experience the resulting demonstrations do not violate any of the assumptions.

2.2.3. Constructing the Obstacle Free Region

The preceding proof shows that any individual Jordan curve defined from segments of demonstration trajectories has an obstacle free interior. To show that the union of all Jordan curves generated from two demonstrations are also obstacle free, a final assumption is required.

Assumption 6: Once a trajectory forms a Jordan curve by intersecting with another trajectory, it does not re-enter the region of that Jordan curve. Specifically,

$$x_i(s>s_{ilm}) \cap R_{ijlm} = \varnothing \tag{2.6}$$

The proof that the region defined by a single Jordan curve is obstacle free is based on the fact that the test line switches from interior to exterior at every intersection with the curve. If the regions from two Jordan curves overlap, then an intersection with a test line may occur that simply switches from the interior of one Jordan curve to the interior of the other one. However, assumption 6 eliminates this possibility. Thus the sequence of regions R_{ijlm} for m=1,2,...M are guaranteed to be obstacle free using the same argument presented for an individual region.

The obstacle free regions identified from one pair of demonstrations can be determined by finding the intersections between the trajectories. Combining the obstacle free regions from all possible pair combinations result in the obstacle free region, R_F. The procedure used to define the boundaries of R_F is presented in the following steps with references to figure 2.4. Implementation requires finding the intersection points between trajectories, which is implemented using the sweep line algorithm [Latombe 1991].

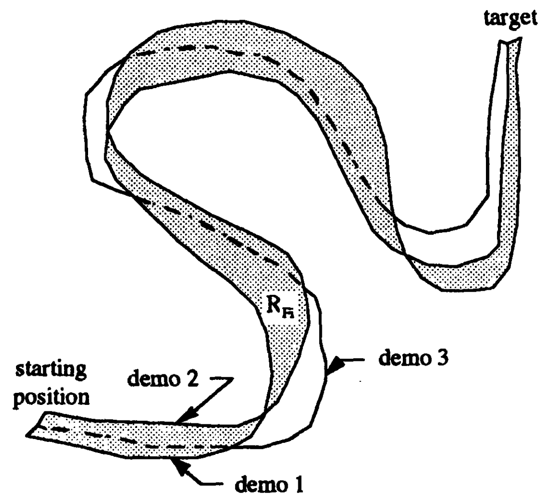


Figure 2.4: Identifying an Obstacle Free Region
The first two demonstrations define the initial region R_{Fi}. Additional demonstrations extend the boundaries with the segments that are exterior to R_{Fi} as shown by the solid segments of demonstration 3.

- Average the beginning and ending position of each trajectory to find the starting and target positions. Append segments to each trajectory so they all start and end at common positions.
- Find the intersections within the individual trajectories and remove the loops as explained in equation 2 to define the modified trajectories $\mathbf{x}'_i$.
- Use the first two demonstration trajectories, $\mathbf{x}'_1$ and $\mathbf{x}'_2$, to define initial boundaries of the obstacle free region, R_{Fi}. The boundaries of the region can be divided into a right and left side, which begin at the starting point and end at the target point. The interior of R_{Fi} is on the right of the left boundary and vice versa. If $\mathbf{x}'_1$ starts off as the right side, then at the first intersection between the trajectories $\mathbf{x}'_2$ switches to the right and $\mathbf{x}'_1$ switches to the left. A single test line is used to identify the right and left boundaries at one location. The complete right and left sides are defined by switching between the trajectories at every subsequent and preceding intersections..
- An additional demonstration can extend the boundaries of the obstacle free region with segments that are external to R_{Fi}, as shown in figure 2.4. The intersections between the new trajectory and the right and left boundaries of R_{Fi} are found. The new trajectory switches between interior and exterior of R_{Fi} at every intersection with the boundaries. The new

trajectory starts on the interior of R_{Fi} if its first segment of the trajectory is on the interior of the vertex at the starting position. Thus, by keeping track of the intersections and the starting direction of the new trajectory, the segments that are exterior to R_{Fi} can be found. These segments define new boundaries of the obstacle free region. The region R_{Fi} is redefined to incorporate the new obstacle free region.

- Repeat the previous step for all the trajectories. Define the final region R_{Fi} as the obstacle free region R_F.

2.2.4. Shortest robot path

To generate a robot program it is necessary to determine a path within the obstacle free region identified by the demonstrated trajectories, from the starting position to the target. A number of methods have been developed to generate trajectories given a model indicating obstacle location. A comprehensive review of these methods is presented by Latombe [1991]. The objective of some methods is to identify the shortest path, while others identify the shortest time trajectory by incorporating robot dynamics, or maximize the clearance between the robot and the obstacles. In this section a method is presented to generate the shortest path, which has been specifically adapted for use with PHD. However, the advantage of the approach presented in this paper is that the demonstrations define a range of obstacle free trajectories. Determining the shortest path represents a single use of this region. Depending on one's application one may choose to apply other methods for selecting a path within the obstacle free region.

The obstacle free region generated from the demonstrations is a polygon, since the trajectories are generated from linear interpolation of sampled data points. The problem of identifying a shortest path for 2D translation with polygonal obstacles has been solved using the Visibility Graph method [Latombe 1991]. This method can also be applied to our case by simply designating the exterior of the obstacle free polygon as a virtual polygonal obstacle.

The Visibility Graph method is based on the fact that the shortest path must also be locally the shortest. Accordingly, at any location at which the curvature of the shortest path is not zero, an obstacle must exist on the concave side of the curved trajectory. Otherwise, a locally shorter path could be generated with a straight line segment. A result of this reasoning is that for polygonal obstacles the shortest path is a piecewise linear path, and the edges in the path all occur at vertices on the obstacles. Since there are a finite number of vertices on a set of polygonal obstacles it is possible to search all combinations to identify the shortest path.

An alternate method for identifying the shortest robot path is presented here that avoids the need for an exhaustive search. This method takes advantage of the fact that the obstacle free region is a single polygon in which both the starting and target points are vertices of the polygon. The procedure for finding the shortest path is presented and then the proof is provided that the resulting path is indeed the shortest.

The following steps present the algorithm for determining the shortest path within the polygon region R_F, and an example is shown in Figure 2.5.

1. Define the start and target positions, $\mathbf{p}_{start}$ and $\mathbf{p}_{target}$, as the average of the demonstration starting and ending positions. Set the beginning position of the current segment, $\mathbf{p}_{beg}$, to $\mathbf{p}_{start}$. Divide the polygon border into two at the starting and target points, defining a right and left side.
2. Guess an initial direction of a straight line that departs the starting point.
3. Evaluate the proposed straight line segment, and find which side of the boundary it intersects first (for example the right side in Figure 2.5).
4. Propose a new line, by either incrementing or decrementing the slope of the previous line in the direction towards the opposite side found in step 3. Evaluate this line and continue modifying the slope until a line is found that is limited by the side opposite that of step 3. Eventually both "right" and "left" lines are found.
5. Select a new line between the right and left by dividing the angle between them. Evaluate this line, and replace the previous "right" or "left" line according to the side of the new line.
6. Repeat the previous step until both the right and left lines converge, which is indicated when the angle between them is less than a desired accuracy threshold. A robot path segment is defined in the direction of convergence with a length of the shorter of the right and left lines.

7. Define a new beginning point, p_{beg}, at the end of the previous robot segment. Repeat the above steps to find the next robot segment, starting at step 2. The shortest path is completed when the target can be reached with an obstacle free straight line from the starting point of the previous segment.

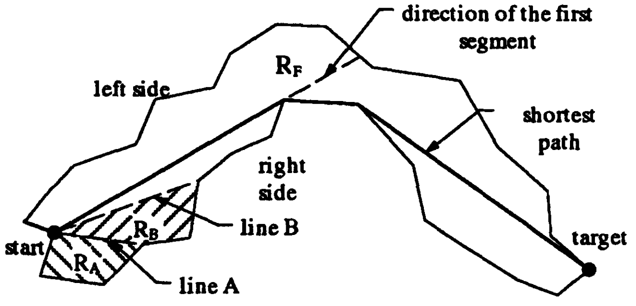


Figure 2.5: Shortest Robot Trajectory
The direction of each segment of the shortest path is selected so that if it was extended within the obstacle free region, R_F, it would intersect both the right and left side. All other directions are not locally the shortest.

The proof that the preceding procedure identifies the shortest valid robot path is based on the fact that the shortest path must at all points also be locally the shortest. Consider a potential first segment that intersects the boundary of the obstacle free region, R_F, on the right side, represented by line A in Figure 2.5. Region R_A is defined by line A and a section of the right boundary of R_F. Since the target is outside R_A, a path to the target that includes line A must exit region R_A. However, a line from region R_A to the target will intersect line B. Between this intersection point and the starting position the shortest line is along line B, and thus line A is not locally the shortest. In general, for any line whose interior segment intercepts only the right or left side, there will exist another line that is locally shorter.

The procedure for converging on a line that intersects both the right and left side identifies the only direction that is not excluded from being the shortest path. The length of the shortest path segment is selected so that it ends at whatever side of the polygon it intersects first, since once the path intersects the boundary of the polygon it is a potential edge point on the shortest path. Accordingly, a new line segment is started at that point and the procedure to identify the direction of the shortest path is repeated.

2.2.5. Buffer Between Robot and Obstacles

An alternative to identifying the shortest path in the obstacle free region, is to identify a path that contains a buffer between the robot and boundary of the obstacle free region. The buffer ensures obstacle avoidance, even in the presence of robot error, as long as the robot error is less than the buffer. Of course the buffer cannot be wider than the narrowest part of the obstacle free region.

One method to implement a buffer would be to move the right and left boundaries toward each other; thereby defining a smaller obstacle free region. The shortest robot path could then be defined within the smaller obstacle free region. A more convenient method for implementing a buffer is presented in Appendix III.

With either method, the width of the obstacle free region reduces to zero at the start and target points. However, these points were defined from an average of demonstration trajectories. Therefore, a robot error is permissible at the start and end of the trajectory as long as it does not exceed the variation at the beginning and end of the demonstrations.

2.3. Experimental Results

An example planar translational task was performed, which consisted of moving a sphere between obstacles as shown in Figure 2.6. The task was demonstrated using the teaching interface shown in Figure 1.1, for a total of ten demonstrations. The demonstrator was asked to perform the task in a consistent fashion by passing all obstacles on the same side, yet not to worry about unnecessary motion. The demonstrator performs the task with their eyes open, since no adaptation to the environment is possible. The part was grasped continually throughout the task.

Details of the design of the teaching gripper is presented by Pinkney[1993]. To ensure that both the part and the robot gripper avoided obstacles, the teaching gripper was designed with the same shape gripping surfaces and approximately the same overall shape as the robot gripper. The position of the teaching gripper was measured with an ultrasonic sensor developed by the Logitech Corporation, which allows relatively unencumbered human motion. The sensor has three stationary speakers which transmit to three receivers mounted on the teaching gripper. Through triangulation the position and orientation of the gripper are measured. The accuracy of the sensor claimed by the manufacture is 2% of the distance between the transmitter and the receivers, which for the example task consisted of ±2mm. A coordinate system transformation was performed from the position sensor to the center of the sphere, using the sensor orientation information. Each demonstration trajectory, $x_i(s)$ is defined in terms of motion at the sphere center.

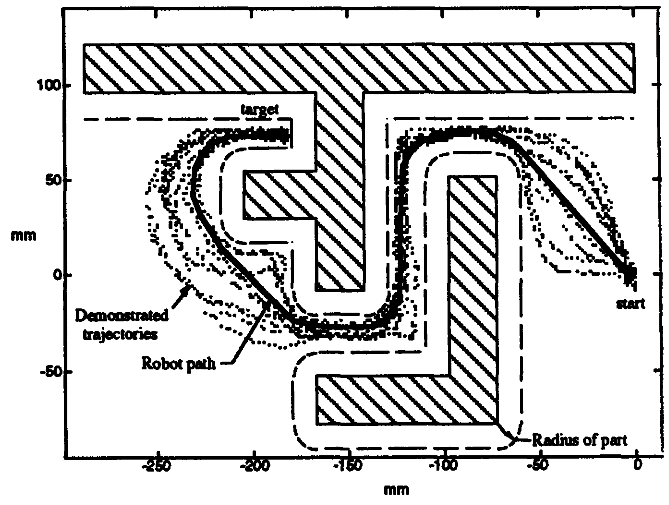


Figure 2.6 : 2D Experimental Results
The range of human inconsistency is used to improve robot performance by generating a robot trajectory is shorter than any of the demonstrations and that has a buffer with the obstacles.

The technique for finding the shortest path presented in the previous section was used with a buffer of 2.5mm. The average distance of the human demonstrations was 547mm and the minimum was 507mm. The robot trajectory, shown in Figure 2.6, has a distance of 449mm. Thus, the robot travel distance was reduced by almost 20% relative to the average demonstration and by over 10% from the shortest demonstration. Furthermore, the robot trajectory does not contain the unnecessary motion or "wiggles" present in the human motion.

The task was implement by a Mitsubishi Movemaster robot, shown in Figure 2.7, without contacting the obstacles. The robot has five degrees of freedom, and is controlled by specifying a sequence of end effector positions in the robot coordinate system. To perform the coordinate system transformation between the robot and the teaching gripper, an additional position receiver was mounted on the robot gripper. An automatic calibration procedure was

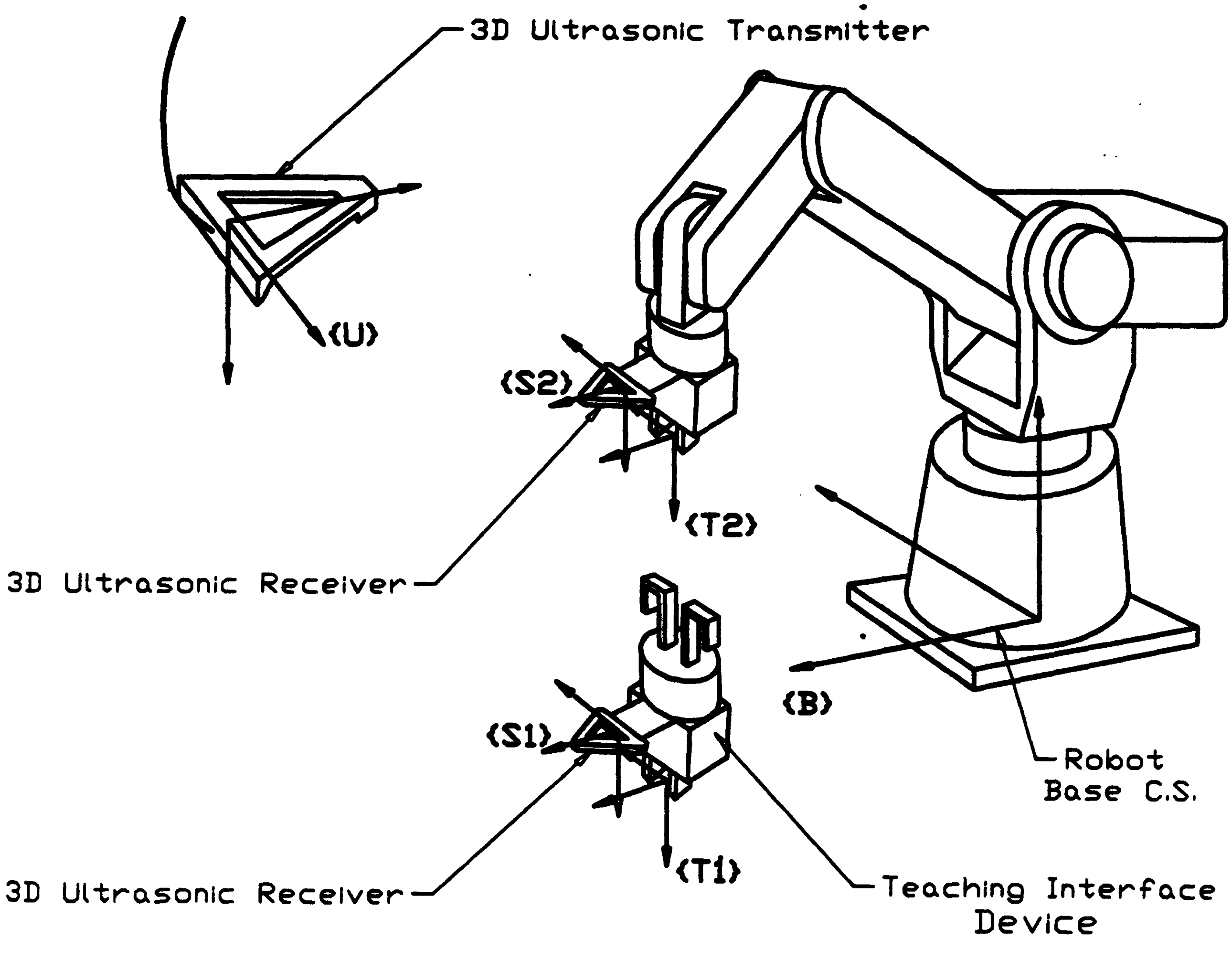


Figure 2.7: Mitsubishi Movemaster Robot and Ultrasonic Position Sensor
An ultrasonic receiver measures the position (translation and orientation) of the teaching gripper and robot gripper. The position of the robot gripper is also measured by sensors at the robot joints. The redundant position measurement allows automatic calibration between the robot and ultrasonic sensor coordinate system. (This figure has been provided by Cheng-Jung Chiu.)

developed by Cheng-Jung Chiu [1994] which moved the robot through a series of positions defined in the robot coordinate system, and measured the same positions in the coordinate system of the ultrasonic position sensor. The robot trajectory shown in Figure 2.6, was specified by defining the position at the beginning and end of each straight line segment. The robot controller generated the trajectory of each straight line segment by passing thorough the beginning, and ending points, as well as two intermediate points. The repeatability of the robot according to the manufacture is ±0.1mm.

2.3.1. Discussion

The presence of inconsistent human variation is used to improve robot performance. The robot trajectory in Figure 2.6 is shorter than any of the demonstrated trajectories and does not contain unnecessary motion or "wiggles" present in the human motion. As more demonstrations are performed, the size of the region identified as obstacle free increases, and the length of the robot path is decreased. Additional demonstrations correspond to an increase in information about the region of allowable motion. After a certain number of demonstrations have been performed, most additional demonstrations will remain inside the previously defined obstacle free region and not contribute new information about the task.

A limitation of the proposed method is that it does not consider interference between the robot arm and the environment. However, during our experiments we did not encounter any collisions or near collisions between the robot arm and the environment. Our experience concurs with the conclusions presented by Lozano-Pérez et al. [1992], which indicated that most potential collisions and analytical difficulties occur in close proximity to the part being manipulated and the gripper.

2.4. Conclusion

In this chapter pick and place tasks are addressed which do not require adaptation to the environment. A method is presented that generates a robot trajectory from observation of multiple human demonstrations. The robot trajectory is guaranteed to avoid obstacles, yet does not contain undesirable components of the human demonstrations such as "wiggles" or vibrations.

Since there is no variation in the environment, all human variation between demonstrations is interpreted as inconsistency. However, human inconsistency is not necessarily undesirable, and need not be eliminated from the analysis, but provides information regarding the task requirements. Human inconsistency is used to identify a region of acceptable robot motion. Components of motion that are present in all demonstrations results in a section where the obstacle free region is narrow, and thereby transfer that motion to the robot. However, a wide section of the obstacle free region indicates that human accuracy is not high, and therefore the robot accuracy does not need to be high in that section.

The region of obstacle free motion defined from human inconsistency, identifies conditions *sufficient* to guarantee robot success. It is not *necessary* that the robot remain in the region bounded by the demonstrated trajectories. Indeed, if one had a model of the obstacle locations one could identify alternate obstacle free paths to the target that are outside the demonstrated region. However, the PHD approach uses only information from the demonstration data, and therefore identifies conditions *sufficient* to guarantee robot success.

The obstacle free region identified from human inconsistency provides a number of advantages: obstacle avoidance can be guaranteed without knowledge of the obstacle locations, a buffer can used to avoid obstacles in the presence of robot error, unnecessary vibrations can be removed from the robot motion, the human does not feel pressured to perform the 'perfect' demonstration, and robot performance can be higher than any of the demonstrations in terms of the distance traveled.

CHAPTER 3

Constrained Motion And Adaptation To Workpiece Misalignment

3.1. Introduction

The goal of an assembly task is to place a part in a desired location relative to another part. Assembly usually consists of moving one part until it contacts the other part, and then continuing the motion while maintaining contact between the parts. For example when assembling a lid onto a box, the lid is brought into contact with the top of the box, and then aligned and pressed into place while contact with the box is maintained. The motion until contact is referred to as unconstrained motion, and moving a part while it is in contact with an object in the environment is referred to a constrained motion. In the previous chapter unconstrained motion was analyzed, and an obstacle free trajectory was identified. To complete the analysis, this chapter applies PHD to constrained motion. In constrained motion it is necessary for the manipulator (human or robot) to adapt to the environment. Otherwise, any misalignment of the parts could result in improper assembly and excessive contact forces that could damage the parts. In this chapter human demonstrations are used to identify a robot controller for constrained motion that can adapt to variations in the environment.

In the analysis of unconstrained motion, presented in Chapter 2, human inconsistency is used to identify a range of acceptable motion. This approach is extended in this chapter to the case of constrained motion. To successfully implement an assembly task it is not necessary to specify an exact position and force trajectory, rather the demonstrations identify a range of acceptable positions, magnitudes of force, and directions force.

In the analysis of unconstrained motion there was no variation in the environment, and thus all the human variation corresponded to inconsistency. However, in the constrained case human motion contains both adaptation and inconsistency. Accordingly, a method is presented in this chapter to distinguish between human adaptation and inconsistency. Human adaptation is used to identify necessary robot adaptation, and human inconsistency identifies the robot accuracy required for position and force.

The analysis in this thesis is applied to 3D translation. To eliminate the effect of rotation it is assumed that the part being moved is a sphere. The objective of the task is to place the sphere in a desired location relative to the workpiece. It is further assumed that the surfaces of the workpiece are flat, i.e. the workpiece is a polyhedral. These tasks are referred to as contact tasks, and represent simplified assembly. An example contact task is shown in Figure 3.1. The workpiece remains stationary during the task, yet each time the task is performed its location (translation and orientation) may vary, corresponding to part misalignment on an assembly line.

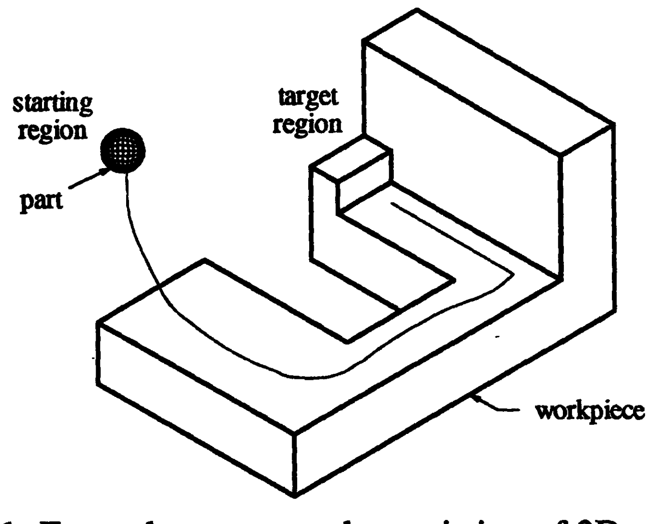


Figure 3.1: Example contact task consisting of 3D translation

To generate a successful robot program using the PHD approach, it is necessary to capture how the human adapts to variations in the environment. Accordingly, the programmer demonstrates the task a number of times, while small variations in workpiece location are introduced between the demonstrations. The demonstrator is restricted to using only sensory information that is also available to the robot. In this thesis the emphasis is on the use of position and force information, and the robot is not equipped with a vision system. Accordingly, the demonstrator closes their eyes during the demonstration[1]. This approach ensures that the sensory information available to the robot is sufficient to adapt to the environment and perform the task successfully.

To facilitate the analysis, each task is segmented into a sequence of subtasks. The method for segmenting the demonstration trajectory into a sequence of subtasks, is presented in Chapter 4. In most cases each subtask corresponds to a single contact state, where a contact state is defined by the surfaces of the workpiece in contact with the sphere. However in certain circumstance a contact state transition cannot be detected from the demonstration data, and a subtask consists of two contact state. However, it is shown in Chapter 4 that for such cases, the robot controller can be specified as if the motion was within a single contact state. This chapter addresses motion in a single contact state, and thus for the analysis of this chapter it is assumed that each subtask consists of a single contact state.

As indicated in the introductory chapter, the scope of this thesis is limited to tasks which can be demonstrated by moving the part through the same sequence of contact states each time the task is repeated. Therefore, the robot can execute the task if a robot controller is specified that can implemented each contact state, and if the robot can switch from one contact state to the next. Chapter 5 identifies subtask termination conditions that allow the robot to detect completion of motion in one contact state and switches to the next.

[1] Due to the inconvenience of demonstrating a task with one's eyes closed, the use of vision is allowed when it is not used to adapt to variations in the environment. One such case is where adaptation only occurs during the fine motion part of the task, and thus the use of vision is allowed during gross motion.

The objective of this chapter is to identify a robot controller that can perform motion within a single contact state, as long as workpiece misalignment is not larger than that encountered during the demonstrations. To avoid excessive contact forces, a robot compliance controller is used, in which the robot reacts with the environment like a spring. If the robot stiffness is too high excessive contact forces could occur, yet if the stiffness is to low excessive position errors could occur due to disturbances from workpiece misalignment and friction. The analysis identifies a range of robot compliance appropriate for the task, for which the robot errors do not exceed the range of acceptable position and force errors identified from human inconsistency.

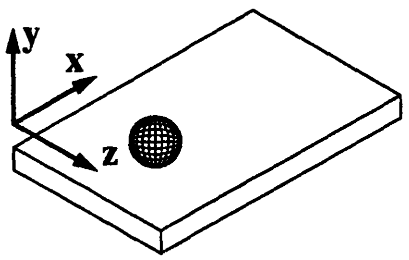
3.1.1. Problem Definition

In this chapter human demonstration data is used to determine a robot controller, including both trajectory and compliance, that successfully performs motion within a given contact state, as long as workpiece misalignment is not larger than during the demonstration. Successful implementation of a complete contact task requires that the part be placed in a desired location relative to the workpiece, without generating excessive contact forces that may damage the parts. To implement motion within a single contact state, the requirements are defined more specifically as the following conditions.

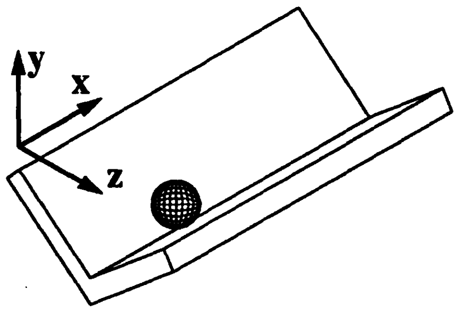
- Reach the target position of the current contact state, while avoiding obstacles on the constraint surface.
- Maintain contact with the same constraint surfaces throughout the motion.
- Avoid damage to the parts by limiting the robot contact forces to less than the maximum demonstrated human contact force.

The target position for a single contact state is the position where the part moves to the next contact state. A robot controller that can successfully implement motion within each contact state as well as switch between contact states, will be able to implement the same sequence of contact states as the human, and thus complete the task.

For the case of 3D translation there exists two possible configurations for constrained motion, as shown in Figure 3.2: a single direction of constraint and two directions of constraint. When the part is constrained in three directions, no translation is possible.



a. Single direction of constraint



b. Two directions of constraint

Figure 3.2: Two Possible Constraint Configurations for 3D Translation

The nature of constrained motion was formalized by Mason [1981]. When a part is constrained with a single constraint surface, there remains two direction of admissible motion tangential to the surface. In this configuration the constraint force (the component that does not include friction) is normal to the surface and its direction is completely defined by the constraint. Accordingly, the position trajectory on the constraint surface may vary between the demonstrations as shown in Figure 3.3. Here obstacles on the surface include any geometric location that would result in an undesirable change in the contact state. In Figure 3.3 the demonstrated trajectories circumvent a "hole" obstacle to avoid dropping off the surface. The demonstrated trajectories define a region of obstacle free motion on the workpiece surface similar to the one identified in Chapter 2 for unconstrained motion. However, here the obstacle free region is relative to the workpiece surface, whose location varies between demonstrations.

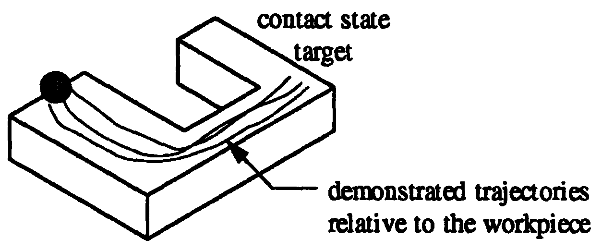


Figure 3.3: Demonstrated Trajectories On a Single Constraint
The demonstrations define an obstacle free region relative to the workpiece surface.

If the sphere traced its path along the workpiece, then the obstacle free region would be drawn on the workpiece as it is in Figure 3.3. However, the human motion is measured by the teaching gripper in an absolute coordinate system. The measured human motion contains both adaptation necessary to maintain contact with the surface, and unnecessary human motion which defines the width of the obstacle free region. The analysis in this chapter distinguishes between human adaptation to workpiece misalignment and unnecessary motion. The results are used to identify an obstacle free region relative to the workpiece surface, and the range of workpiece misalignment.

In a contact state with two constraints there exists only a single degree of freedom in the direction of motion, as shown in Figure 3.2. Accordingly the direction of motion is completely defined by the workpiece; there is no error in the direction of motion and thus obstacle avoidance is not an issue. However, to successfully implement motion with two constraints it is necessary to maintain contact with both surfaces. Here there are two degrees of freedom in direction on the constraint force. As long as the force remains in the region of acceptable force directions, as shown in Figure 3.4, the desired contact state will be maintained.

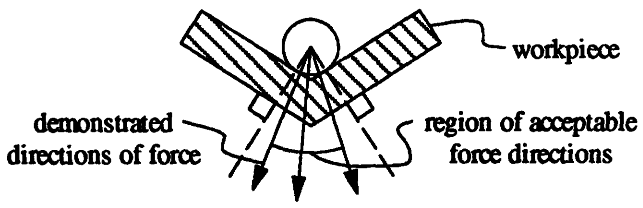

workpiece
demonstrated directions of force
region of acceptable force directions

Figure 3.4: Cross Section of Two Constraint Configuration
The demonstrated forces identify a region of acceptable force directions that ensure contact with both surfaces is maintained

Unnecessary human variation in force direction is used to identify a region of acceptable force directions, by extending the method presented in Chapter 2 for utilizing human inconsistency in position. As the demonstrator performs motion constrained by two surfaces, the direction of force can vary with motion along the surfaces and between demonstrations, as indicated by the force vectors in Figure 3.4. Nevertheless, the human successfully maintains contact with both surfaces for the range of applied forces, and thus human variation in force direction does not effect task performance. Accordingly, the human variation and identifies the region of acceptable forces shown in Figure 3.4. In this chapter, a region of acceptable force directions is identified, and a robot controller is specified that maintains the constraint force to be within this region.

The layout of this chapter combines different components of analysis which are integrated together to specify an appropriate robot controller. The first is presented in Section 3.2 where a method is presented to distinguish between human adaptation and inconsistency. A simplified 2D example of the analysis is presented in Section 3.3. The approach is then applied to 3D in Section 3.4, where ranges of acceptable motion and force are defined from human inconsistency, and the range of necessary robot adaptation is identified from the human adaptation. In Section 3.5 a robot controller is specified that can adapt to the workpiece misalignments, without exceeding the range of acceptable motion and force. The controller is specified in terms of robot compliance and trajectory. It is shown that the robot will succeed regardless of whether there are one or two constraints surfaces, and thus a model of the workpiece geometry is not required in the analysis. In Section 3.6 experimental results are presented, and the analysis is summarized in Section 3.7.

3.1.2. Assumptions

The assumptions used in the analysis are consolidated here for purposes of completeness. The first four assumptions are in regards to the task, and the remaining assumptions refer to how the human demonstrates the task.

<u>Task</u>

1. The task objective is to place a part on the target region of a workpiece, without damaging the parts with excessive forces. Required motion of the part consists of translation, but not rotation.
2. The workpiece is a rigid polyhedral and is stationary throughout the task.
3. The part is a rigid sphere and is held by the gripper without slip.
4. Random misalignments in the workpiece position and orientation occur between demonstrations.

Human Demonstrations

5. The demonstrator performs the task successfully using only position and force information. The demonstrator is successful throughout the range of workpiece misalignment. Accordingly, the success does not rely on coincidental (i.e. lucky) motion that happens to corresponds to workpiece misalignment. Rather the demonstrator adapts to the environment by using sensory information.
6. The demonstrator uses the *same* manipulation strategy each time the task is performed. Specifically, in each demonstration the demonstrator:
   - Avoids obstacles by passing them on the *same* side.
   - Uses the *same* sequence of contact states.
7. While performing motion within a contact state, the demonstrator does not use information acquired during previous contact states[2]
8. The demonstration is performed quasi-staticaly and thus dynamic forces can be neglected.
9. Only geometric features of the workpiece are used to adapt to misalignments, and not variations in surface roughness.
10. The teaching device sensors are accurate enough to detect the geometric features that the human uses.

3.1.3. Background

It is recognized that to perform assembly operations reliably it is necessary to adapt to part misalignment. A number of model based methods have been implemented which utilize a detailed model of the part geometry as well as knowledge of the range of variation in part location. A common approach is to specify the robot's impedance, so that the manipulator reacts with the environment as spring or a damper. Whitney [1982] presents a compliance controller for the task of inserting a peg into a hole. Whitney

[2] It is not possible to restrict the human from using previously acquired information. Nevertheless, in model based robot programming significant adaptation to the environment can be implemented using only information from the current contact state, as is illustrated by Lozano-Pérez, Mason, and Taylor [1984]. Here we assume that the human uses a similar adaptation technique.

[1977], Lozano-Pérez, Mason, and Taylor [1984], and Schimmels and Peshkin [1992] present methods of using damping control for assembly operations. A limiting factor in the use of model based methods is their analytical difficulty and the lack of an algorithm that can be applied to arbitrary part geometry. An additional difficult with model based methods is that inaccuracies in the model often require fine tuning of the robot program.

Due to the difficulties of model based methods, the use of human demonstration data has been recognized as an alternative. Asada and Izumi [1987] use human demonstration data exclusively to identify both a robot trajectory and a hybrid force/position controller to perform 3D contact tasks. A difficulty addressed by Asada and Izumi is that a single measurement of force and motion directions is not sufficient to identify the directions of admissible motion and force, which is required by a hybrid controller. Thus, to implement the controller it is assumed that the constraint surfaces are perpendicular to each other and that each contact state transition can only add a constraint, which limits possible workpiece geometries. Asada and Izumi [1987] originate the objective of generating both a robot trajectory and controller exclusively from demonstration data. The analysis in this chapter incorporates this objective and extends the analysis by allowing workpiece geometries with arbitrary surface orientations, and by allowing the number of constraints can increase or decrease at each contact state transition throughout the task. In addition, here the analysis explicitly evaluates the effect of misalignment in workpiece orientation, which was not done in the previous analysis.

Another method of using human demonstration data is through observation with a vision system [Kuniyoshi, Inaba, and Inoue 1992; Ikeuchi, Kawade, and Suehiro 1993]. A limitation of this approach is that forces used by the human are not measured, which may contain a significant component of the manipulation strategy especially during constrained motion. Accordingly, Ikeuchi et al. implement detailed constrained motion by supplementing the demonstration data with model based knowledge. In contrast, the approach presented here uses only the demonstration data to specify all aspects of the robot program, including the details of the force and position control.

Liu and Asada [1992] use human demonstration data for the task of automating a grinding task for the purposes of deburring. They present a method of identifying how the human adapts to the environment and transfer these skills to the robot. In their approach, inconsistent human motion is identified and removed from the analysis. An alternative approach of interpreting human motion is presented here, whereby human inconsistency is not interpreted as noise that should be eliminated from the analysis, but provides information regarding the position and force accuracy requirements for the task.

One of the difficulties of interpreting human demonstration data is that the internal model of how the human controls their motion is not known. Some research approaches have made assumptions regarding the method of human control, yet these methods have lead to unreliable results when the demonstrated actions do not coincide with the model. Takahashi, Ogata, and Moto [1993] present a method for segmenting a task into subtasks by identifying when the human slows down, yet the demonstrator does not consistently slow down at the same locations in all demonstrations. Delson and West [1992] present a method that assumes that the demonstrator uses compliance control and performs each demonstration using the same compliance and reference trajectory, yet this method did not provide reliable results. To address this difficulty, the approach presented here does not rely on a model of how the human implements the position and force control of their arm. Instead, the demonstration data is used to identify sufficient conditions for task success, and then a robot controller is specified that can achieve these conditions.

3.2. Distinguishing Between Adaptation and Inconsistency

In the obstacle avoidance analysis in Chapter 2, the demonstrator repeats the same task with no variation in the environment. In that case, variation between demonstrations is unnecessary for task performance and is interpreted as human inconsistency. However, when a demonstrator performs contact tasks they adapt to workpiece misalignment, which is essential for task success. While the human is adapting, their motion also contains some unnecessary motion. Accordingly, demonstrations of contact tasks contain both human adaptation and inconsistency. To implement PHD and generate a robot program, human adaptation is distinguished from inconsistency. Identifying human adaptation enables us to develop a robot controller that can also adapt to the environment. Furthermore, identifying human inconsistency provides information regarding the accuracy requirements for the task, as in the case of obstacle avoidance. To distinguish between human adaptation and inconsistency, the following theorem is presented:

Theorem I

The demonstrator varies their position and force trajectories each time the task is performed. Human variations that are geometrically equivalent to *detectable* workpiece misalignment, are interpreted as adaptation to the environment. All other variations are unnecessary for task success, and are interpreted as human inconsistency.

The justification for this theorem is based on causality. The human can only adapt to variations in the environment that they detect through their sensory channels. For an arbitrary task it may be quite difficult to determine how the human uses sensory information. However, for the case of assembly or a contact task, the adaptation strategy can be defined in specific geometric terms. The task objective is to place the part in a desired position relative to the workpiece. Accordingly, we assume that a "perfectly consistent" demonstrator would attempt to generate the same trajectory *relative* to the workpiece in all demonstrations. Whenever, a "perfectly consistent" demonstrator *detects* workpiece misalignment, they modify the part trajectory to geometrically correspond to the detected workpiece position. In actuality the demonstrated motion contains both "perfectly consistent" adaptation and unnecessary motion. The above theorem distinguishes between the two by correlating adaptation to detectable workpiece misalignment.

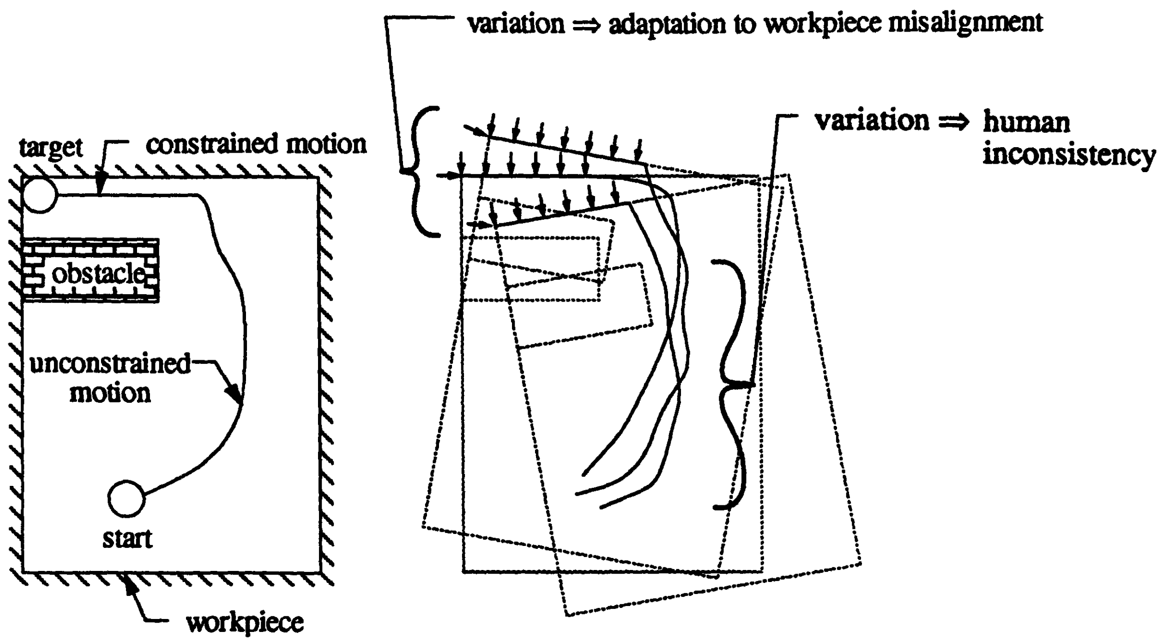
Adaptation is based on *detectable* workpiece misalignment, which is not equivalent to actual misalignment. Accordingly, the first step in the analysis is to extract from the sensor measurements the information which indicates the workpiece position. For example, as the part is moved across a workpiece surface certain directions of workpiece orientation can be identified while other directions cannot be discerned. Indeed, no information is available prior to contact with the workpiece. In Section 3.4

which addresses 3D translation, the directions of detectable and undetectable directions of motion are presented. The demonstrator cannot adapt to misalignments that are undetectable, nevertheless they consistently succeed at performing the task. Accordingly, it is not necessary for the robot to adapt to these undetectable misalignments either.

To illustrate how the theorem is applied, a simplified example is presented in the following section by limiting the analysis to 2D translation. Adaptation is distinguished from inconsistency, and the results are used to specify sufficient conditions for robot success. This approach is then extended to the 3D case in Section 3.4.

3.3. 2D Translation

An example of a contact task consisting of 2D translation is shown in Figure 3.6a. The first part of the task consists of moving the part in unconstrained motion until it contacts a constraint surface, while avoiding the obstacle. The second part of the task consists of constrained motion along a workpiece surface. 2D translation is significantly simpler than 3D translation, since once motion is constrained there remains only one degree of freedom. Thus the directions of motion force are completely defined by the constraint surface. As will be shown, this attribute is used in this example to simplify the distinction between adaptation and inconsistency.


variation ⇒ adaptation to workpiece misalignment
variation ⇒ human inconsistency
target
constrained motion
obstacle
unconstrained motion
start
workpiece

a. Nominal configuration b. Demonstrated position and force trajectories

Figure 3.6: 2D contact task.
The task objective is to move the round part to the target in the workpiece corner using only position and force information. Variations between demonstrated trajectories are interpreted as due to human inconsistency during unconstrained motion, and due to workpiece misalignment during constrained motion.

The trajectories from multiple demonstrations are illustrated in Figure 3.6b; the constraint force without friction is depicted by the force vectors and the dotted outlines indicate the workpiece misalignment. During the unconstrained portion of the task, the demonstrator has no information regarding workpiece misalignment. Nevertheless, the demonstrator is able to avoid the obstacle by providing a sufficient margin from the nominal obstacle location. The demonstrated trajectories during unconstrained motion define an obstacle free region similar to the region defined with stationary obstacles, in Chapter 2. As long as the workpiece misalignment is not larger than during the demonstrations, obstacle avoidance can be guaranteed with a robot trajectory that remains within the region bounded by the demonstrated trajectories.

Variations between demonstrations in sections of unconstrained motion are attributed to human inconsistency. During unconstrained motion no information regarding workpiece location is available, and thus there is no reason for the human to vary their motion from one demonstration to the next. Indeed the causal relationship between sensory information and intentional adaptation indicates that it is impossible for these variations to correspond to adaptation to the environment. Accordingly, these variations are unnecessary for task success, and the region bounded by demonstrated trajectories represents a range of acceptable robot motion. As in Chapter 2, human inconsistency can be used to remove unnecessary robot motion and provide a buffer from the obstacles in case of robot error.

In one respect, obstacle avoidance differs in this case from the case of stationary obstacles presented in Chapter 2. Here the obstacle location varies with the workpiece, yet the demonstrations which define the obstacle free region are measured relative to a fixed coordinate system. Guaranteeing that the region bounded by the demonstrations is obstacle free, relies on assumption 5, which indicates that the demonstrator is successful in all demonstrations. One could imagine a single demonstration where an obstacle is avoided only through luck. For example a demonstration that passed through the nominal location of the obstacle could succeed if during that specific demonstration workpiece misalignment coincidentally moved the obstacle away from that location in an appropriate direction. A region define from such a 'lucky' trajectory would not always be obstacle free. However, such a demonstrator would eventually hit an obstacle when their luck runs out in one of the demonstrations. On the other hand, the trajectories from a demonstrator that is consistently successful, will lie outside of the region in which the obstacle *could occur* even when there is misalignment. The fact that the demonstrator can repeatedly avoid the obstacle indicates that the obstacle misalignments are small enough so that no adaptation to their position is required, and that the region bounded by the demonstrations is obstacle free. In conclusion, the demonstrator cannot adapt to undetectable misalignment, yet if the demonstrations are consistently successful then a robot does not need to adapt to these undetectable misalignments either.

In the second section of the task in Figure 3.6, the part is constrained by the workpiece. Here the direction of motion is defined by the constraint surface, and thus variation between demonstrations are due to workpiece misalignment. These variations are treated differently than the variations in unconstrained motion that are due to human inconsistency. By measuring the range of variation in direction of constrained motion, it is possible to identify the range of workpiece misalignment. The fact that the demonstrator maintains contact with the surface, indicates that the human adapts to the environment by modifying their trajectory in a direction geometrically equivalent to the workpiece misalignment. The human may perform this level of adaptation subconsciously, however this type of adaptation is explicitly required when programming a robot to perform a contact task. A successful robot controller is required to adapt to the range of workpiece misalignment, which is identified from the human adaptation, by maintaining contact with the constraint surface and avoiding excessive contact forces.

In 2D translation, distinguishing between adaptation and inconsistency is simple. During unconstrained motion all variations between demonstrations correspond to inconsistency, while during constrained motion all variations correspond to workpiece misalignment. A robot program for the example task of Figure 3.6 can be specified as the following:

1. Move from the starting position to the constraint surface while staying in the obstacle free region defined by the unconstrained sections of demonstrated trajectories.
2. Detect when contact with the constraint surface occurs.
3. Slide along the constraint surface to the left. Maintain contact with the surface and avoid excessive contact forces, despite workpiece orientation misalignment.

The first step can be implemented using techniques presented in Chapter 2; a robot trajectory is generated within the obstacle free region and implemented with position control. Implementing the second step requires detection of subtask termination conditions, which is addressed in Chapter 5. The third step, however, is the subject of this chapter, and requires specifying a robot controller for constrained motion that can adapt to workpiece misalignment.

The human can implement constrained motion in a number of ways that range from maintaining a constant force to controlling the impedance of their arm. In our approach we do not attempt to duplicate the internal control method the human uses for constrained motion, rather we seek a robot controller whose performance is sufficient for the task requirements. The type of robot controller which is selected is a compliance controller, where the manipulator is programmed to respond as a linear spring. The details of compliance controller performance are presented in Section 3.5. However, for the purposes of completing the 2D example, a brief analysis is presented here.

Compliance control enables a robot to maintain contact with a surface without creating excessive contact forces. Figure 3.7 illustrates the use of compliance control to slide along a surface. The equilibrium position of the spring, referred to as the reference position, is located in a position that preloads the spring and generates a contact force. The reference trajectory is programmed to be parallel to the nominal surface location, as shown in Figure 3.7a.

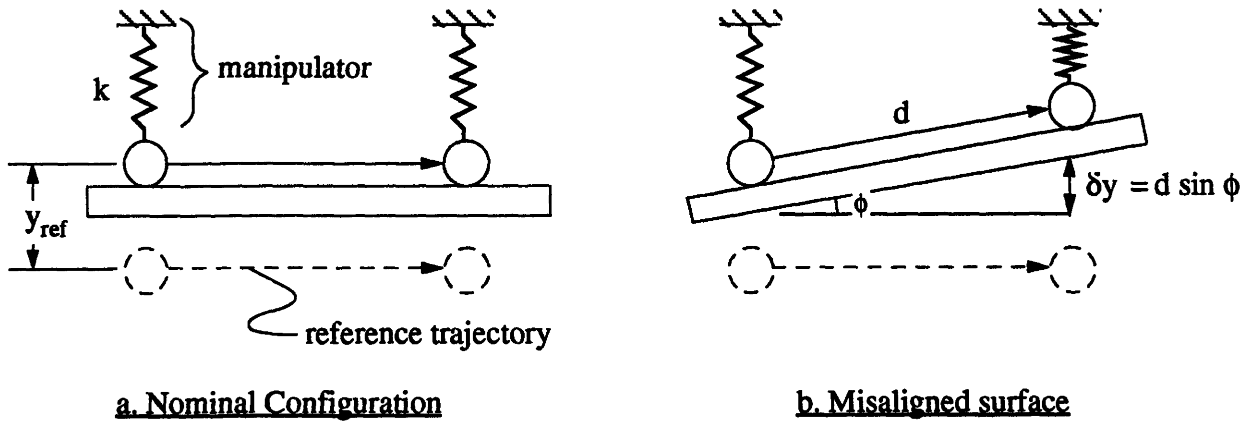


Figure 3.7: 2D Compliance Control.

When the surface is in the nominal configuration, the contact force is equal to its nominal value, f^n, and is given by:

$$f^n = k\, y_{ref} \tag{3.1}$$

where k is the manipulator's stiffness and y_{ref} is the position of the reference trajectory relative to the nominal surface location as shown in Figure 3.7a.

Workpiece misalignment causes the surface orientation to vary by the angle ϕ as shown in Figure 3.7b. The compliance of the manipulator is used to adapt to the environment, and thus there is no need to modify the reference trajectory. When ϕ is positive, the contact force increases due to compression of the spring. The change in the spring compression is dsinϕ, where 'd' is the distance traveled, and the increase in contact force is kdsinϕ. To prevent damage to the parts the maximum allowable contact force, f_{max}, is set is to the maximum force measured during the demonstrations. Excessive contact forces are avoided as long as:

$$f^n + k\, d \sin\phi < f_{max} \tag{3.2}$$

In the case where ϕ is negative, the surface moves away from the manipulator. To maintain contact with the surfaces, the reference trajectory is required to remain beneath the surface, which is satisfied as long as:

$$|d \sin\phi| < y_{ref} \tag{3.3}$$

If the robot stiffness is properly selected it is possible to satisfy both of the above inequalities. When the nominal force is then selected to be $f_{max}/2$, both of the inequalities result in the same condition for k, given by:

$$k < \frac{f_{max}}{2\, d \sin\phi} \tag{3.4}$$

The above analysis provides an upper bound on the robot stiffness. In Section 3.5 the 3D case is analyzed, and the result provides both upper and lower limits for robot stiffness. In addition, the effect of friction is considered.

A summary of the analysis is shown in the flow chart in Figure 3.8. As indicated, distinguishing between adaptation and inconsistency allows us to program the robot to adapt to the environment, and to identify a range of acceptable motion.

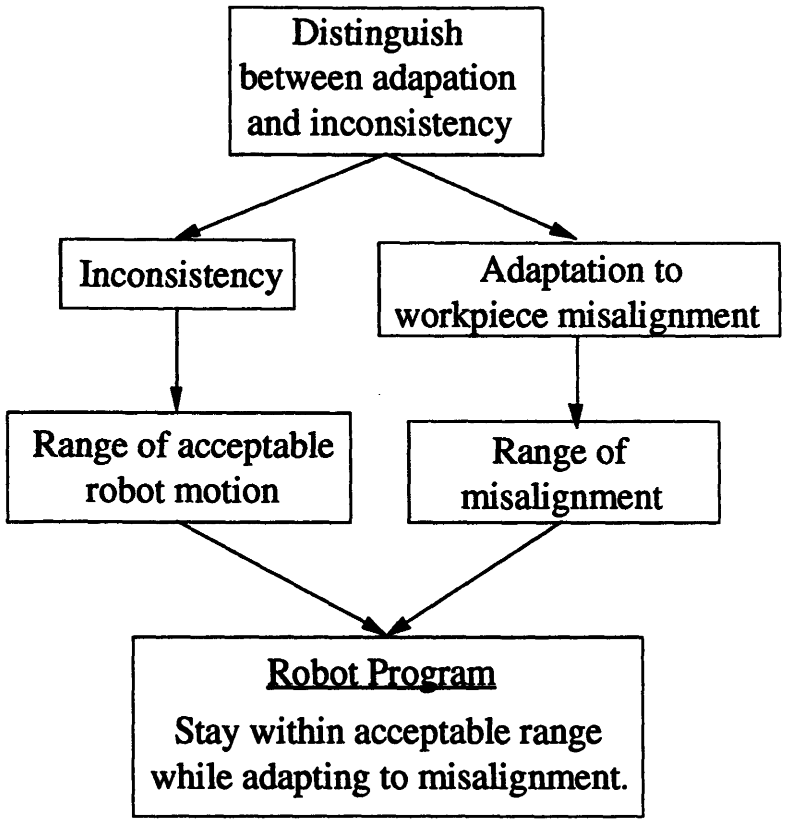
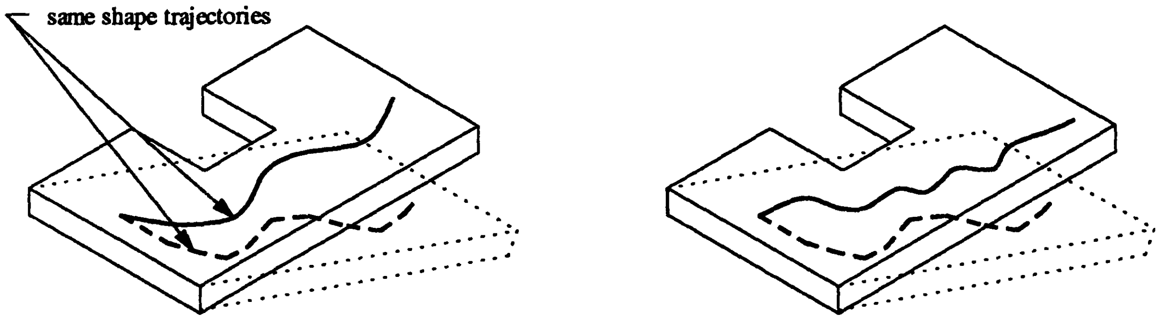

Distinguish between adapation and inconsistency
Inconsistency
Adaptation to workpiece misalignment
Range of acceptable robot motion
Range of misalignment
Robot Program
Stay within acceptable range while adapting to misalignment.

Figure 3.8: Flow chart indicating use of both adaptation and inconsistency.

3.4. 3D Translation

In the case of 3D translation, human adaptation and inconsistency can occur in the same contact state. An example is presented in Figure 3.9, where trajectories from two demonstrations are shown. If the demonstrator was "perfectly consistent", then only adaptation would occur as shown in Figure 3.9a and the shape of the two trajectories would be the same. With a "perfectly consistent" demonstrator the only modification between trajectories would correspond to misalignment in workpiece orientation to enable the trajectories to maintain contact the surface. In actuality inconsistency occurs together with adaptation, as shown in Figure 3.9b, and the shape of the trajectory varies in directions other than necessary to maintain contact.



a. Only Adaptation b. Adaptation and Inconsistency

Figure 3.9: 3D Adaptation and Inconsistency
Trajectories from two demonstrations are shown, where the dashed lines correspond to the misaligned workpiece. If the demonstrator was "perfectly consistent", then only adaptation would occur as shown in 'a'. In actuality both adaptation and inconsistency occur as show in 'b'.

In this section adaptation is distinguished from inconsistency for 3D translation, for motion within a single contact state. The analysis is applied to both possible constraint configurations, shown in Figure 3.2. First the directions of detectable workpiece misalignment are identified for each constraint configuration. Then the ranges of workpiece misalignment and acceptable motion are identified.

3.4.1. Detectable and Undetectable Misalignments

The demonstrated force and position trajectories are designated by **x**(t) and **f**(t), and have three components corresponding to the *x, y, z* directions in a fixed Cartesian coordinate system. The demonstration is repeated N times, and when necessary to indicate the specific demonstration the subscript 'i' is used. The analysis in this chapter applies to motion within a single contact state, and thus the specific contact state is not designated, with the understanding that the analysis is repeated for each contact state in the task sequence.

The force information used in the analysis is the force component due to contact between the part and the workpiece. Accordingly, the force measurements from the teaching gripper are transformed to a global coordinate system and the gravitational forces are removed prior to their use in the analysis. In addition, the dynamic forces can be neglected since the motion is quasi-static. The force sensor orientation varies with the gripper motion, yet the position sensor provides the force sensor's orientation with which the force measurements are transformed to a global coordinate system. The gravitational load of the teaching gripper is then subtracted from the measured force, and the vector **f**(t) indicates the force due to contact with the environment. The contact force, **f**, consists of a constraint force, $\mathbf{f}_c$, which is normal to the surface, and a frictional force, $\mathbf{f}_f$, that is tangential to the surface. The friction is modeled as Coulomb friction with an isotropic coefficient of friction, and thus the friction force is opposite the direction of motion [Peshkin and Sanderson 1989]. The direction of motion is evaluated by taking the time derivative of the position trajectory, which is indicated by the normalized velocity vector $\hat{\mathbf{v}}(t)$. The friction force is the component of **f** that is aligned with $\hat{\mathbf{v}}(t)$, which is calculated by projecting $\hat{\mathbf{v}}(t)$ onto **f**(t) using the transpose of one of the vectors and matrix multiplication. Accordingly, the vector of friction forces can be evaluated when the velocity is greater than zero, and is given by the following equation, where the superscript 'T' indicates the transpose of a vector.

$$\mathbf{f}_f(t) = \left([\mathbf{f}(t)]^T \hat{\mathbf{v}}(t)\right) \hat{\mathbf{v}}(t) \qquad \text{for } |\mathbf{v}(t)| > 0 \qquad (3.5)$$

The forces $\mathbf{f}_c$ and $\mathbf{f}_f$ are orthogonal complements, and the constraint force is calculated by subtracting friction component, which is given by:

$$\mathbf{f}_c(t) = \mathbf{f}(t) - \mathbf{f}_f(t) \qquad \text{for } |\mathbf{v}(t)| > 0 \qquad (3.6)$$

The magnitude of the friction components is the product of the coefficient of friction, μ, with the normal force, $\mathbf{f}_c$. Accordingly, μ can be estimated by:

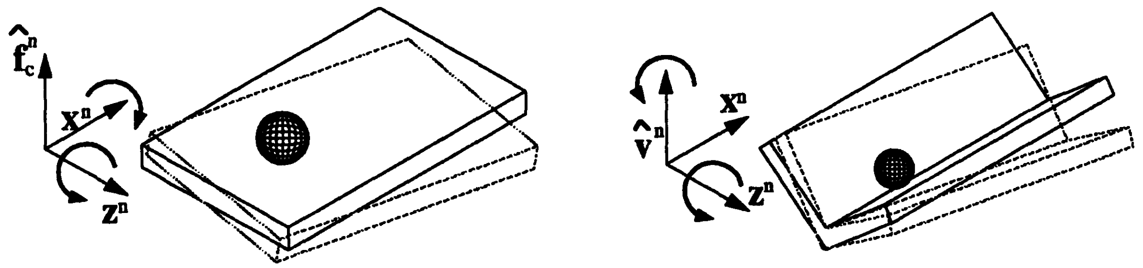
$$\mu(t) = \frac{|\mathbf{f}_f(t)|}{|\mathbf{f}_c(t)|} \qquad \text{for } |\mathbf{v}(t)| > 0 \qquad (3.7)$$

The position and force trajectories are a function of time, but can also be represented as a function of the distance traveled, 's'. Since the demonstrations are performed quasi-staticaly, the magnitude of the velocity is not critical for task success. In addition, when the velocity is zero the friction force cannot be distinguished from the constraint force using Equation 3.6, yet both $\mathbf{f}_c$ and $\mathbf{f}_f$ can be defined for all values of 's'. Accordingly, for the remainder of analysis in this chapter the demonstrated trajectories are represented by **x**(s), $\mathbf{f}_c$(s), and $\mathbf{f}_f$(s) .

The ability to detect workpiece misalignment, depends on how the constraint effects the motion and force. The nature of constrained motion was formalized by Mason [1981]. Admissible directions of constraint force are normal to the surface, and admissible directions of motion are tangential to the surface. In the case of a single constraint surface (Figure 3.2a), there exist two directions of admissible motion and one directions of admissible force. When there are two constraint surfaces (Figure 3.2b), there exist one direction of admissible motion and two directions of admissible force.

Figure 3.10 displays the directions of workpiece orientation that can be detected during motion constrained in one and two directions. In the single constraint configuration, the direction normal to the surface can be detected through measurement of the normalized constraint force vector, $\hat{\mathbf{f}}_c$. However, rotation of the surface about the axis of $\hat{\mathbf{f}}_c$ cannot be detected, since it does not vary either the measured force or position. Accordingly, orientation misalignment due to rotation about the 'x' and 'z' in Figure 3.10a can be detected, but rotations about the 'y' axis are undetectable. In two constraint configuration, the direction of motion can be detected through measurement of $\hat{\mathbf{v}}(t)$, which corresponds to workpiece rotations about the 'y' and 'z' axes in Figure 3.10b, yet rotation about the 'x' axis does not change either force or position and cannot be detected.

Of course if the rotation about the 'x' in the two constraint configuration was large enough, contact with one of the surfaces could be lost, and additional misalignment could be detected. However, it is assumed that the same sequence of contact states be used in all demonstrations (assumption 6), which precludes intermittent loss or gain of contact (either unintentionally or as a result of probing). Accordingly, motion corresponding to a single contact state is demonstrated without loss of contact with the desired constraint surfaces.



a. one constraint b. two constraints

Figure 3.10: Detectable Workpiece Misalignment
Detectable directions of workpiece rotation are shown by arrows in each coordinate system.

This analysis assumes that once the demonstrator is within a contact state, the only misalignments they can detect are those indicated by $\hat{f}_c$ and $\hat{v}$ as shown in Figure 3.10. To ensure that this assumption is not violated, we have assumed that the demonstrator does not use information acquired from previous contact states (assumption 7), and assumption 9 indicates that anisotropic surface roughness or variation in surface roughness either do not exist or are not used by the demonstrator to detect misalignment.

3.4.2. Identifying the Constraint Configuration

The PHD approach does not use a geometric model of the workpiece, and thus it is not explicitly known whether a constraint is applied in one or two directions (Figure 3.2). Identifying transitions from one constraint surface to the next is a subject onto itself, which is presented in Chapter 4. For the purposes of the analysis in this chapter it is assumed that the set of demonstration data, **x**(s) and **f**(s), belongs to motion in which the surfaces in contact with the part do not change. In this section, the number of constraints is identified from the demonstration data when possible.

Due to the nature of constrained motion, if $\hat{f}_c(s)$ is constant while $\hat{v}(s)$ varies, then the motion occurs in a single constraint configuration. Conversely, if $\hat{v}(s)$ is constant while $\hat{f}_c(s)$ varies, then there are two constraints. Regardless of the constraint configuration, at least one of the directions, $\hat{v}(s)$ or $\hat{f}_c(s)$, will theoretically remain constant. In actuality, noise in the sensor measurements creates some variations in all of the data.

To determine whether $\hat{f}_{c,i}(s)$ can be considered constant in the ith demonstration, the average direction is calculated from all the points measured within the ith demonstration which is designated as $\bar{\hat{f}}_{c,i}$. The difference in direction between a point in the trajectory $\hat{\mathbf{f}}_{c,i}(s)$ and $\bar{\mathbf{f}}_{c,i}$ is given by:

$$\beta_k = \arccos\left(\hat{\mathbf{f}}_{c,i,k} \bullet \bar{\mathbf{f}}_{c,i}\right) \qquad \text{for } k = 1,2,\ldots K_i \qquad (3.8)$$

where 'k' is the index indicating the point sampled, and K_i is the number of points sampled in the ith demonstration. If the mean value of β_k is less than a threshold level, then variations in the direction of force are attributed to noise and $\hat{\mathbf{f}}_c(s)$ is considered constant. The threshold level is determined experimentally, from measurements over a known single constraint surface.

It would be possible to evaluate whether $\hat{\mathbf{v}}(s)$ can be considered constant using a similar approach as with $\hat{\mathbf{f}}_c(s)$. However, $\hat{\mathbf{v}}(s)$ is estimated by taking the derivative of the position trajectory, which magnifies the level of noise in $\hat{\mathbf{v}}(s)$. To reduce sensitivity to noise, the position trajectory is evaluated directly to determine whether it lies on a straight line. A least squares fit to a straight line is performed, and the slope of the line is given by $\bar{\mathbf{v}}_i$. To determine whether $\hat{\mathbf{v}}(s)$ can be considered constant, the cumulative error between the straight line and the sampled data is compared to a threshold level identified from the noise in the position measurement, which is presented in Appendix I. The noise is assumed Gaussian and the threshold level is given by $\sigma_n^2\left[K + 2\sqrt{2K}\right]$, where σ_n is the standard deviation of the noise in the position data.

If within each demonstration $\hat{\mathbf{f}}_c(s)$ is considered constant and $\hat{\mathbf{v}}(s)$ varies, then it is established that there is a single constraint surface. If the reverse is true and $\hat{\mathbf{v}}(s)$ is considered constant in each demonstration and $\hat{\mathbf{f}}_c(s)$ varies, then it is established that there are two constraint surfaces.

There exists, however, a third alternative in which both $\hat{\mathbf{v}}(s)$ and $\hat{\mathbf{f}}_c(s)$ are considered constant. There is nothing that prevents a demonstrator from moving in a straight line on a single constraint surface or keeping the direction of force constant while constrained in two directions. If this is done in all the demonstrations, it is impossible to identify the constraint configuration. Under these circumstances the analysis is performed first assuming there is a single constraint and then repeated assuming that there are two constraints. In Section 3.5.2.3 it is shown that for this case a robot controller can be specified that can implement the motion regardless of the constraint configuration. Accordingly, restrictions placed on the geometry of the contact task by Asada and Izumi [1987] to allow identification of the number of constraints, are not necessary in this method of analysis.

3.4.3. Nominal Surface Orientation

The nominal orientation of a constraint surface is the expected orientation of that surface, and occurs when there is no workpiece misalignment. Since the workpiece misalignments are random (assumption 4), the nominal orientation is calculated by averaging the detected directions of surface orientation from all the demonstrations.

In a single constraint configuration, the surface normal is indicated by $\tilde{\mathbf{f}}_{c,i}$, where the subscript 'i' designates the ith demonstration. The nominal direction of the constraint force, $\hat{\mathbf{f}}_c^n$, is calculate by averaging over the N demonstrations and is given by:

$$\hat{\mathbf{f}}_c^n = \frac{\sum_{i=1}^{N} \tilde{\mathbf{f}}_{c,i}}{\left| \sum_{i=1}^{N} \tilde{\mathbf{f}}_{c,i} \right|} \tag{3.9}$$

A coordinate system is defined with the 'y' axis aligned with $\hat{\mathbf{f}}_c^n$ as shown in Figure 3.11.

In the two constraint configuration, the direction of motion is indicated by $\bar{\mathbf{v}}_i$, where the subscript 'i' designates the ith demonstration. The nominal direction of motion, $\hat{\mathbf{v}}^n$, is calculate by averaging over the N demonstrations and is given by:

$$\hat{\mathbf{v}}^n = \frac{\sum_{i=1}^{N} \bar{\mathbf{v}}_i}{\left| \sum_{i=1}^{N} \bar{\mathbf{v}}_i \right|} \tag{3.10}$$

A coordinate system is defined with the 'x' axis aligned with $\hat{\mathbf{v}}^n$ as shown in Figure 3.12.

3.4.4. Single Constraint Configuration

Performing motion constrained in a single direction requires reaching the target while avoiding obstacles and excessive forces. The demonstrator maintains contact with the surface, thereby adapting to orientation misalignment and eliminating position errors normal to the surface. In addition, the demonstrator avoids obstacles through familiarity with the workpiece geometry. These aspects of the manipulation strategy are quantified in this section, so that they may be transferred to the robot program.

In the ith demonstration, misalignment is detected when the direction $\vec{\mathbf{f}}_{c,i}$ differs from the nominal direction, $\hat{\mathbf{f}}_c^n$. A fixed coordinate system is defined that is aligned with the nominal surface orientation; the 'y' axis is aligned with $\hat{\mathbf{f}}_c^n$, while the '$\mathbf{x}^n$' and '$\mathbf{z}^n$' axes are selected to lie on the plane normal to $\hat{\mathbf{f}}_c^n$. In addition a relative coordinate system that corresponds to the detected workpiece orientation of the ith demonstration is given by, $\mathbf{x}_i$, $\vec{\mathbf{f}}_{c,i}$, and $\mathbf{z}_i$. Figure 3.11 shows the misalignment between these two coordinate systems, which is represented by a rotation about the vector $\vec{\mathbf{r}}_i$ by the angle ϕ_i.

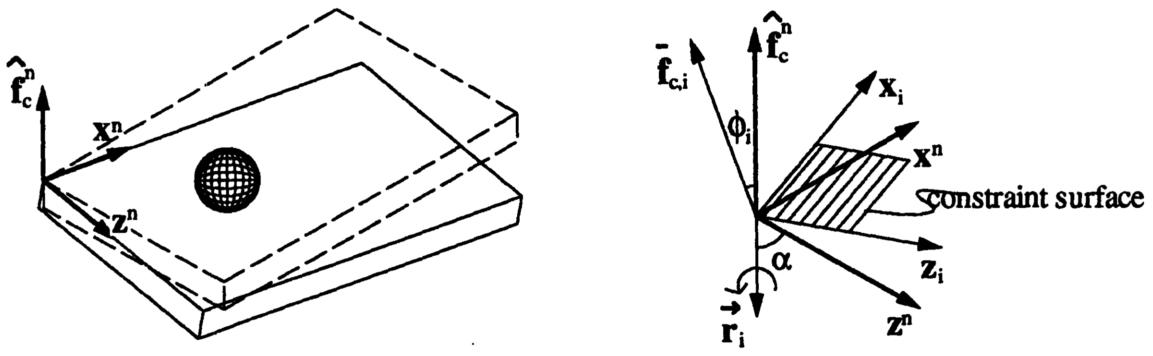


Figure 3.11: Detectable Misalignment Of a Single Constraint

The detected misalignment corresponds to the rigid body rotation that would cause $\hat{\mathbf{f}}_c^n$ to rotate to the direction of $\vec{\mathbf{f}}_{c,i}$. Rotation about the axis $\hat{\mathbf{f}}_c^n$ cannot be detected, and for the purposes of interpreting human motion it is assumed that no rotation occurs in the undetected direction. Accordingly, the orientation of $\vec{\mathbf{r}}_i$ is given by the cross product between $\hat{\mathbf{f}}_c^n$ and $\vec{\mathbf{f}}_{c,i}$:

$$\vec{\mathbf{r}}_i = \hat{\mathbf{f}}_c^n \times \vec{\mathbf{f}}_{c,i} \tag{3.11}$$

and the angle of misalignment is the arc cosine of the dot product:

$$\phi_i = \arccos\left(\hat{\mathbf{f}}_c^n \bullet \vec{\mathbf{f}}_{c,i}\right) \tag{3.12}$$

The $\vec{\mathbf{r}}_i$ vector is defined in the nominal coordinate system and lies in the 'xz' plane; its components can be written as:

$$\vec{r}_i = \begin{bmatrix} -\sin(\alpha) \\ 0 \\ \cos(\alpha) \end{bmatrix} \qquad (3.13)$$

where, the value of α is given by the arc tangent of the 'x' and 'z' components of $\vec{r}_i$.

A rigid body rotation can be expresses by a 3x3 rotation matrix whose columns are the unit vectors in the x, y, and z directions of a coordinate system attached to the rigid body. The rotation matrix for a rotation about an arbitrary vector is presented by Crandall et al [1985]. Here the rotation is about the vector $\vec{r}_i$ whose 'y' component is zero, and the rotation matrix indicating workpiece misalignment, $\mathbf{R}_{wp}$, is given by:

$$\mathbf{R}_{wp} = \begin{bmatrix} c\phi + (1 - c\phi)\, s^2\alpha & -s\phi\, c\alpha & -(1 - c\phi)\, s\alpha\, c\alpha \\ s\phi\, c\alpha & c\phi & s\phi\, s\alpha \\ -(1 - c\phi) s\alpha\, c\alpha & -s\phi\, s\alpha & c\phi + (1 - c\phi)\, c^2\alpha \end{bmatrix} \qquad (3.14)$$

where 's' and 'c' are abbreviations for sine and cosine.

Successful implementation of the motion along the surface requires that obstacles be avoided. The demonstrated trajectories trace 2D paths on the workpiece surface, and human inconsistency defines a region of acceptable robot motion, as shown in Figure 3.3. As indicated by theorem I, human inconsistency is identified from variations between demonstrations that do *not* correspond to detectable workpiece misalignment. Accordingly, each demonstration is transformed to the coordinate system aligned with the surface, thus removing variations due to workpiece misalignment.

Transforming each demonstration to a coordinate system aligned with the workpiece surface allows the shapes of the 2D trajectories to be compared to one another. It is likely that all the trajectories do not start at the same position on the surface. However, variations in the starting position cannot be detected by the demonstrator, since the demonstrator does not utilize information from prior contact states (assumption 7) and translation misalignment of the workpiece cannot be detected in the directions of admissible motion. Accordingly, it is assumed that all demonstrations begin at the same location on the surface which is designated as the origin. Orientation misalignment that can be detected is indicated by the matrix

$\mathbf{R}_{wp}$. The trajectory relative the detectable surface location is designated by $\mathbf{x}_i'(s)$ and the coordinate system transformation is given by:

$$\mathbf{x}_i'(s) = \begin{bmatrix} x_i'(s) \\ 0 \\ z_i'(s) \end{bmatrix} = \mathbf{R}_{wp}^T [\mathbf{x}_i(s) - \mathbf{x}_i(s=0)] \qquad (3.15)$$

The 'y' component of $\mathbf{x}_i'(s)$ is equal to zero because the motion is constrained on the 'xz' plane aligned with the surface. For each demonstration a 2D trajectory is defined on 'xz' plane. These trajectories are combined using the methods presented in Chapter 2 to define a region of acceptable motion on the constraint surface. For consistent notation in this chapter the region of acceptable motion is designated by $\mathbf{X}_A$, which corresponds to the same region designated by R_F in Chapter 2. As in the case of 2D motion, the region $\mathbf{X}_A$ excludes all possible obstacle locations even when undetected workpiece misalignments occur.

The above analysis identifies conditions sufficient to guarantee robot success for motion constrained by a single surface. The magnitude of the workpiece misalignment is given by ϕ_i in the direction of $\hat{\mathbf{r}}_i$. The demonstrator adapts to this misalignment by maintaining contact with the surface, and a successful robot controller is required to also maintain contact with surface and avoid excessive forces for the maximum value of ϕ encountered during the demonstrations. In addition the target can be reached while avoiding obstacles, by specifying a robot trajectory that stays with the region of acceptable motion, $\mathbf{X}_A$, defined from human inconsistency. A robot controller that can satisfy these sufficient conditions is presented in Section 3.5.

3.4.5. Two Constraint Configuration

A duality exists between the single constraint and the two constraint configurations. With a single constraint the direction of force is defined by the workpiece yet there remain two directions of admissible motion, while with two constraints the direction of motion is defined and there exists two directions of admissible force. In the two constraint configuration, position accuracy is assured as long as contact is maintained with both constraint surfaces. Accordingly, the robot performance requirements are indicated in terms of directions and magnitudes of applied force. The following analysis parallels the single constraint analysis.

Workpiece misalignment for the ith demonstration is detected when the direction, $\bar{v}_i$, differs from the nominal direction, $\hat{v}^n$. Here the fixed coordinate system is defined with the 'x' axis aligned with $\hat{v}^n$, and a relative coordinate system is aligned with the detected workpiece location as shown in Figure 3.12.

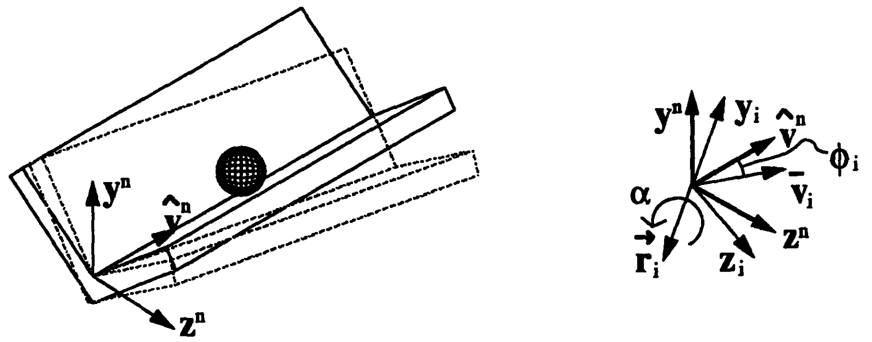


Figure 3.12: Detectable misalignment for the two constraint configuration

The detected misalignment is again given by a rotation about the vector $\vec{r}_i$, but here the vector lies in the 'yz' plane, and is given by:

$$\vec{r}_i = \hat{v}^n \times \bar{v}_i \qquad (3.16)$$

and the angle of misalignment is given by:

$$\phi_i = \arccos(\hat{v}^n \bullet \bar{v}_i) \qquad (3.17)$$

The components of the $\vec{r}_i$ vector can be written as:

$$\vec{r}_i = \begin{bmatrix} 0 \\ \sin(\alpha) \\ \cos(\alpha) \end{bmatrix} \qquad (3.18)$$

The detected workpiece orientation is represented by the matrix $\mathbf{R}_{wp}$. Here the 'x' component of $\vec{r}_i$ is zero and thus $\mathbf{R}_{wp}$ is given by:

$$\mathbf{R}_{wp} = \begin{bmatrix} c\phi & -s\phi\, c\alpha & s\phi\, s\alpha \\ s\phi\, c\alpha & c\phi + (1-c\phi)\, s^2\alpha & (1-c\phi)\, s\alpha\, c\alpha \\ -s\phi\, s\alpha & (1-c\phi)\, s\alpha\, c\alpha & c\phi + (1-c\phi)\, c^2\alpha \end{bmatrix} \qquad (3.19)$$

In the two constraint configuration successful implementation of motion requires maintaining contact with both constraint surfaces. Each surface provides a unilateral constraint, meaning that it is possible to push against the surface but attempting to pull in the direction of the constraint will result in a loss of contact. There exists a range of force directions that will maintain contact with both surfaces, and as with obstacle avoidance human inconsistency identifies a range of acceptable robot trajectory. Variations between demonstrations in the direction of force are unnecessary to achieve the task and are thus interpreted as human inconsistency. The acceptable region of force direction is defined by the acute angle between the directions of force from different demonstrations, thereby ensuring that a positive component of force is applied to both surfaces, as shown in Figure 3.13.

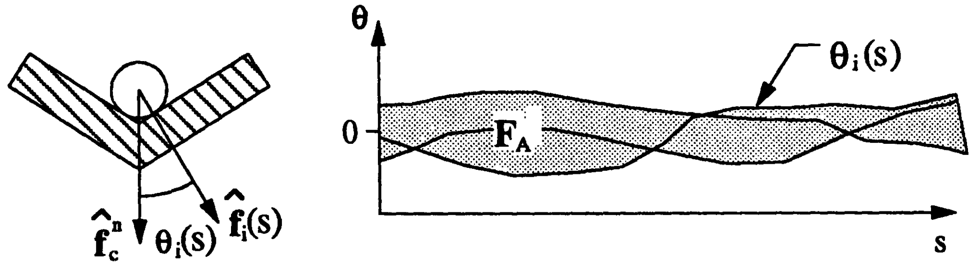


Figure 3.13: Range of Acceptable Force Directions, F_A.

As with the single constraint configuration, to compare different demonstrations the human trajectory is transformed to the coordinate system aligned with the detected workpiece orientation. Accordingly, the force vector in the detected workpiece coordinate system, $\mathbf{f}'(s)$, is given by:

$$\mathbf{f}'(s) = \begin{bmatrix} 0 \\ f'_y(s) \\ f'_z(s) \end{bmatrix} = \mathbf{R}_{wp}^T \hat{\mathbf{f}}_{c,i}(s) \qquad (3.20)$$

and the direction of force is:

$$\theta_i(s) = \arctan\left(\frac{f'_z(s)}{f'_y(s)}\right) \qquad (3.21)$$

The demonstrated direction of force is plotted versus the distance traveled as shown in Figure 3.13. The acceptable range of force directions is defined by the region enclosed by the demonstration trajectories and is designated by F_A. Thus a range of acceptable force directions is identified without using a geometric model of the workpiece that indicates the angle between the two constraint surfaces. As in the single constraint configuration, it is assumed that the demonstrator has no a priori knowledge of workpiece misalignment, and thus the beginning position of all the demonstrations are assumed to coincide. Furthermore, the region $\mathbf{F_A}$ indicates acceptable force directions even when undetected workpiece misalignments occur. In specific, the demonstrator succeeds at maintaining the desired contact despite the undetected rotation of the workpiece about the direction of motion. Consistent human success precludes undetectable rotations larger than the angle between the constraint surfaces, just as consistent obstacle avoidance by the demonstrator indicates that excessive workpiece misalignment does not occur.

As long as the contact state remains constant, the region of acceptable force directions does not change as the part moves within the contact state. Thus, one could argue that the region $\mathbf{F_A}$ should have a constant width as the distance 's' increases, defined by the maximum and minimum θ from all the demonstrations. However, it will be shown in Chapter 4 that when there is an undetectable contact state transition, it is necessary for the robot to stay within the boundary of $\mathbf{F_A}$ defined as a function of 's'. Accordingly, $\mathbf{F_A}$ is defined as a function of 's' for all contact states, which corresponds to the definition of $\mathbf{X_A}$ where the width of the region can vary throughout the contact state.

The conditions sufficient to guarantee robot success in the two constraint configuration have been identified. The range of workpiece misalignment is given by ϕ_i and $\vec{r}_i$. A successful robot controller is required to adapt to this detectable misalignment and modify their trajectory accordingly. Furthermore, to ensure that contact is maintained with the desired constraint surfaces, the robot force direction is specified to remain in the region $\mathbf{F_A}$ defined from human inconsistency. A range in the magnitude of the force is specified by requiring that the robot force be less than the maximum human force.

3.5. Robot Compliance Controller

In the previous sections demonstration data was used to identify performance requirements sufficient to guarantee success for constrained motion. In this section a robot controller is specified to achieve these conditions. The method of compliance control is used, where the robot gripper interacts with the environment as a linear spring. By selecting an appropriate robot compliance workpiece misalignment does not cause excessive forces and position errors.

In Section 3.5.1 the performance of a compliance controller for 3D translation is presented. The performance is evaluated in terms of position and force errors that occur as a result of workpiece misalignment. In Section 3.5.2 the results of demonstration data analysis are integrated with the results from the robot compliance controller analysis. The control parameters including robot compliance and reference trajectory are specified so that accuracy requirements specified by the human demonstrations are met by the robot.

A compliance controller can be implemented either actively or passively. An active controller adjusts the manipulator's joint torques to generate the desired actions at the end effector. The advantage of active control is that the robot compliance can be modified during a task. A passive controller is implemented by placing physical springs between the gripper and the manipulator arm. The advantage of passive control is its simplicity and that it can be easily applied to a standard position controlled robot, typical of the type used in industry. However, it is not possible to modify the stiffness of a passive compliance device during a task. In the analysis presented here a range of acceptable robot compliance is identified for each contact state. By identifying a range of compliance as opposed to a specific value, it increases the possibility that a given passive compliant device will be acceptable for all the contact states in the task.

3.5.1. Compliance Controller Performance

Hogan [1988] demonstrated that a compliant controller is stable when the manipulator is in contact with a passive environment, which is the typical case for assembly operations. It was further shown by Whitney [1982] that by selecting an appropriate robot compliance, the robot could adapt to workpiece misalignment. Whitney identified a range of robot compliance that could implement the task of inserting a peg into a hole, and the range of hole misalignment for which insertion will occur. Schimmels and Peshkin [1992] presented a method of specifying a robot control law in which the robot is programmed to model a damper. Both the work by Whitney, and Schimmels and Peshkin, identified a single robot controller which could implement motion consisting of a sequence of contact states. They also considered tasks which required both part translation and rotation. However, they assumed that the orientation misalignment of the workpiece was infinitesimal, and thus did not change the direction of contact forces.

The analysis presented here is simpler in some respects from the previously mention approaches, since the robot controller performance is evaluated for motion within a single contact state and the tasks analyzed require only part translation and no rotation. However, this analysis extends aspects of the previous approaches since the effect of orientation misalignment in the workpiece is evaluated, whereas the work by Whitney, and Schimmels and Peshkin assumed that the workpiece orientation misalignments are infinitesimal. Xiao [1991] does address finite orientation misalignments when the evaluating the performance of damping control. However, Xiao only considers force errors caused by orientation misalignment, and neglects position errors which is an essential performance criteria for assembly operations. The analysis presented here is for use with the PHD approach, however the results are also valid for modeling compliance control in model based programming methods. Indeed the results presented in this section are independent of the analysis of human demonstrations.

Compliance control is specified such that the force exerted by the robot, $\mathbf{f}^r$, is given by:

$$\mathbf{f}^r = \mathbf{K}(\mathbf{x}_{ref} - \mathbf{x}^r) \qquad (3.22)$$

where $\mathbf{x}^r$ is the robot gripper position, $\mathbf{x}_{ref}$ is the reference position, and $\mathbf{K}$ is a positive definite stiffness matrix. The analysis is restricted to translational motion, and thus the force and position vectors are 3x1 and the stiffness matrix is 3x3. Programming a compliance controller involves specifying the stiffness matrix, $\mathbf{K}$, and a reference trajectory $\mathbf{x}_{ref}(s)$; reaction with the environment specifies the robot's force and position. For example in unconstrained motion the robot trajectory is equal to the reference trajectory. We assume that the robot motion is quasi-static and dynamic forces can be neglected.

The complete robot program is implemented by moving the robot in a sequence of guarded moves between contact states. When contact with a new workpiece surface is detected, the reference trajectory is generated for motion within the new contact state. Since the reference trajectories are defined relative to a starting point on the constraint surface, workpiece translation misalignment normal to the surface does not effect robot performance. Thus, adaptation to workpiece translation misalignment occurs by detecting changes in the contact state. However, workpiece orientation cannot be detected at the initial contact with the surface. Accordingly, the reference trajectory is generated as if the workpiece was in the nominal orientation, and the robot compliance is used to adapt to orientation misalignment.

The desired robot position trajectory when the workpiece is in the nominal orientation is given by $\mathbf{x}^n(s)$, and the desired force trajectory without friction is given by $\mathbf{f}_c^n(s)$. The coefficient of friction can vary for each workpiece, and therefore friction is treated as a disturbance. The compliance controller reference trajectory is calculated by substituting the nominal trajectories for the robot trajectories in Equation 3.22. Thus, when the robot is in the nominal location the desired trajectories will be achieved. The reference trajectory is given by:

$$\mathbf{x}_{ref} = \mathbf{K}^{-1}\,\mathbf{f}^n + \mathbf{x}^n \qquad (3.23)$$

The coordinate system in which the robot performance is evaluated is aligned with the nominal surface orientation. Thus in a single constraint configuration the 'y' axis is normal to the surface, and in the two constraint configuration the 'x' axis is aligned with the direction of motion. The robot stiffness matrix is selected so that its eigenvectors are aligned with the coordinate system of the nominal surface and is given by:

$$\mathbf{K} = \begin{bmatrix} k_x & 0 & 0 \\ 0 & k_y & 0 \\ 0 & 0 & k_z \end{bmatrix} \qquad (3.24)$$

The details of the compliance controller analysis are presented in Appendix III, and just the results of this analysis are presented in Sections 3.5.1.1 and 3.5.1.2. The results are simplified to facilitate selection of a robot compliance, by assuming a small angle of misalignment in workpiece orientation and selecting the components of **K** such that k_x, k_y, and k_z are within the same order of magnitude. Small angle analysis is applicable to many assembly environments and still incorporates variations in the direction of motion and force (analysis that assumes infinitesimal orientation misalignment assumes no change in force and motion directions and thus would predict no robot errors for this analysis). For small angles of misalignment, rotations about the different axes can be evaluated independently and then combined in a linear fashion.

The evaluation of robot performance is done by comparing the desired robot motion to the actual robot motion, at a point on the trajectory corresponding to $\mathbf{x}^n(s)$ and$\mathbf{f}_c^n(s)$. The distance of $\mathbf{x}^n(s)$ from the initial location is given by d, and the magnitude of the nominal force is given by f^n. The rotation misalignments of the workpiece are given by ϕ_x, ϕ_y, and ϕ_z corresponding to rotations about the different axes.

3.5.1.1. Single Constraint Configuration

The results of the robot performance analysis for a single constraint are shown in Figure 3.14. The desired robot performance is to trace the shape of the nominal trajectory onto the surface of the workpiece despite misalignments. The maximum robot error occurs when the part is farthest from the starting position, which is where the error is evaluated. The 'x' axis is aligned with the vector from the starting position to the point of evaluation. As shown in Figure 3.14 workpiece rotation ϕ_x results in position errors in the 'z' direction, and rotation ϕ_z results in position errors in the 'x' direction. Friction is evaluated using a worst case basis. Accordingly, a friction component only appears in the 'x' direction, because fricxtion actually reduces position errors in the 'z' direction as shown in Appendix III. Orientation misalignments about the 'y' axis does not change the constraint directions nor effect robot performance, and thus is not shown the Figure 3.14.

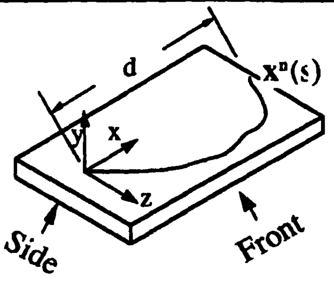
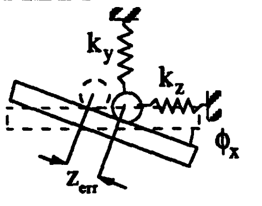
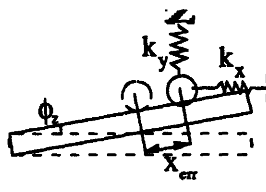
nominal orientation	ϕ_x misalignment	ϕ_z misalignment
	 Side View $z^r_{err} = \frac{\phi_x f^r_y}{k_z}$ $\lvert f^r \rvert = \lvert f^r_y \cos\phi_x \rvert$	 Front View $x^r_{err} = \frac{\mu f^r_y}{k_x} + \frac{\phi_z f^r_y}{k_x}$ $f^r_y = f^n + k_y \phi_z d$

Figure 3.14: Compliance Controller Performance with a Single Constraint
The dashed lines show the nominal workpiece orientation and the desired robot position, while the solid lines show the actual orientation and position.

3.5.1.2. Two Constraint Configuration

The results of the robot performance analysis for two constraints are shown in Figure 3.15. In the two constraint configuration the desired robot performance is to maintain contact with both constraints, since the position accuracy is determined by the workpiece. Here the 'x' axis is aligned with the nominal direction of motion. As shown in Figure 3.15 orientation misalignments ϕ_y and ϕ_z change the robot force, while misalignment ϕ_x does not change the constraint directions nor effect robot performance. Misalignments do modify the robot position in terms of the distance traveled, but not in the direction of motion. Since reaching the target is detected by the robot with a subtask termination condition, errors in the distance traveled do not effect robot performance and thus are not shown in Figure 3.14.

The nominal robot force is in the 'y' direction, yet misalignments in ϕ_y generate a force in the 'z' direction. As f^r_z increases relative to f^r_y, the direction of force changes. If the changes in direction of force is large enough loss of contact with one of the constraints will occur.

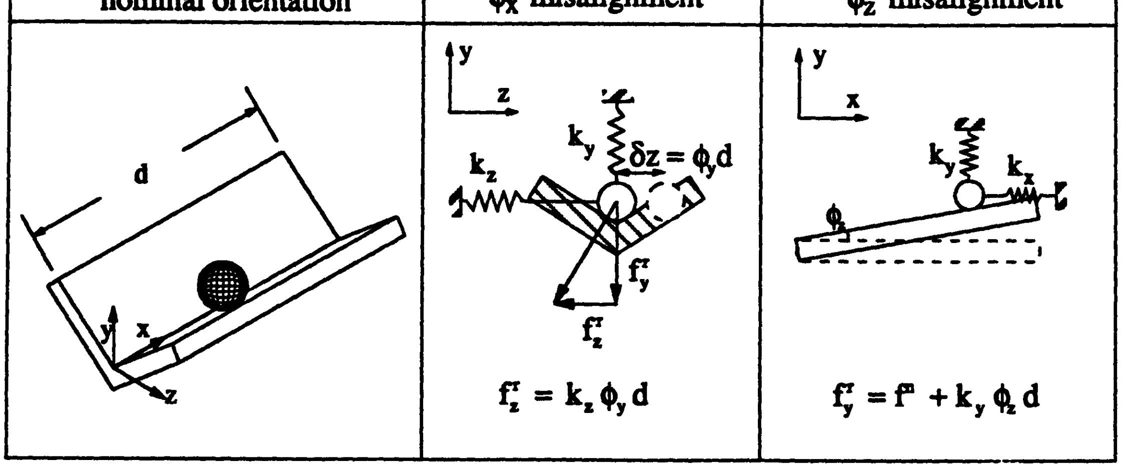


Figure 3.15:

Figure 3.15: Compliance Controller Performance with a Two Constraints
The dashed lines show the nominal workpiece orientation and robot position, while the solid lines show the actual orientation and position.

3.5.2. Selection of Compliance

In this section the parameters of the robot controller consisting of a reference trajectory, x_{ref}, and compliance, **K**, are selected. The reference trajectory is specified in terms of the nominal position and force trajectories, $x^n(s)$ and $f^n(s)$, as indicated by Equation 3.23. Here the analysis of human motion is combined with the robot analysis. Specifically, the range of acceptable motion and forces identified from the demonstration data are used to specify the accuracy requirements of the robot. In addition the range of workpiece misalignment identified from the demonstrations is used in specifying robot compliance. The analysis is applied first to the single constraint configuration and then the two constraint configuration.

3.5.2.1. Single Constraint Configuration

In the single constraint configuration, robot position errors occur, as shown in Figure 3.14. Nevertheless, the motion can be implemented successfully, since human inconsistency identifies a range of acceptable motion, X_A. As long as the robot error does not result in the trajectory leaving X_A, then obstacle avoidance can be guaranteed. Accordingly, a buffer with the boundary of X_A is used when generating the nominal robot trajectory, $x^n(s)$, as shown in Figure 3.16. The width of the buffer is designated by 'b' and identifies the accuracy requirement for the robot.

The value of 'b' is set to a default value, and if a trajectory to the target region cannot be generated because the minimum width of the region $\mathbf{X}_A$ is less than the default 'b', then a lower value of 'b' is selected through an iterative process until an acceptable value is found.

A non zero buffer value can be defined throughout $\mathbf{X}_A$, except near the starting location. The method used to define $\mathbf{X}_A$ in Equation 3.15 places the starting point from all the demonstrations at the same location on the constraint surface, leaving no room for a buffer. In actuality variation occurs in the demonstrated starting location, however these variations are undetectable once contact has been established and it is assumed that sensor information from prior to the contact is not used. Accordingly, the zero width of $\mathbf{X}_A$ at the starting location does not indicate perfect accuracy by the demonstrator nor require perfect accuracy from the robot. Therefore, for the initial segment of the robot trajectory, up to a distance designated by d_{start}, the buffer requirement is not imposed. A convenient method for selecting the value of d_{start} is to set it equal to the average distance between starting locations of the overall task when motion is unconstrained.

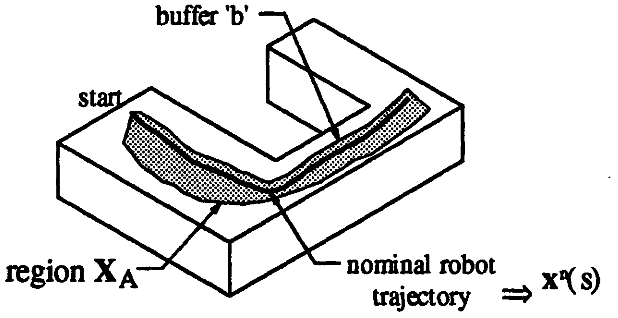


Figure 3.16: Nominal Robot Trajectory For a Single Constraint
The buffer between the nominal robot trajectory and the boundaries of $\mathbf{X}_A$ identifies the robot position accuracy requirement

In the single constraint configuration, robot position errors occur in the 'x' and 'z' directions and are a function of k_x and k_z, as shown in Figure 3.14. To select an appropriate value for robot stiffness, the values of k_x and k_z are set equal to each other and their value is designated by k_{xz}. A worst case analysis in terms of robot performance is done, and the maximum workpiece misalignment, ϕ, from all the demonstrations is given by ϕ_{max}. Without friction, ϕ contributes equally to error in the 'x' and 'z' directions, since $k_x=k_z$. Since friction increases the error caused by ϕ_z, the maximum error with friction occurs when ϕ_{max} occurs in the direction of

ϕ_z. The maximum friction coefficient, μ, is estimated from the demonstration data using Equation 3.7. In addition, the position error is a function of the robot force in the 'y' direction, f_y^r, which will be limited to a maximum value of f_{max}. Accordingly, the worst case robot error is:

$$x_{err}^r = \frac{\phi_{max} f_{max}}{k_{xz}} + \frac{\mu f_{max}}{k_{xz}} \tag{3.25}$$

To guarantee obstacle avoidance, the position accuracy requirement is applied conservatively by requiring that the position error in any direction be less than the buffer. Accordingly, the robot stiffness value for k_{xz} is selected so that x_{err}^r is less than or equal to 'b'. The compliance level is selected by setting x_{err}^r (as defined by the right hand side of Equation 3.25) to be less than or equal to 'b', and then solving for k_{xz}. The result is the following inequality:

$$k_{xz} \geq \frac{f_{max}(\phi_{max} + \mu)}{b} \tag{3.26}$$

Thus to limit the robot's position errors a minimum stiffness in the 'x' and 'z' directions is required. As can be seen from Equations in Figure 3.14 increasing either k_x or k_z independently reduces the position error. Accordingly, k_x can differ from k_z as long as they are both above the threshold level for k_{xz} shown in Equation 3.26. Furthermore, as the amount of human inconsistency in the demonstrations increases, larger buffer values 'b' can be incorporated into the region $\mathbf{X}_A$ and thus permitting a larger range for k_x and k_z. The analysis shows that if the human demonstrates low accuracy motion, then the robot accuracy can also be low and a low stiffness is sufficient.

In addition to avoiding obstacles, task success requires that contact be maintained with the surface without generating excessive forces. In the single constraint configuration, the force analysis is similar to the 2D case. When ϕ_z is positive, the contact force increases due to compression of the spring in the 'y' direction, while misalignment ϕ_x only decreases the force. To prevent damage to the parts the maximum allowable constraint force, f_{max}, is set is to the maximum force measured during the demonstrations. The worst case is considered by setting ϕ_z to ϕ_{max}, and excessive contact forces are avoided as long as:

$$f^n + k_y \phi_{max} d \leq f_{max} \tag{3.27}$$

To maintain contact with the surfaces the contact force is required to remain positive. Rotation ϕ_x causes force changes that are a function of $\cos\phi_x$ which cannot result in a negative force for small angles of misalignment. However, when ϕ_z is negative the surface moves away from the reference trajectory and the condition to guarantee that contact is maintained is given by:

$$0 \leq f^n - k_y \phi_{max} d \tag{3.28}$$

The nominal robot force magnitude, f^n, is selected to be $f_{max}/2$, and its direction is identified from the demonstration data which is given by $\hat{f}_c^n$. By substituting $f_{max}/2$ for f^n in inequalities 3.27 and 3.28, a limit for the robot compliance k_y is calculated:

$$k_y \leq \frac{f_{max}}{2\,\phi_{max}\,d} \tag{3.29}$$

Here the threshold identifies a maximum stiffness in the 'y' direction. Increases in workpiece misalignment and the distance 'd' lower the maximum acceptable robot stiffness.

The analysis has identified the parameters for a compliance controller for a motion constrained in a single direction, directly from the demonstration data. The stiffness matrix has been defined by ranges of acceptable values for k_x, k_y, and k_z. In addition nominal robot trajectories $\mathbf{x}^n(s)$ and $\mathbf{f}^n(s)$ have been identified, which can be substituted into Equation 3.23 to identify the robot reference trajectory.

3.5.2.2. Two Constraint Configuration

In the two constraint configuration, robot position errors do not occur. However to ensure this position accuracy it is necessary to maintain contact with both surfaces, which requires maintaining the appropriate force direction. To guarantee robot success it is also necessary to avoid excessive contact forces.

Workpiece misalignment can result in errors in the magnitude and direction of force as shown in Figure 3.15. However, as in the single constraint configuration, human inconsistency has identified an acceptable range of robot error, $\mathbf{F_A}$, as shown in Figure 3.17. The nominal robot force direction, $\mathbf{f}^n(s)$ is generated within the region $\mathbf{F}_A$, just as $\mathbf{x}^n(s)$ is generated within $\mathbf{X}_A$. A buffer in the direction of force, θ_b, exists as shown in Figure 3.17. Thus, as long as the error in the direction of the robot force does not exceed the angle θ_b, it is guaranteed that contact will be maintained with both surfaces.

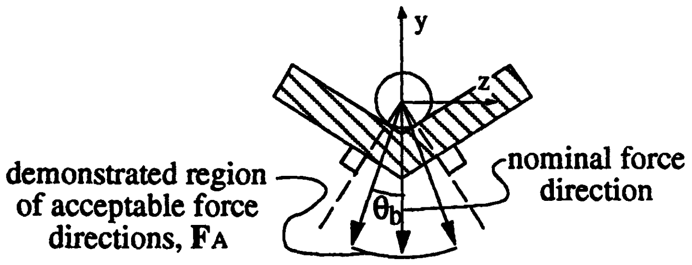

y
z
demonstrated region of acceptable force directions, $\mathbf{F}_A$
nominal force direction
θ_b

Figure 3.17: An Allowable Error in the Direction of Force is Given by θ_b.

The equations in Figure 3.15 show that misalignment ϕ_y results in a force in the 'z' direction of the magnitude $k_z\phi_y d$. In addition misalignment ϕ_z changes the magnitude of the force in the nominal direction (aligned with the 'y' axis) by $k_y\phi_z d$. To select an appropriate value for robot stiffness, the values of k_y and k_z are set equal to each other and their value is designated by k_{yz}. The maximum workpiece misalignment from all demonstrations is given by, ϕ_{max}, and its components in the 'y' and 'z' directions contribute equally to changes in f_y and f_z, since k_y=k_z. Accordingly, the maximum change in robot force due to workpiece misalignment is given by:

$$\Delta f = k_{yz}\,\phi_{max}\,d \tag{3.30}$$

For a worst case analysis it is assumed that Δf can occur in an arbitrary orientation in the 'yz' plane. The total robot force is given by combining the nominal force, $\mathbf{f}^n$, with $\Delta\mathbf{f}$, and the vector sum is depicted graphically in Figure 3.18. The possible robot force vectors can be drawn from the origin to any point on the circumference of the circle with radius Δf. The boundary of the region of acceptable force directions is shown by the lines offset by the buffer angle, θ_b, from the nominal force. In addition the boundary on force magnitude is shown by the arc of radius f_{max}.

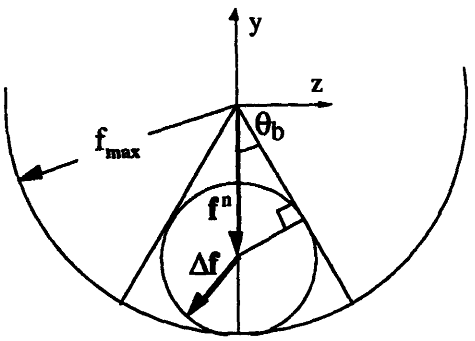


Figure 3.18: Graphical depiction of the boundaries on the acceptable directions and magnitudes of robot force.

The circle with the largest radius, Δf, that can fit within the acceptable force boundaries, identifies the widest range of acceptable values for k_{yz}. Such a circle is shown in Figure 3.18, where its circumference touches the boundaries defined by both θ_b and f_{max}. The trigonometric relationship is defined by the equations: $\sin(\theta_b) = \Delta f/f^n$, and $\Delta f + f^n = f_{max}$. These two equations are solved for the unknowns Δf and f^n, and the results are given by:

$$\Delta f = \frac{f_{max} \sin(\theta_b)}{(1 + \sin(\theta_b))} \qquad (3.31)$$

and

$$f^n = \frac{f_{max}}{1 + \sin(\theta_b)} \qquad (3.32)$$

The robot stiffness that will prevent the change in force from exceeding Δf is given by substituting Equation 3.30 into Equation 3.31 with the result:

$$k_{yz} \leq \frac{f_{max} \sin(\theta_b)}{\phi_{max}\, d\,(1 + \sin(\theta_b))} \qquad (3.33)$$

Here an upper limit is placed on the values of k_y and k_z. Higher stiffness levels could result in unacceptable errors in robot force direction and magnitude. As can be seen from the equations in Figure 3.15 decreasing either k_y or k_z independently reduces the change in robot force. Accordingly, k_y can differ from k_z as long as they are both beneath the threshold level for k_{yz} shown in Equation 3.33.

The analysis has identified the compliance controller parameters for motion constrained in two directions, directly from the demonstration data.

The stiffness matrix has been defined by ranges of acceptable values for k_y and k_z in Equation 3.33, and the value of k_x is not restricted in this configuration. In addition the direction of the nominal force, $\mathbf{f}^n$, is selected to be in the middle of the region $\mathbf{F}_A$, and its magnitude is given by Equation 3.32. The nominal position trajectory, $\mathbf{x}^n(s)$, is a straight line in the direction $\hat{\mathbf{v}}^n$ which was identified from the demonstration data. Thus, the values for $\mathbf{x}^n(s)$ and $\mathbf{f}^n(s)$ have been identified, and can be substituted into Equation 3.23 to identify the robot reference trajectory.

3.5.2.3 Constraint Configuration Ambiguity

As indicated in Section 3.4.2, if each demonstration consists of straight line motion with a force applied in a constant direction, i.e. $\hat{\mathbf{v}}(s)$ and $\hat{\mathbf{f}}_c(s)$ are constant, then it is not possible to determine whether the motion is constrained in one or two directions. Under these circumstances, the sufficient conditions for robot success are calculated for both constraint configurations.

Variations between demonstrations in the direction of motion and force is calculated by both Equations 3.12 and 3.17, which provide values for rotations about all three axes, ϕ_x, ϕ_y, and ϕ_z. The interpretation of variation between demonstrations depends on which configuration one is considering, as shown in Figure 3.20. If one assumes that there is a single constraint, then variation ϕ_x corresponds to workpiece misalignment, and variation ϕ_y corresponds to human inconsistency. The reverse is true if one assumes there are two constraints. Figure 3.19 shows the alternate interpretations for a rotation ϕ_y in the demonstration data.

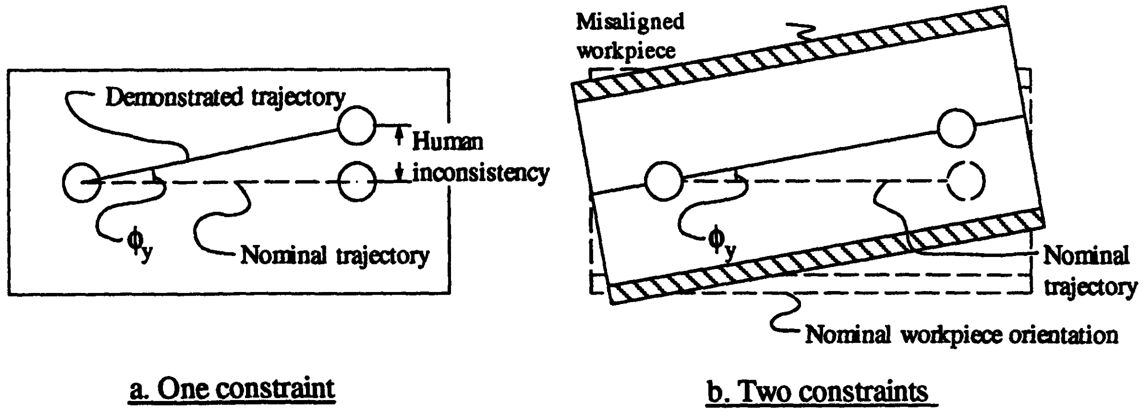


Figure 3.19: Rotation of a straight line position trajectory about the 'y' axis (top view)

In each configuration, human inconsistency defines a region of acceptable motion or force. With a single constraint the straight trajectories fan out from the starting point to generate a wedge shaped obstacle free region, as shown in Figure 3.19. The acceptable robot position error increases with the distance traveled, and the buffer value is equal to '$\phi_{y,max}$ d'. In the two constraint configuration, the buffer in the direction of force is given by the maximum rotation about the 'x' axis, $\phi_{x,max}$. The equations in Figure 3.20 show the buffer magnitudes and the robot errors due to workpiece misalignment for both configurations.

Configuration	Interpretation of ϕ_x	Interpretation of ϕ_y
One constraint	workpiece misalignment $z^r_{err} = \frac{\phi_{x,max} f^r_y}{k_z}$	human inconsistency $b = \phi_{y,max}\, d$
Two constraints	human inconsistency $\theta_b = \phi_{x,max}$	workpiece misalignment $f^r_z = k_z \phi_{y,max}\, d$

Figure 3.20: Interpretation of demonstration data when $\hat{v}(s)$ and $\hat{\underline{f}}_c(s)$ are constant.

To identify a robot compliance controller to perform the straight line motion, the robot errors due to workpiece misalignment are compared to the buffer size. For a single constraint configuration, the robot position error is required to be less than the buffer to ensure obstacle avoidance. In straight line motion, errors in the 'x' direction are aligned with the direction of motion and do not cause the robot trajectory to move outside the obstacle free region. Accordingly, only the position error in the 'z' direction is compared to the buffer size, which are shown in Figure 3.20. The worst case position error could occurs if f^r_y is equal to f_{max}, and the condition to ensure obstacle avoidance results in the following lower limit for k_z.

$$k_z \geq \frac{\phi_{x,max} f_{max}}{\phi_{y,max}\, d} \qquad ((3.34)$$

In the two constraint configuration, an upper limit on k_z is required to prevent the error in force direction from exceeding θ_b. The maximum error in force direction is caused by workpiece misalignment, $\phi_{y,max}$. For straight line motion, the limit on robot compliance k_z that prevents excessive errors in force direction is given by substituting into Equation 3.33: $\phi_{x,max}$ for the buffer angle and $\phi_{y,max}$ for ϕ_{max}, and applying the small angle approximation.

$$k_z \leq \frac{\phi_{x,max}\, f_{max}}{\phi_{y,max}\, d} \qquad (3.35)$$

The force error in the 'y' direction caused by workpiece misalignment ϕ_z is the same in both configurations (Figures 3.14 and 3.15). The analysis for the two constraint configuration identifies a condition for k_y that is equal to k_z, for the worst case misalignment about the 'y' and 'z' axis (Equation 3.33). If $\phi_{z,max}$ is less than $\phi_{y,max}$, then Equation 3.35 is equivalent to Equation 3.33, and k_y can be set equal to k_z. However, if there is a larger rotation about the 'z' axis, a lower stiffness for k_y is selected to prevent the contact force from exceeding f_{max}.

Although the interpretation of ϕ_x and ϕ_y depends on the constraint configuration, their effect on the limits of k_z, shown in Equations 3.34 and 3.35, is the same. A large value of ϕ_x indicates that a large robot position error in the 'z' direction can occur if there is a single constraint, or that there is wide range of acceptable force directions for the two constraint configuration. In both configurations, a large ϕ_x corresponds to increasing the value of k_z. In a similar fashion a large value of ϕ_y indicates that a large position buffer is available if there is a single constraint, or that there can be a large robot force error in the 'z' direction. In both constraint configurations a large ϕ_y corresponds to decreasing the value of k_z.

The upper and lower limits shown in Equations 3.34 and 3.35 are equal to each other, and the value of k_z can be set equal to these limits to satisfy both constraint configurations. However here k_z is a function of the distance traveled. Theoretically one could use active compliance control to modify the compliance along the trajectory. However, a more practical solution can be implemented with a two stage approach.

In the first stage the robot position error requirement can be relaxed, since as indicated in Section in 3.5.2.1 all the demonstrations do not start at the same location for the single constraint configuration. The initial allowable error is set equal to the buffer value after motion of a distance d_{start} from the starting position. For motion up to d_{start} the following stiffness satisfies the force requirement for the two constraint configuration shown in Equation 3.35, and prevents position errors greater than the initial allowable error for the single constraint configuration.

$$k_z = \frac{\phi_{x,max} f_{max}}{\phi_{y,max} d_{start}} \quad \text{for} \quad 0 \leq d \leq d_{start} \qquad (3.36)$$

Once the robot has moved a measurable distance (less than d_{start}) the directions of $\hat{v}(s)$ and $\hat{f}_c(s)$ can be calculated. The detected directions of motion replace the nominal directions, and are used to update the compliance controller reference trajectory. Thus during the second stage of motion, adaptation is performed by explicitly using sensor measurements to adjust to workpiece misalignment.

The analysis presented in this chapter identifies robot performance requirements which can guarantee success, but does not ensure that a single compliance controller can be found to satisfy these requirements. When using passive compliance control, only a single compliance is available for the complete sequence of contact states in the task. Accordingly, it may be advantageous to use the two stage approach even if the number of constraints are known. This approach allows for a larger range of robot compliance, and makes it more likely that a single passive complaint device can be used for the complete sequence of contact states in the task.

3.6. Experimental Results

The example task used in the experiments was introduced in Chapter 1, and is shown in Figure 3.21. The task is performed with a sequence of six contact states. The demonstrator familiarized themselves with the task with their eyes open, and practiced with their eyes closed. After the practice, ten demonstrations were performed with the demonstrator blindfolded. The demonstrator was requested to perform the motion with the same sequence of contact state, but otherwise was not told how the data would be analyzed, to prevent intentional force or motion aimed at assisting the analysis.

Misalignments in workpiece position and orientation were introduced between demonstrations. The demonstrator knew that misalignments would occur, but since their eyes remained closed they were not aware of the extent of the misalignment. The sequence of contact states used, allowed the demonstrator to successfully implement the task, despite significant workpiece misalignment. The demonstrator could reach the target position, as long as workpiece misalignment does not cause the demonstrator to miss the initial contact state, and the orientation misalignment is not excessive. The maximum translation misalignments were 20mm in the 'x' direction, 40mm in the 'y' direction, and 8mm in the 'z' direction. The maximum orientation misalignment was 1° about the 'x' axis, 1° about the 'y' axis, and 6° about the 'z' axis.

The teaching gripper is the same one used for the obstacle avoidance task presented in Chapter 2. In this application measurements from the six axis force/torque sensor (manufactured by Zebra Robotics Inc.) were used. The position and force were both sampled at a frequency of 37 Hz. The spherical part was held continually without slip throughout the demonstration.

As in Chapter 2, the position of the sphere center is calculated to give the position trajectories, $x_i(t)$. The force measurements at the force sensor are the same as at the sphere center. However, it is necessary to remove the gravitational load. The orientation measurement from the position sensor identifies the gripper orientation relative to the gravitational field where the weight of the gripper is subtracted, to provide the force trajectory $f(t)$. The position and force trajectories from one of the demonstrations are shown in Figure 3.21. The force vectors are plotted at every third sample point, where the length of the vector corresponds to the magnitude and the maximum force is 16 Newtons. The force $f(t)$ includes a friction component, and as can be seen by the component of force in the direction of motion in Figure 3.21.

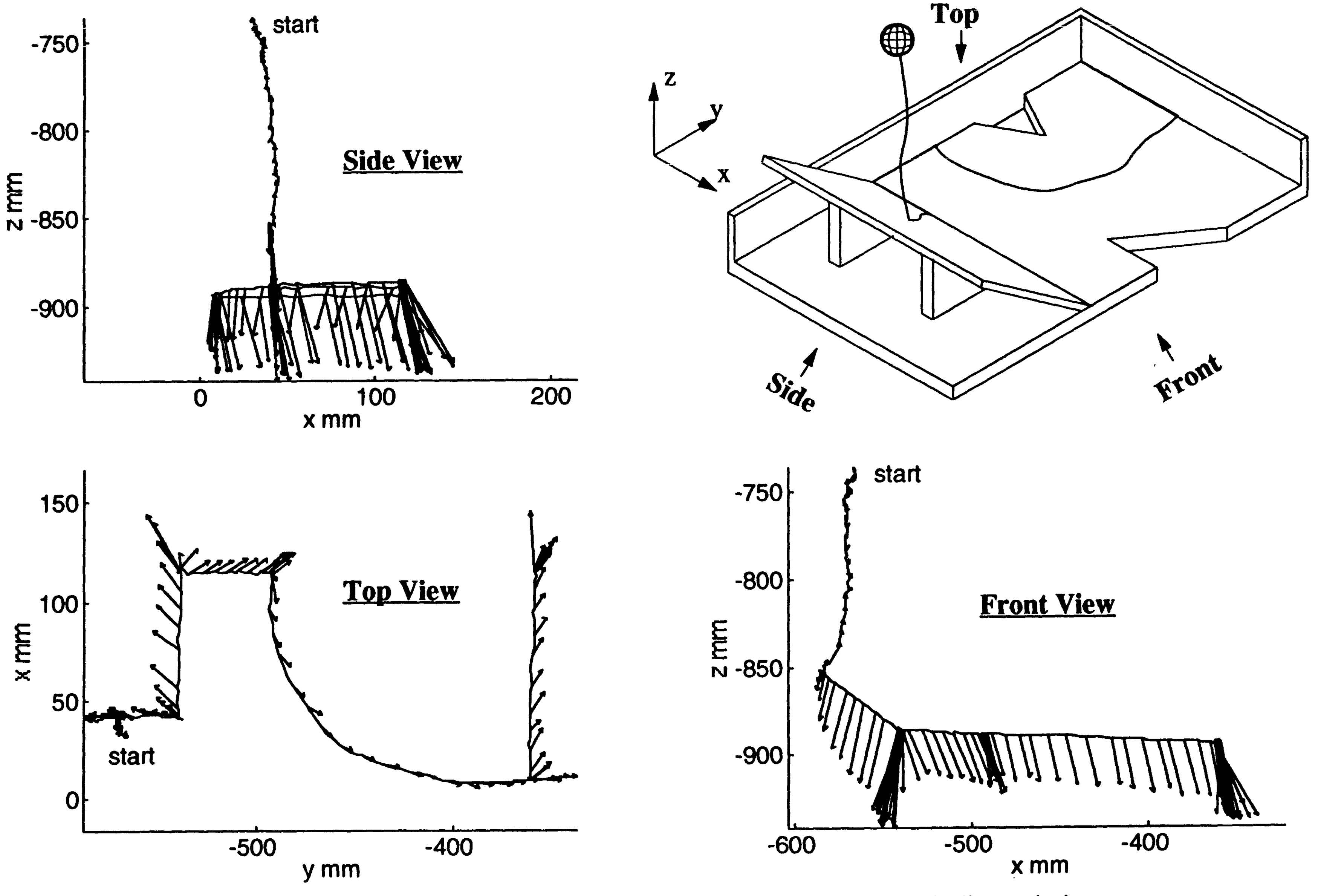


Figure 3.21: Measured Position Trajectory With Force Vectors Including Friction

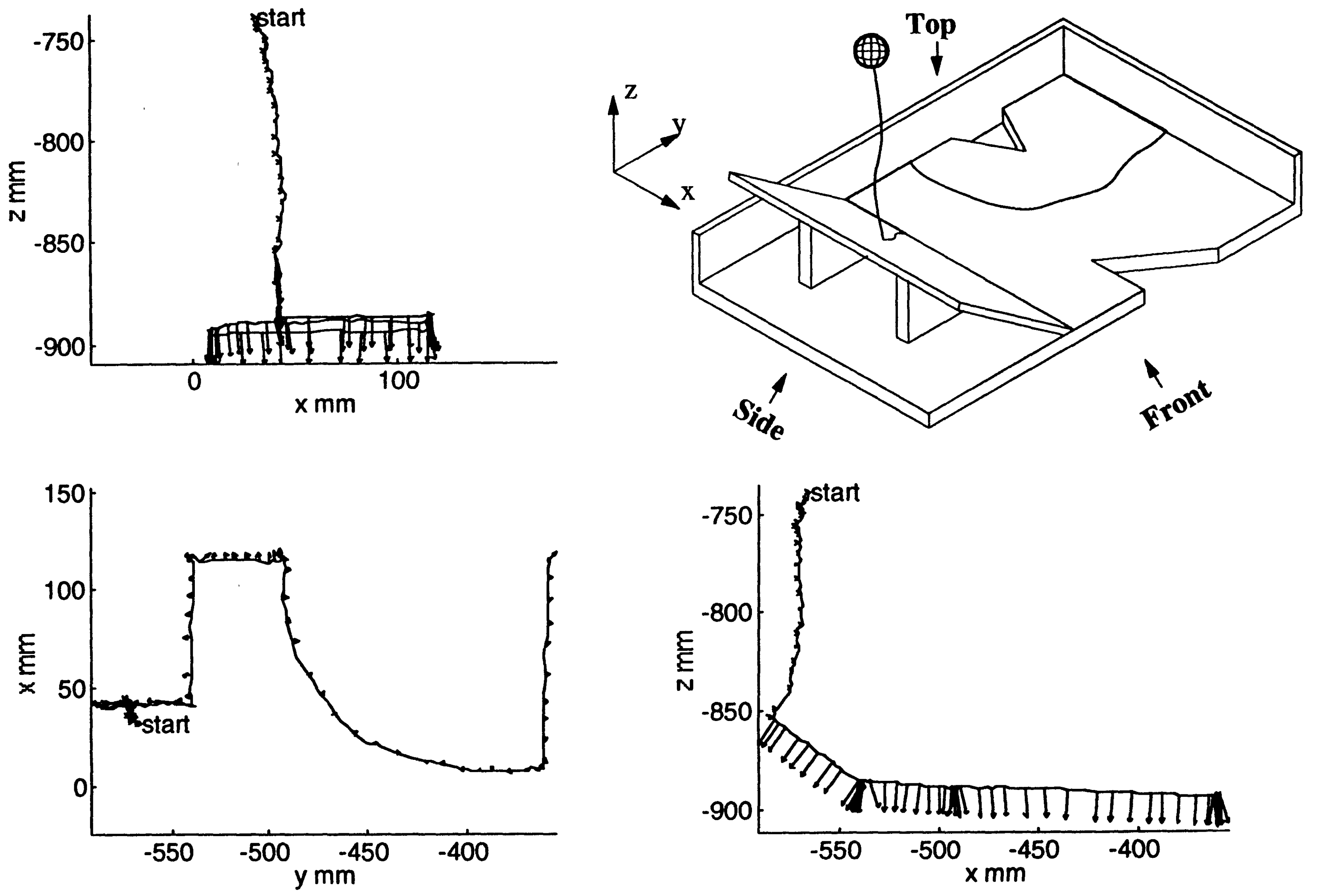


Figure 3.22: Position Trajectory With Constraint Force Vectors (Friction Removed)

The friction component is identified and subtracted from $\mathbf{f}$, using Equations 3.5 and 3.6, to provide the constraint force $\mathbf{f}_c(t)$. The normalized constraint force vectors, $\hat{\mathbf{f}}_c(t)$, are plotted in Figure 3.22, which shows that $\hat{\mathbf{f}}_c(t)$ is normal to the constraint surfaces. The length of the vectors in the Figure do not reflect the magnitude of $\mathbf{f}_c$, but the normalized components of $\hat{\mathbf{f}}_c$. The maximum constraint force from all the demonstrations is 12 Newtons, and is used as the maximum allowable robot constraint force; thereby ensuring that the total robot force will not exceed the total demonstrated force (as long as the coefficient of friction is not larger than the maximum coefficient of friction in the demonstrations). The maximum coefficient of friction is identified using Equation 3.7 and is equal to 0.66.

Each demonstration trajectory is segmented into subtasks using an approach to be presented in Chapter 4. In this task, each subtask corresponds to a single contact state. The analysis presented in this chapter is applied by comparing demonstrations for each contact state. Motion from the fifth contact state is shown in Figure 3.23.

Without using the model, the constraint configuration for the fifth contact state is identified as a single direction of constraint, since $\hat{\mathbf{f}}_c$ is constant and $\hat{\mathbf{v}}$ varies. Accordingly, task success requires that the robot trajectory remain within the obstacle free region bounded by the demonstrations. Workpiece misalignment is identified using Equation 3.11. The maximum detected orientation misalignment is 1.5°, which corresponds to the misalignments about the 'x' and 'y' axes (misalignment about the 'z' axis is undetectable in this contact state). Each demonstration is transferred to a coordinate system relative to the detectable surface location using Equation 3.15. The trajectories relative to the surface, $\mathbf{x}'_i(s)$, are plotted in Figure 3.23 in the plane of x' and z'.

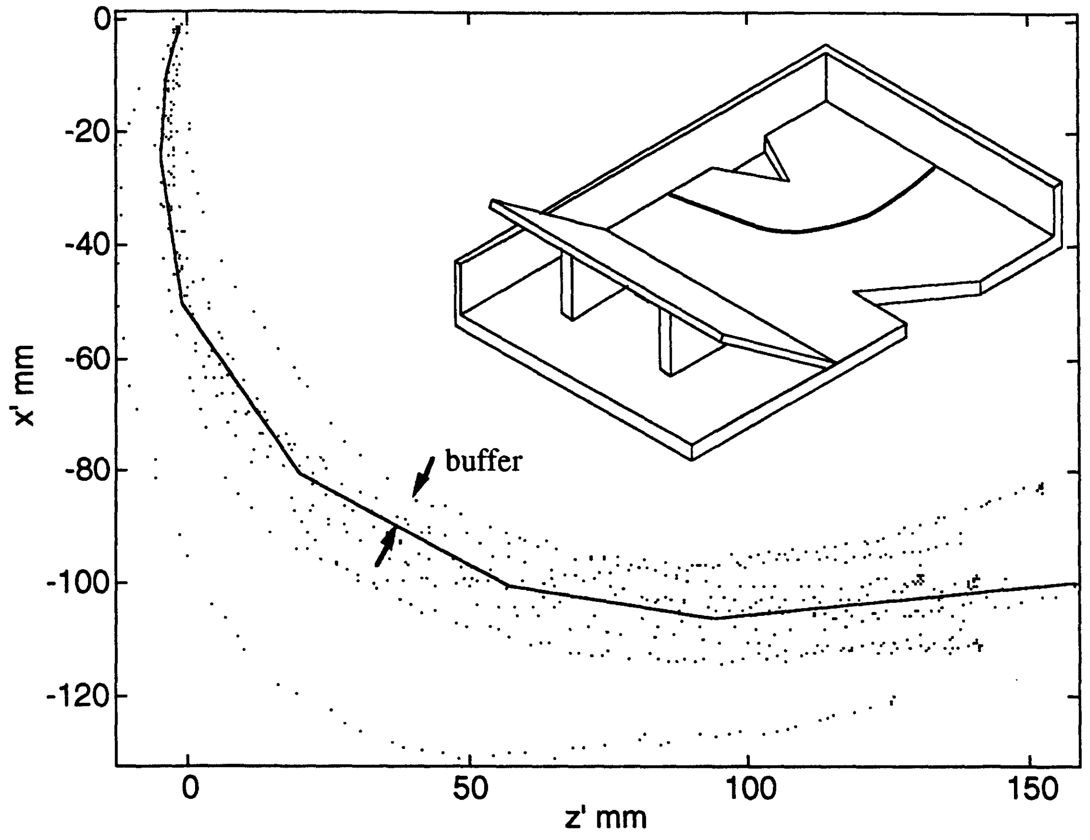


Figure 3.23: Position Trajectories in the Fifth Contact State
Motion within the fifth contact state is shown in the inset diagram of the workpiece. The dotted lines are the demonstrated trajectories relative to the detectable surface location, and the solid line is the robot trajectory generated with a buffer of 10mm.

The trajectories $x_i'(s)$ defined a region of acceptable motion, and a robot trajectory is generated within that region using the methods presented in Chapter 2. The robot trajectory is generated with a 10mm buffer with the boundary of the obstacle free region. The value of d_{start} is set equal to 15mm, which was the average variation in the task starting position. As indicated in Chapter 5, the end of the robot trajectory is not specified as the average ending position; instead the robot motion continues until the next contact state is reached. Accordingly, once the robot trajectory passes the end of the shortest demonstration, the direction of motion is kept constant throughout the remainder of the region of acceptable motion.

A minimum compliance in the 'x' and 'z' directions is specified by Equation 3.26, which limits the robot position error to less than the buffer value. The calculation results in:

$$k_{xz} \geq 0.8 \text{ N/mm} \qquad (3.37)$$

A maximum compliance in the 'y' direction is specified by Equation 3.29, to prevent the contact force from exceeding the maximum demonstrated constraint force. The calculation depends on the maximum distance from the from the contact state starting position, which is 189mm. The result is:

$$k_y \leq 1.2 \text{ N/mm} \qquad (3.38)$$

Thus robot compliance limits and a nominal robot trajectory have been identified for the fifth contact state. This process is repeated for each contact state in the task.

3.7. Summary

In this chapter demonstration data is used to specify a robot controller that can adapt to workpiece misalignments. The analysis is applied to 3D translation for motion within a single contact state, and evaluates the effect of orientation misalignment in the workpiece. Robot position accuracy is specified to avoid obstacles, and force accuracy is specified to maintain contact the constraint surfaces and avoid damage to the parts. Robot success is guaranteed as long as workpiece misalignment is not larger than that encountered during the demonstrations. This adaptation at the level of force and position trajectories may be performed subconsciously by the human. Nevertheless it is essential to explicitly specify such adaptation when programming a robot, and the PHD approach presents an automatic method for specifying these robot performance requirements.

The PHD approach presented in Chapter 2 for unconstrained motion, that uses human inconsistency to identify an obstacle free region, is extended in this chapter to the case of constrained motion. If the demonstrator consistently performs aspects of the task with similar position and force trajectories, then these components of the trajectories are transferred to the robot program. However, when the human demonstrates a wide range in motion or force, it is an indication that high robot accuracy is not necessary in these regions. In the single constraint configuration, the position trajectories identify an obstacle free region on the constraint surface. In the two constraint configuration, the demonstrated directions of force identify a range of acceptable force direction for which contact with both surfaces is maintained. Finally, in both constraint configurations, the magnitude of the demonstrated force identifies a range of forces for which damage to the parts can be avoided.

While the human is performing the task their motion includes both adaptation and inconsistency, and to distinguish between them a theorem is presented. Adaptation occurs in response to detectable variation in the environment, and aspects of the motion and force trajectory that correlate to detectable workpiece misalignment are interpreted as adaptation. The demonstrations and tasks are structured such that it is possible to quantify all the sensory information used to adapt to the environment. Specifically, the demonstration is restricted to using position and force information measured by the teaching gripper, and adaptation consists of modifying the trajectory to correlate with workpiece misalignment. The human adapts to the environment by modifying their trajectory to maintain contact with the constraint surfaces and this adaptation is incorporated into the robot performance requirements. In addition, within each contact state configuration, their exists a direction of orientation misalignment that is undetectable. However it is not necessary for the robot to adapt to these undetectable misalignments, since the human consistently succeeds without the ability to adapt to these misalignments either.

The method used to specify a robot controller does not attempt to duplicate the internal control method the human uses. Rather the demonstrated trajectories identify conditions sufficient to guarantee robot success. In this analysis a robot compliance controller is selected. The performance of compliance control is quantified in terms of position and force errors due to workpiece orientation misalignment. The results of the compliance controller analysis are combined with the analysis of the demonstration data, to identify a robot controller directly from the demonstrations. A range of acceptable robot compliances are identified, which prevent excessively high stiffness that would result in unacceptable force errors and excessively low stiffness that would result in unacceptable position errors

The PHD method does not utilize a geometric model of the task, but instead relies on the demonstration data. Accordingly, the constraint configuration is not explicitly provided, and in certain circumstances it is not possible to determine if motion is constrained in one or two directions. It is shown that the methods used in the analysis provide for conditions that can guarantee robot success, without explicit knowledge on the number of constraints. Accordingly, this approach extends previous research which limited the task geometry so that it would be possible to explicitly identify the number of constraints.

CHAPTER 4

Segmentation of Task Into Subtasks

4.1. Introduction

Segmenting a task into a sequence of subtasks allows the task to divided into simpler components, and each subtask can then be implemented with a simpler controller. Accordingly, it is advantageous in both model based programming and PHD to define subtasks. In model based methods, segmentation can be done using part geometry and the task objective. However, in the PHD approach the segmentation is done using only the demonstration data.

In this Chapter, an algorithm is presented that segments each demonstrated trajectory into a sequence of subtasks. The subtask definition corresponds to the robot controllers defined in Chapter 3, where each robot controller is specified for motion within a single contact state. Accordingly, the algorithm segments the demonstrated trajectory at each contact state transition that is detected. However, some of the contact state transition cannot be detected from the demonstrations data. For these cases it is shown that the robot controllers defined in Chapter 3 for a single contact state, can also perform motion consisting of two contact state if the transition between them is undetectable. Finally, a method is presented to ensure corresponding segmentation of the different demonstrated trajectories.

It should be noted, that the analysis used to segment a task into subtasks is different from the analysis used to identify subtasks termination conditions, which allow the robot to switch from one subtask to the next and is addressed in Chapter 5.

4.1.1 Background

Segmenting a task into subtasks is useful in both model based programming and PHD. For model based programming, Lozano-Pérez, Mason, and Taylor [1984] segment a task into sections based upon how much of the task a robot controller can perform with a predefined trajectory and gain matrix. They specify a robot controller, programmed to react as a linear damper, and identify the length of motion that the controller can perform, which can include motion that covers multiple contact states. Whitney [1982], and Schimmels and Peshkin [1992] also present robot controllers that can achieve motion that contain multiple contact states.

An alternative model based segmentation approach is presented by Asada and Hirai [1989], and Desai and Volz [1989]. Here the task is segmented at each contact state transition. This method is well suited for implementing hybrid force/position control, where the direction of admissible motion and force vary at each contact state.

The method presented in this Chapter combines components from both of the above approaches used in model based programming. The task is segmented into subtasks at contact state transitions, where these transitions can be detected from the demonstration data, which corresponds to the approach of Asada and Hirai [1989], and Desai and Volz, [1989]. In the case of undetectable transitions, it is shown that the robot controller can perform the motion for both contact states. Thus ensuring that each subtask can be implemented by the robot controller, as with the approach of Lozano-Pérez et al [1984].

Research in robot programming methods that are based on human demonstrations, has also addressed the subject of segmenting a task into subtasks. Ikeuchi, Kawade, and Suehiro [1993], define a new subtask when they detect that a part has moved, yet do not address the details of how the human moves the part while adapting to the environment, which is the subject of this thesis. Asada and Izumi [1987] define a new segment at right angles in the position trajectory, yet this approach is limited to workpieces where all the surfaces are at right angles to each other.

Other approaches are based on assumptions of how the human performs the task. Kuniyoshi, Inaba, and Inoue [1992] segment the task when they "detect a qualitative change of the hand movement," such as exceeding a threshold velocity or change in direction. A similar approach is implemented by Takahashi, Ogata, and Moto [1993], yet the results are not satisfactory in all demonstrations, and difficulty in selecting appropriate threshold levels is reported. Indeed, the demonstrator may pause at different points in the task for no apparent reason, and thus segmenting at each pause will not yield consistent results.

Another approach that is based on a model of how the human controls their motion, is presented by Delson and West [1992]. In that approach it is assumed that the demonstrators performs each subtask with a constant compliance and a straight line reference trajectory, yet the demonstrations were not all segmented at corresponding points in the task.

A more satisfactory segmentation approach is presented in this Chapter, that does not rely on when the human pauses or on the internal method the human uses to control their arm impedance.

4.2. Segmentation Algorithm

The category of tasks addressed in this chapter is the same as in Chapter 3. The task objective is to place a spherical part in a desired location relative to a polyhedral workpiece, and the assumptions outlined in Section 3.1.2. apply. The part is held continually by the gripper throughout the task. Accordingly, any opening or closing of the gripper indicates the beginning and end of the task. (The gripper status is detected from pressure sensors as shown in Figure 1.1).

A single task consists of sections of unconstrained motion, addressed in Chapter 2, and constrained motion, addressed in Chapter 3. Identifying when the part becomes constrained is straightforward. When the force magnitude exceeds the level of noise in the measurement, it indicates that contact has been established. A more challenging analysis is to segment sections of constrained motion into appropriate subtasks, which is the focus of this Chapter.

The criterion for valid segmentation, is that a robot controller can be specified that can perform each subtask. The analysis in Chapter 3 identifies a robot controller for a section of motion in which the part is constrained by the same workpiece surfaces, i.e. motion within a single contact state.

Accordingly, a natural approach is to segment the task at each contact state transition. However, a difficulty with this approach is that not all the changes in contact state can be detected from the demonstration data. A segmentation algorithm is presented in Section 4.2.1 which identifies contact state transitions when possible. In Section 4.2.2 it is shown that undetectable contact state transitions do not need to be detected, and that the robot controller identified for a single contact state will be able to perform the motion for both contact states.

4.2.1 Identifying Contact State Transitions

In 3D translation, motion is possible in only three contact configurations: unconstrained motion where the contact force is zero, a single of direction of constraint where the direction of constraint force, $\hat{\mathbf{f}}_c$, is constant, and two directions of constraint where the direction of motion, $\hat{\mathbf{v}}$, is constant. Starting at the beginning of a demonstration, the first subtask is identified as the longest segment in which either $\hat{\mathbf{f}}_c$ or $\hat{\mathbf{v}}$ is constant (including $\hat{\mathbf{f}}_c$ equal to zero). This process will identify most contact state transitions, and is repeated until the complete demonstration is segmented into a sequence of subtasks.

To determine if the constraint force is constant it is necessary to evaluate $\hat{\mathbf{f}}_c$ when the velocity is equal to zero as well as when the part is moving. It is possible (and indeed likely) that the part will stop when it contacts a new constraint surface. However, the part can also stop in the middle of a contact state. Accordingly, it is necessary to distinguish between the two scenarios in which the part stops, shown in Figure 4.1, in order to detect all changes in $\hat{\mathbf{f}}_c$.

In Figure 4.1a, no change in the contact state occurs but the demonstrator stops. As shown, motion does not occur as long as the applied force, $\mathbf{f}$, is inside the friction cone, which is known as the sticking condition. The demonstrator may have stopped simply to pause or because of a local increase in the friction coefficient which results in a stop and go motion referred to a sticktion. If the friction and constraint forces could be calculated at this point, it would shown that the direction $\hat{\mathbf{f}}_c$ is still normal to the surface of the subtask, indicating that motion is within the same contact state. However, the friction force cannot be calculated by projecting $\mathbf{f}$ onto the direction of motion, as in Equation 3.5, since $\hat{\mathbf{v}}$ is not defined when the part stops.

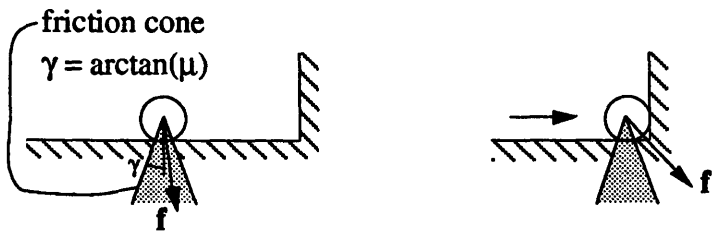


Figure 4.1: Two Scenarios in Which the Motion Stops

In Figure 4.1b, motion stops when the part contacts a new constraint surface. Upon contact with the new surface the directions of admissible constraint force change. Accordingly, the direction of **f** can be outside of the previously define friction cone, without motion occurring. Here the direction of $\hat{\mathbf{f}}_c$ does change, and a successful segmentation algorithm should be able to detect this change and indicate a new contact state. Distinguishing between the sticking condition and a new contact is possible by evaluating whether the applied force is within the friction cone. Indeed the demonstrator is only able to detect if they are in scenario 'a' or 'b' by applying a force at the edge or outside of the friction cone, and observing whether motion occurs.

The friction force, constraint force, and coefficient of friction are calculated for the case where the velocity is greater than zero, with the Equations 3.5, 3.6, and 3.7 presented in Chapter 3. The segmentation algorithm proceeds by evaluating $\hat{\mathbf{v}}$ and $\hat{\mathbf{f}}_c$ when the velocity is greater than zero. If the velocity becomes zero and $\hat{\mathbf{f}}_c$ has been constant up to that point in the subtask, then it is necessary to determine if the stopping indicates a sticking condition or a new contact state. A test is performed to determine if one is in the sticking condition by evaluating the hypothetical forces that would occur under the sticking conditions. The hypothetical sticking constraint force, $\mathbf{f}_{c,stick}$, is calculated by projecting the total force onto the direction of the constraint force previously identified from motion up to that point in the subtask, designated by $\hat{\mathbf{f}}_{c,prev}$. The projection is given by the following equation where the superscript 'T' designates the transpose.

$$\mathbf{f}_{c,stick}(t) = \left([\mathbf{f}(t)]^T \hat{\mathbf{f}}_{c,prev}\right) \hat{\mathbf{f}}_{c,prev} \qquad \text{for } |\mathbf{v}(t)| = 0 \qquad (4.1)$$

where $\hat{\mathbf{f}}_{c,prev}$ is the direction of constraint force up to that point in the subtask. The hypothetical friction force, $\mathbf{f}_{f,stick}$, is given by subtracting $\mathbf{f}_{c,stick}$ from $\mathbf{f}$:

$$\mathbf{f}_{f,stick}(t) = \mathbf{f}(t) - \mathbf{f}_{c,stick}(t) \qquad \text{for } |\mathbf{v}(t)| = 0 \tag{4.2}$$

The contact force is within the friction cone and corresponds to the sticking condition if:

$$\frac{|\mathbf{f}_{f,stick}(t)|}{|\mathbf{f}_{c,stick}(t)|} \leq \mu_{max} \qquad \text{for } |\mathbf{v}(t)| = 0 \tag{4.3}$$

where μ_{max} is the maximum coefficient measured during the demonstrations. When Equation 4.3 is satisfied, then the direction of $\hat{\mathbf{f}}_c$ has not changed and motion is within the same contact state.

The segmentation algorithm is performed for each demonstration and identifies regions in which either $\hat{\mathbf{f}}_c$ or $\hat{\mathbf{v}}$ are constant. The beginning time for each subtask is t_b, and the procedure is performed by evaluating $\hat{\mathbf{f}}_c$ and $\hat{\mathbf{v}}$ while incrementing the time. Figure 4.2 depicts the segmentation algorithm, where t_{end} is the time at the end of the demonstration.

```
FOR  $t = 0$ to $t_{end}$
{
 IF $\{\hat{\mathbf{v}}(t) = \text{const. for } t \in [t_b, t]\}$ OR
   $\left\{\left(\hat{\mathbf{f}}_c(t) = \text{const. for } |\mathbf{v}([t_b, t])| > 0\right) \text{ AND } \left(\frac{|\mathbf{f}_{f,stick}(t)|}{|\mathbf{f}_{c,stick}(t)|} \leq \mu_{max} \text{ for } |\mathbf{v}([t_b, t])| = 0\right)\right\}$
 THEN
     Motion is within a single contact state
     Therefore, continue incrementing 't'
 ELSE
     Define a new subtask begining at 't'
 END
}
CONTINUE THE LOOP
```

Figure 4.2: Segmentation Algorithm

The methods used to identify whether $\hat{\mathbf{f}}_c$ or $\hat{\mathbf{v}}$ are constant are the same methods used in Section 3.4.2. Variation in the direction of $\hat{\mathbf{f}}_c$ is compared to a noise threshold level, and variation in $\hat{\mathbf{v}}$ is evaluated by comparing the position trajectory to a least squares straight line. In addition, the condition,

|v|=0 in the above algorithm is based on a comparison to a threshold noise level and not absolute zero.

The necessity of considering the case when the velocity is equal to zero is depicted in the example shown in Figure 4.3. Here the demonstrator moves along a single constraint surface, momentarily contacts an additional constraint, and then continues along the original constraint surface. The demonstrator uses the momentary contact to identify a position on the surface, and changes their direction of motion after the contact. If the segmentation analysis did not evaluate $\hat{\mathbf{f}}_c$ when the velocity was zero, then both sections of motion on the surface would have been considered a single subtask (since $\hat{\mathbf{f}}_c$ is the same during both parts of the motion).

However, to successfully implement the motion in Figure 4.3, two subtasks are required using the robot controllers of the type defined in Chapter 3. Each controller can only adapt to orientation misalignment, and not change motion direction in response to contact with a new surface, as is require by the task. The momentary contact and resulting turning point in the trajectory can occur at a different point in the trajectory depending on workpiece misalignment. If both segments of motion were evaluated as a single subtask, then variation between demonstrations in the position of the turning point would be interpreted as inconsistency, while in actuality these variations correspond to adaptation to the environment. The segmentation algorithm successfully identifies the momentary contact as a change in contact state, and valid comparisons between demonstration trajectories are possible for both segments of motion.

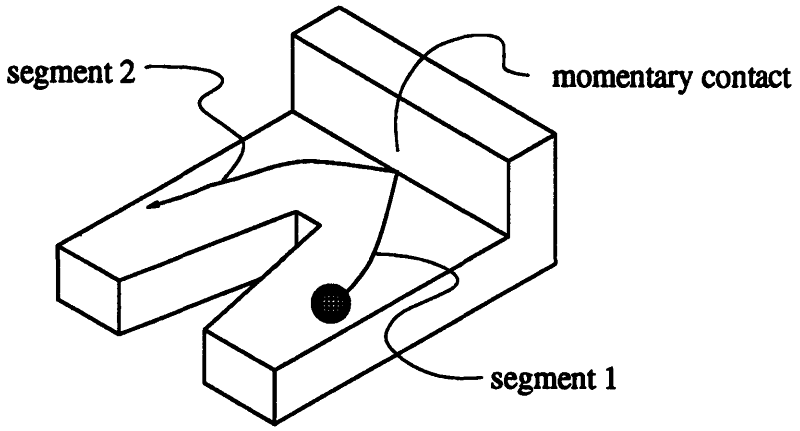


Figure 4.3: Momentary Contact Between Two Segments of Motion

The algorithm presented in this section segments the demonstration data into subtasks consisting of a single contact states, except for certain cases which are addressed in the following section. In addition random pauses by the demonstrator during the task do not effect the segmentation results. When each contact state is successfully identified, the segmentation algorithm can be directly integrated with robot controller analysis of Chapter 3, where each robot controller is specified for motion within a single contact state.

4.2.2 Undetectable Contact State Transitions

In both model based robot programming and PHD, the criterion for appropriate subtask definition is the ability of the robot controller for each subtask to successfully complete the subtask. Segmentation is coupled to adaptation to environment. Indeed, if there was no variation in the environment a complete task could be implemented as a single subtask, where the robot controller consisted of a fixed pre-programmed trajectory. Segmenting a task into subtasks simplifies the adaptation process, since the robot controller for each subtask is required to adapt only to workpiece misalignments encountered within the subtask.

In the PHD approach, it is necessary to transfer to the robot program the demonstrator's adaptation to the environment. The adaptation strategy the human uses cannot be directly measured, however all adaptation is derived from detected variations in the environment. Whenever, an aspect of workpiece misalignment is detectable, it is possible that the demonstrator uses that information to adapt. On the other hand, a segment of motion in which *no* additional information is acquired, can be implemented with a fixed trajectory defined from prior information. Such a segment of motion could be implemented as a single subtask even with the simplest 'fixed trajectory' robot controller. If the robot controller has some adaptation capability, then additional motion could be incorporate into the subtask.

In 3D contact tasks, information regarding workpiece misalignment is acquired through contact with the workpiece surfaces. Initial contact with a surface provides information regarding translation misalignment of that surface, and sliding over a surface provides orientation information as shown in Figure 3.10. However, no additional information is acquired from motion within a contact state, after the initial contact and a small amount of sliding motion sufficient to identify orientation misalignment.

The robot controllers defined in Chapter 3 can achieve motion within a single contact state, because they adapt to orientation misalignment which is the only information acquired within a contact state[1]. These same controllers can achieve any motion, as long as no additional adaptation is required beyond the adaptation to orientation misalignment detect during the initial sliding motion. Accordingly, the compliant controller identified in Chapter 3 can achieve motion consisting of two contact states, as long as no additional information is acquired in the second contact state. In the following two sections it is shown that when undetected contact state transitions occur, no additional information is acquired during the second contact state, and how a single robot controller achieves motion for both contact states.

4.2.2.1. Transition From Two to One Constraint

In 3D constrained motion there are two constraint configurations, and therefore four possible types of contact state transitions between them. Undetected contact state transitions can occur in all four types of transitions, when either $\hat{\mathbf{f}}_c$ or $\hat{\mathbf{v}}$ which is constant in one contact state does not change as the part moves to the next contact state. The most likely of these to occur is in the transition from a two constraint configuration to a single constraint, and is shown in Figure 4.4b. Here, in all the demonstrations the direction of motion remains in a straight line even as the part moves to the single constraint contact state. Accordingly, the segmentation algorithm detects a constant $\hat{\mathbf{v}}$, and does not segment the motion at the contact state transition.

The demonstrator is able to continue moving in a straight line, because they are familiar with the workpiece and anticipate the transition. During the initial segment of motion with two constraints, the direction of force is applied so that contact is maintained with both surfaces. If the change in constraint was unanticipated, then a non zero force component in the 'z' direction, as shown in Figure 4.4, would be applied at the point of transition, and would result in a deviation from the straight line motion. Of course the demonstrator could detect the change in motion and correct their trajectory with only a small change in $\hat{\mathbf{v}}$. However it is assumed (assumption 10) that the teaching gripper sensors are accurate enough to detect the geometric

[1] The two stage approach presented in Section 3.5.2.3 changes the robot controller after as small amount of motion in which the orientation information is measured. Here two controllers are used for each contact state, and the task is segmented at each point where information is acquired. Accordingly, the second stage robot controller has no adaptation requirements and is implemented with a fixed trajectory.

features that the human uses, and thus if the transition was unanticipated it would be detected in the measurement of $\hat{v}$.

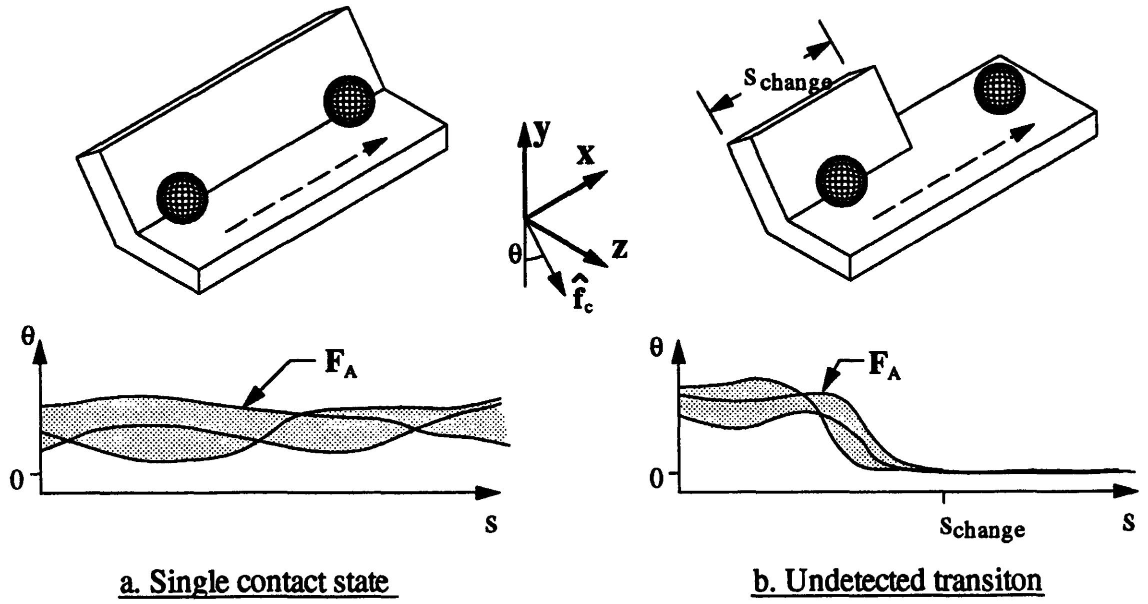


Figure 4.4: Region of Acceptable Force Directions For Two Cases of Straight Line Motion

Both cases in figures 'a' and 'b' are segmented into a single subtask since the direction of motion is constant, however the shapes of the regions F_A are different.

The demonstrator is able to anticipate the change in the constraint surface (with their eyes closed), just as the demonstrator is able to avoid obstacles through familiarity with the obstacle locations. In Figure 4.4 the demonstrated force trajectories are shown as a function of the distance traveled, where the direction of force, θ, is defined in Equation 3.21. The range of acceptable force directions, F_A, is defined by the region bounded by the demonstrated trajectories. In the configuration with a single contact state, Figure 4.4a, the region F_A is relatively wide and consistent throughout the motion. However, in the configuration where a contact state transition occurs, Figure 4.4b, the region F_A becomes narrow just prior to the change in contact states and the direction of force in all the demonstrations is applied in the 'y' direction in anticipation of the upcoming single constraint.

In order for the robot to maintain the same position accuracy as the human, the robot controller should be able to achieve straight line motion despite the change in the constraint. The procedure for specifying a robot controller presented in Chapter 3, is applied here without modification. The nominal trajectory of the robot force is selected to lie within the region F_A. For the motion in Figure 4.4a, a constant direction of robot force could be selected. However, when an undetected change in contact state occurs, as shown in Figure 4.4b, selecting a robot force trajectory within $\mathbf{F}_A$ automatically generates the desired force trajectory that 'anticipates' the contact state transition and results in straight line robot motion. Here the shape of $\mathbf{F}_A$ defines the robot force trajectory, just as the shape of $\mathbf{X}_A$ defines the position trajectory in obstacle avoidance.

Undetected contact states transitions do not occur frequently, because they required accurate human motion and also preclude certain workpiece misalignments. For example, the demonstrator can perform the motion in Figure 4.4a, only if there is no workpiece orientation misalignment about the 'x' axis. Rotation misalignment about the direction of motion is undetectable in the first contact state, yet detectable in the second contact state. If such misalignment occurred, then the force applied by the demonstrator at the transition would not be normal to the constraint surface, and deviation from straight line motion would occur. Therefore, the ability of the demonstrator to continue in a straight line indicates that there is no rotation misalignment of the workpiece about the direction of motion. In general an undetected contact state transition cannot occur if there is workpiece misalignment in directions that are undetectable in the first contact state yet detectable in the second contact state.

In addition, assumption 7 precludes using information detected in the first contact state for motion in the second contact state. Thus, workpiece rotations about the 'y' axis in Figure 4.4a will violate assumption 7, since straight line motion will require that the direction of motion in the second contact state is defined by the constraints in the first contact state. Indeed, a compliance controller specified for a two constraint configuration will deviate from straight line motion if workpiece rotation occurs about the 'y' axis[2]. The remaining direction in which a misalignment is permissible is one that is detectable in both contact states, such as workpiece rotation about the 'z' axis in Figure 4.4.

[2] Assumption 7 could be relaxed for undetected contact state transitions, as long as the two stage robot controller, presented in Section 3.5.2.3, is used. The two stage controller incorporates all the information from initial motion in the first contact state into the reference trajectory for the reminder of the subtask. Thus, in the example in Figure 4.4b, the robot could achieve straight line motion even if workpiece rotations occurred about the 'y' axis.

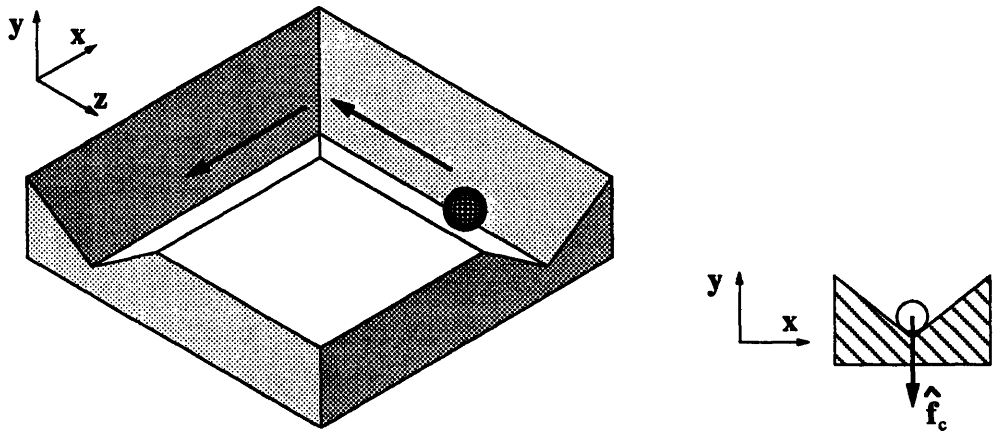
In PHD the workpiece geometry is not known, and thus the presence of a region $\mathbf{F}_A$ shaped like in one in Figure 4.4b does not *necessarily* mean that there is a change in contact state. However, as with the previous analysis in Chapter 2 and 3, components of the motion and force trajectories that are *consistently* demonstrated are transferred to the robot program, which is *sufficient* to achieve robot success.

This Section shows that an undetected contact state transition does not need to be detected, and the method used to specify a robot controller presented in Chapter 3 can be implemented as if the motion was within a single contact state. Even if undetected contact state transitions do not occur often, the analysis is significant because it illustrates the consistency of the analytical approach. In Chapter 3, the specification of the robot controllers is based on the approach that it is only necessary to adapt to workpiece misalignment that can be detected. This same approach is applied in this section to segmentation, where it is shown that it is necessary to define a new subtask only when new workpiece misalignment is detectable. The continuity in the analytical approach also extends to the use of human inconsistency. In this section the analogy between the regions $\mathbf{X}_A$ and $\mathbf{F}_A$ is extended; variation in the width of $\mathbf{F}_A$ as a function of the distance traveled, indicates task accuracy requirements as does variation in the width of $\mathbf{X}_A$.

4.2.2.2 Unlikely Transitions

There are three other types of undetectable contact state transitions, besides the one presented in Section 4.2.2.1. These other types of transitions are unlikely to occur in actual demonstrations, however are included here for completeness. In all undetected transitions, no information is acquired during the transition, and both contact states can be implemented as a single subtask.

Figure 4.5 shows the case of an undetected transition between a pair of contact states that both have two directions of constraint. As shown in the cross section, the direction of force is always aligned with the 'y' axis. Accordingly, the segmentation algorithm would detect a constant $\hat{\mathbf{f}}_c$ and both contact states would be treated as a single subtask. However, it is unlikely that the demonstrator would be able to keep the direction of force constant while performing both segments of motion (with their eyes closed). The human would have to anticipate exactly where the first contact state ends, and make the correct change in direction of motion without using force feedback. It is more likely that the demonstrator would detect the end of the first contact state by a change in the direction of force when the part contacts the new surfaces. The demonstrator would only be able to anticipate the change in contact state if the distance traveled in the first contact state is the same in all demonstrations (and the human had perfect position accuracy). Under these circumstance, the region of acceptable motion would be defined by trajectories that lie on top of each other, and would uniquely define a robot position trajectory that followed the desired motion for both contact states.



a. Isometric view a. Cross section

Figure 4.5: Undetected Transition Between Contact States With Two Directions of Constraint

Another undetected contact state transition is shown in Figure 4.6. Here the transition is between two contact states with a single direction of constraint, yet the direction of motion remains constant. As in the previous example, it is unlikely that the demonstrator could accomplish this motion in a straight line, since it would require very high position accuracy and a change in the direction of force at exactly the right point. Nevertheless, if the demonstrator does perform both segments with a straight line, the regions of acceptable force will define a robot trajectory that can also perform the straight line motion.

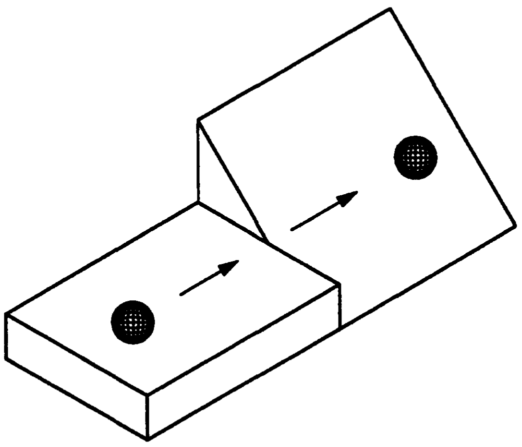


Figure 4.6: Undetected Transition Between Contact States With a Single Direction of Constraint

The final type of undetected contact state transition, occurs with motion from a single constraint to two constraints, which is the reverse of what is shown in Figure 4.4b. Here perfect position accuracy is required to align the direction of motion in the single constraint with that in the two constraint configuration. The region of acceptable force will be the same as shown in Figure 4.4b, but reversed. As in the other cases of undetectable transitions, the demonstrations define an appropriate robot force and position trajectory. All cases of unlikely undetectable transitions, require the human to predict changes in the contact state with perfect position accuracy, and therefore allow almost no workpiece misalignment.

4.3. Ensuring Corresponding Segmentation Between Demonstrations

The PHD approach in Chapters 2 and 3 relies on comparisons between multiple demonstrations. Each subtask is compared to the corresponding subtasks from other demonstrations. If the segmentation algorithm is not consistent, then it is not possible to perform meaningful comparisons. Accordingly, it is necessary to ensure corresponding segmentation between demonstrations.

The category of tasks addressed in this thesis are ones which are demonstrated with the same sequence of contact states in each demonstration (assumption 6). Accordingly, when all the contact state transitions are detected, ensuring corresponding segmentation is straightforward. Each demonstration has the same number of subtasks with a one to one correspondence between contact states.

Segmentation is also straightforward when undetected contact state transitions occur, as long as the transition is consistently undetectable in all demonstrations. Here each demonstration has the same number of subtasks, where the subtasks with undetected transitions have the same two contact states in all demonstrations.

However, it is possible that a contact state transition is detectable in one demonstration but not in another, as shown in Figure 4.7. After applying the segmentation algorithm presented in Section 4.2.1 (referred to as preliminary segmentation), the motion shown in Figure 4.7a is identified as a single subtask with a constant $\hat{\mathbf{v}}$, and the motion in Figure 4.7b is segmented into two subtasks. Here not all demonstrations have the same number of subtasks. A method is presented in this section to identify this circumstance, and a secondary segmentation algorithm is presented to ensure corresponding subtasks between demonstrations.

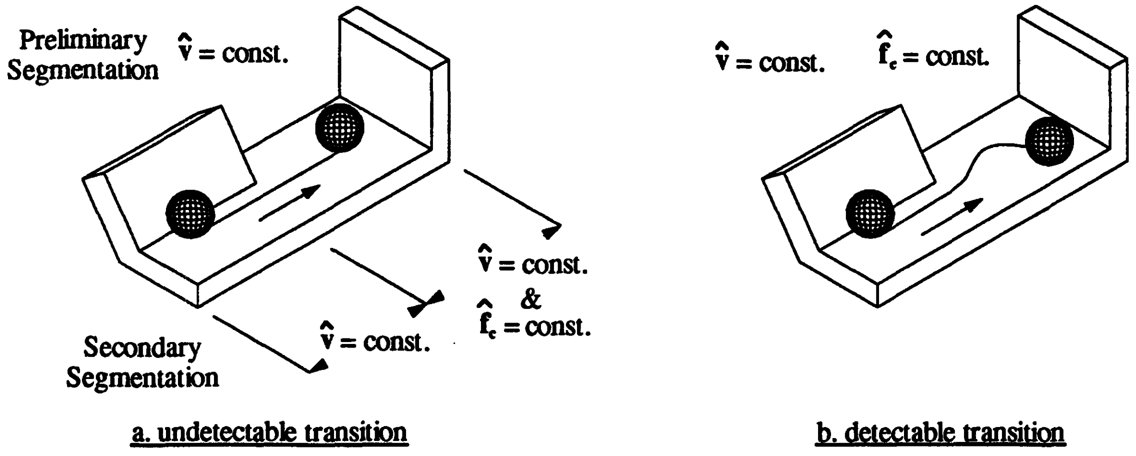

Preliminary Segmentation
$\hat{\mathbf{v}}$ = const.
$\hat{\mathbf{v}}$ = const. & $\hat{\mathbf{f}}_c$ = const.
Secondary Segmentation
$\hat{\mathbf{v}}$ = const.
a. undetectable transition
$\hat{\mathbf{v}}$ = const.
$\hat{\mathbf{f}}_c$ = const.
b. detectable transition

Figure 4.7: Preliminary Segmentation May Be Inconsistent

When a contact state transition is detected in one demonstration, it is possible to use that information to detect the corresponding transition that is undetectable in another demonstration. The secondary segmentation considers only the undetected transition which is possible under realistic circumstance as presented in Section 4.2.2.1, and is the straight line motion between two constraints and a single constraint. In the second contact state of the undetected transition, both $\hat{\mathbf{f}}_c$ and $\hat{\mathbf{v}}$ are constant, as shown in Figure 4.7a. Accordingly, once it is established that an undetected transition has occurred, the transition point is identified by searching backwards from the end of the overall section where $\hat{\mathbf{v}}$ is constant for the segment of constant $\hat{\mathbf{f}}_c$.

To implement the secondary segmentation it is necessary to identify where undetected transitions occur in only some of the demonstrations. The first step is to identify detectable transitions that could be demonstrated in an undetectable fashion, such as shown in Figure 4.7b. In the detectable transition, the position trajectories remains on a single plane of motion for both contact states (otherwise it would not be possible to perform this transition with straight line motion). The contact state that follows is referred to as the third contact state, and will have either one or two constraint directions. If the third contact state is a single constraint, then the motion will be in a new plane of motion, since two planes that intersect cannot have the same orientation.

If a demonstration has an undetected transition, as in Figure 4.7a, then the preliminary segmentation switches directly from motion with a constant $\hat{\mathbf{v}}$ to the third contact state of the detectable transition, since all demonstrations are performed with the same sequence of contact states. Accordingly, demonstrations with undetected transitions are identified by finding where the preliminary segmentation skips a contact state. The skip will occur between a segment of motion with a constant $\hat{\mathbf{v}}$, to a segment corresponding to the third contact state. If the third contact state has two constraints, then the switch will be to motion with a constant (but different) $\hat{\mathbf{v}}$. If the third contact state has a single constraint, then the switch will be to motion with a constant $\hat{\mathbf{f}}_c$ and the plane of the motion will change.

The secondary segmentation algorithm is implemented after the preliminary segmentation is performed on all the demonstrations, and is shown by the following steps:

1. Find all detectable contact state transitions corresponding to Figure 4.7b, by identifying transitions that satisfy both of the following criteria.
    - Identify all pairs of contact states where a segment of constant $\hat{\mathbf{v}}$ is followed by a segment of constant $\hat{\mathbf{f}}_c$.
    - If motion in both contact states lie in the same plane of motion, then that pair of contact states could be performed with straight line motion by an undetectable contact state. A least squares approximation is used to determine if motion lies on a single plane, using the method shown in Appendix II.

2. Find demonstrations where the contact state transition is not detected by the preliminary segmentation and skips to the third contact state. A segment of motion that potentially contains two contact states, is any segment with a constant $\hat{\mathbf{v}}$ that appears in the subtask sequence at the position corresponding to the first contact state in the detectable demonstrations.
    - If this segment with a constant $\hat{\mathbf{v}}$ is followed by another segment with a constant $\hat{\mathbf{v}}$ then an undetected transition occurs, since the second contact state with a constant $\hat{\mathbf{f}}_c$ is skipped.
    - If this segment with a constant $\hat{\mathbf{v}}$ is followed by a segment with a constant $\hat{\mathbf{f}}_c$, yet the motion is not in the same plane, then an undetected transition occurs. The change in the plane of motion indicates that the segmentation could not correspond to the detected segmentation in Figure 4.7b.
3. Re-segment each segment with an undetected transition into two contact states. Search from the end of the segment for the longest region in which $\hat{\mathbf{f}}_c$ is constant, as shown in Figure 4.7a.

In most cases all the contact state transitions are detectable and step 3 is not necessary.

4.4. Experimental Results

The segmentation algorithm is applied to the example task described in Chapter 3. During the constrained motion, each demonstration was performed with the same sequence of contact states, which is a prerequisite for the segmentation analysis. However, in the transition between unconstrained motion to constrained motion, in some of the demonstrations a bounce was detected. The part gained contact, then loss contact momentarily, and then regained contact. The bounce was not perceived from visual observation of the demonstration, but was present in the data.

Since the bounce occurred only in some of the demonstrations, the sequence of contact state transition was not totally consistent throughout the demonstrations. However, since the bounce is not an important component of the manipulation strategy, it was eliminated from the analysis. Any number of transitions between unconstrained motion and constrained motion, that occurred within a short period of time, were treated as a single contact state transition. The threshold period between transitions for eliminating the bounce time, was 0.3 seconds.

To visualize the scope of the segmentation analysis, the position trajectories from all ten demonstrations are shown in Figure 4.8. The variation between the demonstrations is due to both human inconsistency and adaptation to workpiece misalignment. One can observe in Figure 4.8 qualitative changes in motion in each trajectory that correspond to different contact states in the task. To perform a meaningful comparison between demonstrations, segmentation is necessary.

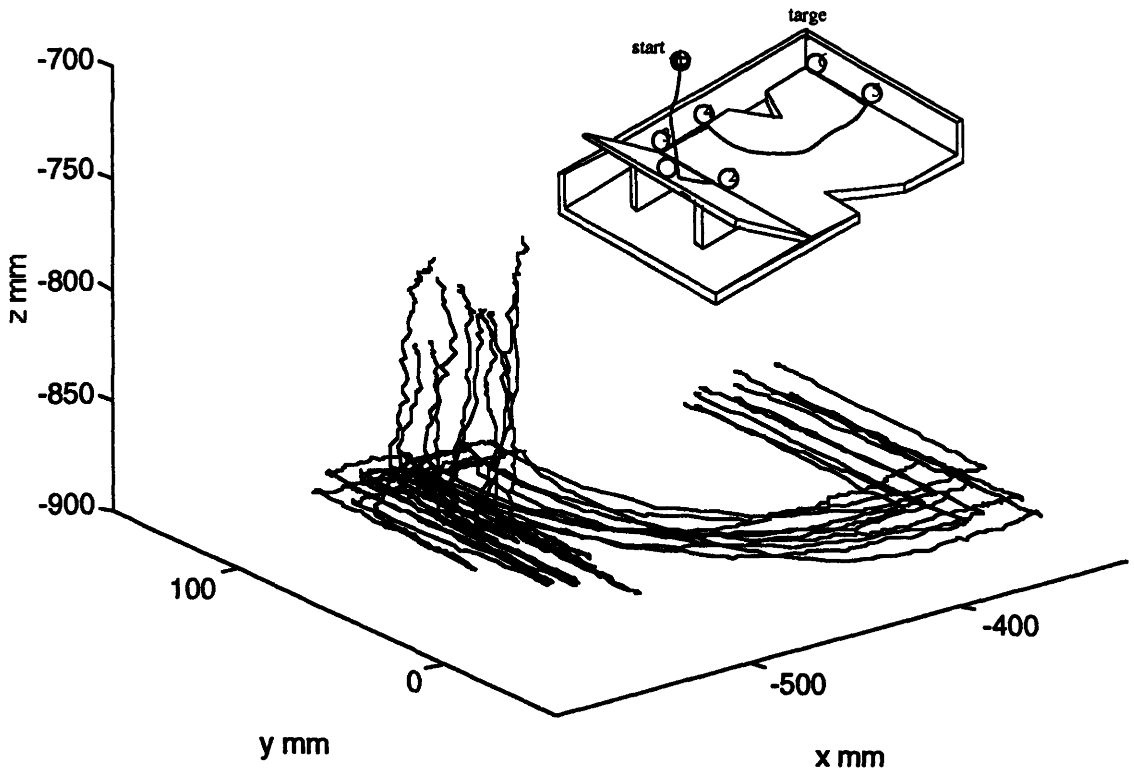


Figure 4.8: Position Trajectory From All Demonstrations

The primary segmentation algorithm presented in Section 4.2.1 was applied to each demonstration. In the analysis three parameters are used that correspond to the noise level in the sensor measurements. The threshold level beneath which $|v(t)|$ is considered zero is 5mm/s. Variation in the direction $\hat{f}_c$ is calculated using Equation 3.8, and is indicated by a mean value of β. The threshold level for β, beneath which $\hat{f}_c$ is considered constant is 5°. Variation in $\hat{v}$ is calculated by performing a least squares fit of a straight line on the position trajectory, as presented in Appendix II. The standard deviation of the noise in the position trajectory is set equal to 1.0mm.

The segmentation of one demonstration trajectory is shown in Figure 4.9. At each point where the segmentation algorithm identified a new subtask, the position trajectory switches between a solid line and a dotted line.

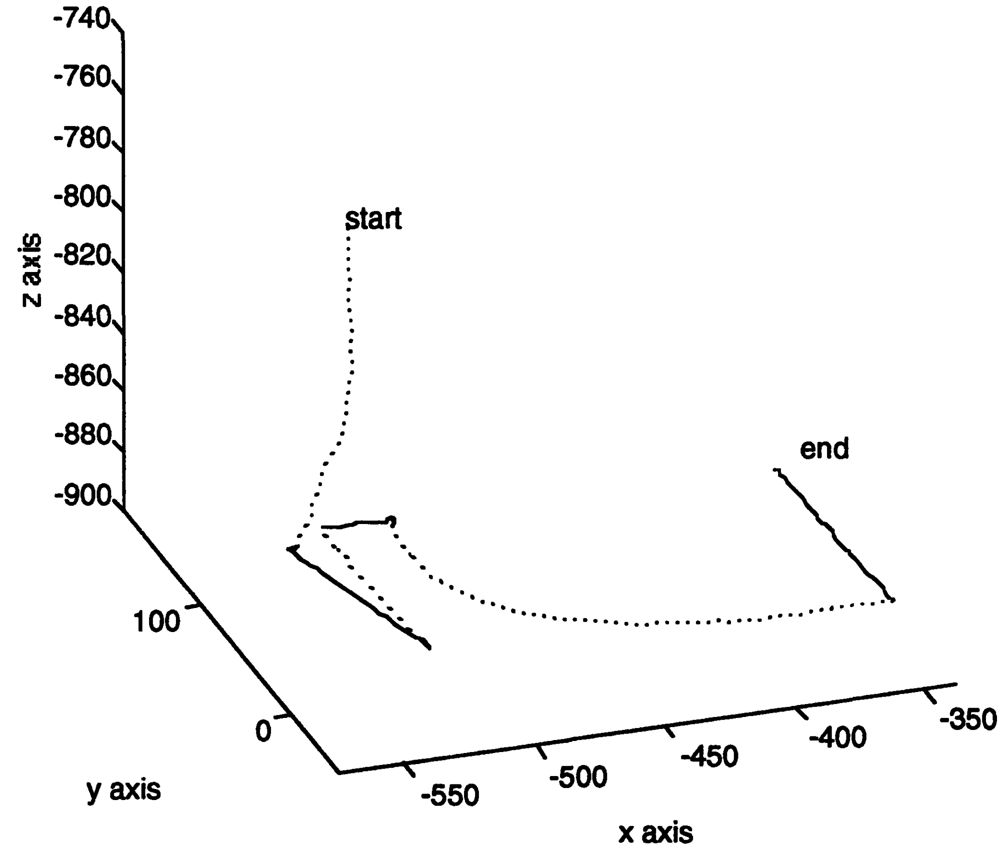


Figure 4.9: Segmentation of One Demonstration

All ten demonstrations were successfully segmented into six subtasks. The results of the segmentation analysis are shown in Figure 4.10. Column two indicates which of the directions $\hat{f}_c$ or $\hat{v}$ was found to be constant. Column three shows the number of constraints that can be identified from the data. In the third subtask, both the force and motion direction were constant in all demonstration, and therefore the number of constraints cannot be determined from the data. In this contact state the demonstrator had no need to avoid obstacles and covered only a short distance of motion, which is why the trajectories were in straight lines even though there is a single constraint. Thus, for the third subtask, the robot controller is defined using the analysis presented in Section 3.5.2.3, which addresses the case of constraint ambiguity. Column 4 shows the nominal detectable surface orientation, which is identified by averaging either directions of $\hat{\mathbf{f}}_c$ or $\hat{\mathbf{v}}$, as indicated by Equations 3.9 and 3.10.

Subtask	Constant	Constraints	Nominal direction
1	$\hat{\mathbf{f}}_c$	0	$\hat{\mathbf{f}}_c^n = [0]$
2	$\hat{\mathbf{f}}_c$ and $\hat{\mathbf{v}}$	1 or 2	$\hat{\mathbf{f}}_c^n = \begin{bmatrix}0.00 & -.60 & -.80\end{bmatrix}$ $\hat{\mathbf{v}}^n = \begin{bmatrix}0.00 & +.80 & -.60\end{bmatrix}$
3	$\hat{\mathbf{v}}$	2	$\hat{\mathbf{v}}^n = \begin{bmatrix}1.00 & -.03 & .02\end{bmatrix}$
4	$\hat{\mathbf{v}}$	2	$\hat{\mathbf{v}}^n = \begin{bmatrix}.03 & .99 & .08\end{bmatrix}$
5	$\hat{\mathbf{f}}_c$	1	$\hat{\mathbf{f}}_c^n = \begin{bmatrix}.01 & -.03 & -1.00\end{bmatrix}$
6	$\hat{\mathbf{v}}$	2	$\hat{\mathbf{v}}^n = \begin{bmatrix}1.00 - .03 & -.01\end{bmatrix}$

Figure 4.10: Segmentation Results

To ensure corresponding segmentation, the secondary segmentation algorithm was applied. In step 1, the only two subtasks in which a constant $\hat{\mathbf{v}}$ is followed by a constant $\hat{\mathbf{f}}_c$ are subtasks 4 and 5. Step 2 indicated that in each demonstration the trajectories of both subtasks lie on the same plane. Accordingly, there was no need for additional segmentation to ensure corresponding segmentation.

4.5. Summary

In this chapter a method is presented to segment the task into a sequence of subtasks. The segmentation is consistent with the specification of the robot controller, presented in Chapter 3, and thus each subtask can be implemented by the corresponding robot controller. The segmentation approach is straightforward in that a new subtask is defined wherever a contact state transition is identified.

In a contact state with a single constraint, the direction of constraint force, $\hat{\mathbf{f}}_c$, is constant, and the direction of motion $\hat{\mathbf{v}}$ is constant when there are two constraints. Accordingly, the preliminary segmentation algorithm searches for sections of motion with either a constant $\hat{\mathbf{f}}_c$ or $\hat{\mathbf{v}}$. To implement this algorithm it is necessary to identify changes in $\hat{\mathbf{f}}_c$ at all points in the trajectory, including points where the velocity is zero. The characteristics of the friction component of the measured force, requires that a separate test be performed to identify changes in $\hat{\mathbf{f}}_c$ when the velocity is zero.

The preliminary segmentation algorithm does not identify all contact state transitions. However, it is shown that these transitions do not need to be detected, as long as they are undetected in all demonstrations. Under these circumstances, the robot controller defined for a single contact state will achieve the motion for both contact states. To account for the case where a transitions is detected in some demonstrations but not others, a secondary segmentation algorithm is applied. This algorithm uses a detected transition in one demonstrations to identify a previously undetectable transitions in other demonstrations. Accordingly, all demonstration are segmented into the same number of subtasks, which have corresponding contact states.

The overall segmentation process requires a number of algorithms and consideration of special cases. However, there is a consistent underlying approach throughout these procedures, which is based on information acquired from the environment. The segmentation algorithm defines new subtasks at points where new information is acquired. The compliant controllers defined in Chapter 3, can adapt to certain directions of orientation misalignment, but any other information results in a new subtask. It is shown that whenever a contact state transition is undetectable, no new information is acquired, which is why these transitions do not need to be detected.

Other methods of segmentation have been presented which rely on a model of how the human performs the task, such as identifying changes in the demonstrator's velocity. However, these methods breakdown when human motion is not consistent with the model. Thus, the information based approach is more reliable, and does not require the human motion to conform to a specific model.

CHAPTER 5

Subtask Termination Conditions

5.1. Introduction

Subtask termination conditions are sensor measurements that indicate to the robot that motion within a subtask has been completed, and that the robot should switch to the next subtask. Accordingly, termination conditions provide an important component of the ability of the robot to adapt to the environment.

Each demonstration is performed with the same sequence of contact states. In Chapter 3, a robot compliance controller is specified that can implement motion and reach the target position within each contact state. Accordingly, robot success can be guaranteed as long as the robot can switch between one contact state to the next. The subtask termination conditions provide the robot with this ability.

In model based robot programming, it is necessary to predict sensor signals that correspond to the termination conditions. However, with the PHD approach the sensor signals are directly available. Accordingly, sensor measurements that indicate completion of a subtask can be extracted from the demonstration data. In Section 5.2 a method is presented for identifying the termination conditions, by correlating the completion of subtasks with sensor measurements.

The analysis used to segment a task into subtasks in Chapter 4, is different from the analysis used to identify termination conditions. In specific, when segmenting a task, all of the data from a demonstration can be used. However, the robot must be able to identify a termination condition during real time implementation of the task, and therefore can only use sensory information that has been measured up to the point of subtask completion. A valid termination condition can not be based on a sensory signal that occurs after the demonstrator detects completion of the subtask and switches to the beginning of the next subtask. A method is presented in Section 5.3, to ensure that such causality inconsistencies are avoided.

5.1.1. Background

It has been recognized that identifying changes in contact states during an assembly process is useful for adapting to part misalignment. A method of guarded moves is presented by Will and Grossman [1975], in which velocity and force sensors are used to determine when contact has been established with a surface.

Asada and Hirai [1989], and Desai and Volz, [1989] use a model of the task geometry to detect a change in the contact state of polyhedral parts, from quasi-static force and position measurements. McCarragher and Asada [1993] use sensor measurements that incorporate part dynamics to identify contact state transitions. These approaches allow for the task be implemented in a variety of contact state sequences, which is more difficult than the case of a constant sequence of contact states addressed by this thesis. However these methods are model based, and therefore cannot be applied to PHD.

Hannaford and Lee [1989] identify subtask transitions in human demonstration data, by using a probabilistic model of these transitions in the form of a hidden Markov model. The probabilistic approach increases robustness in the presence of noise, especially when the demonstrator skips a subtask. However, a model of the task and expected termination conditions are used, which is not available in the PHD method.

In model based approaches it is necessary to synthesize what the sensor measurements will be for each contact state. With PHD, synthesis is not required since the sensor signals are measured directly. However, it is necessary to correlate the appropriate signals with the completion of the subtask.

5.2. Identifying Termination Conditions

Completion of motion within a subtask occurs when the part reaches the target of that subtask. Sensor measurements that indicate completion of a subtask occur in all the demonstrations at the end of the subtask, and thus correlation between sensor measurements and subtask completion can be used to identify the subtask termination conditions.

The demonstrator can detect completion of a subtask in two fashions. The first type of termination condition relies on acquiring new information from the environment; for example by contacting a new workpiece surface. The second type is through familiarity with the task, such as knowing the appropriate distance to travel within the subtask. For each subtask, a search is performed to identify a termination condition of the first type. If no new information is acquired at subtask completion, then by default the termination condition is of the second type.

The contact task used in the experiments is shown in Figure 5.1. The sequence of contact state transitions is numbered from one to six. Contact states from one to three are of the first type, where the completion of the subtask is detected by contacting a new surface. However, the fourth termination condition is of the second type. Through familiarity with the workpiece, the demonstrator knows that to continue in the fourth subtask too long will result in the part falling into the triangular hole in the surface. Accordingly, the subtask is completed when the motion has progressed a specified distance within the contact state.

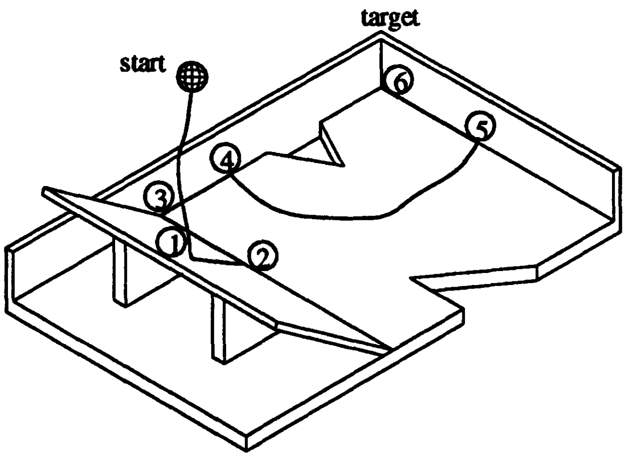


Figure 5.1: Workpiece Used in the Experiments

The first type of termination condition corresponds to new information from the environment, and is due to a change in contact state, since while the part remains in the same contact state no additional information is acquired. Sensor measurements that can indicate a change in contact state are designated as potential termination conditions. As the robot performs the task, the normalized velocity, $\hat{v}$, and the normalized force, $\hat{f}$, are continually monitored. Prior to normalizing the measured force the gravitational load of the robot gripper is subtracted, but $\hat{f}$ still includes frictional forces. Potential termination conditions are components of $\hat{v}$ and $\hat{f}$ that are aligned with the coordinate system of the current subtask.

In motion constrained in a single direction, $\hat{f}_c$ is normal to the surface and $\hat{v}$ lies on the plane tangential to the surface. Accordingly, a change in contact state could be indicated by a component of $\hat{f}$ that is in the plane of motion and exceeds the threshold corresponding to the friction component. Another potential termination condition is a component of $\hat{v}$ that is normal to the plane of motion.

In motion constrained in two directions, $\hat{v}$ is constant and $\hat{f}_c$ lies in a plane normal to the direction of motion. Accordingly, potential termination conditions includes a component of $\hat{v}$ normal to the direction of straight line motion, and a component of $\hat{f}$ in the direction of motion that exceeds a friction threshold.

For a potential termination condition to be a valid termination condition its value should reach a threshold level at the completion of the subtask, but not beforehand. The demonstrator cannot respond to a termination condition instantaneously. Rather there is a period of time, defined as the termination region, between the sensor measurement that indicates subtask completion and the response of the human. The method used to calculate the termination region is presented in Section 5.3.

Each potential termination condition is designated as a function of time, C(t). Figure 5.2 shows the trajectory of a potential termination condition for the third subtask of the task shown in Figure 5.1. Here, C(t) is the component of $\hat{f}$ in the direction of motion. The maximum value of C within the termination region is given by, $C_{tr,max}$, the maximum value within the remainder of the subtask is given by, $C_{st,max}$, and the range of C is given by C_{range}. The function C(t) can only indicate subtask completion if $C_{tr,max}$ is greater than $C_{st,max}$ in all demonstrations. Accordingly, the following delectability criterion is defined, which is greater than zero only if under worst case conditions, C(t) indicates subtask completion.

$$\text{detectability} = \frac{\min[C_{tr,max}] - \max[C_{st,max}]}{\max[C_{range}]} \quad (5.1)$$

where the minimum and maximum operations are performed between all demonstrations.

Each potential termination condition with a delectability value greater than zero is valid. If there exists more than one valid termination condition, the robot uses the one with the highest delectability. The termination condition the robot uses to detect subtask completion is when the value of C(t) exceeds $\min[C_{tr,max}]$.

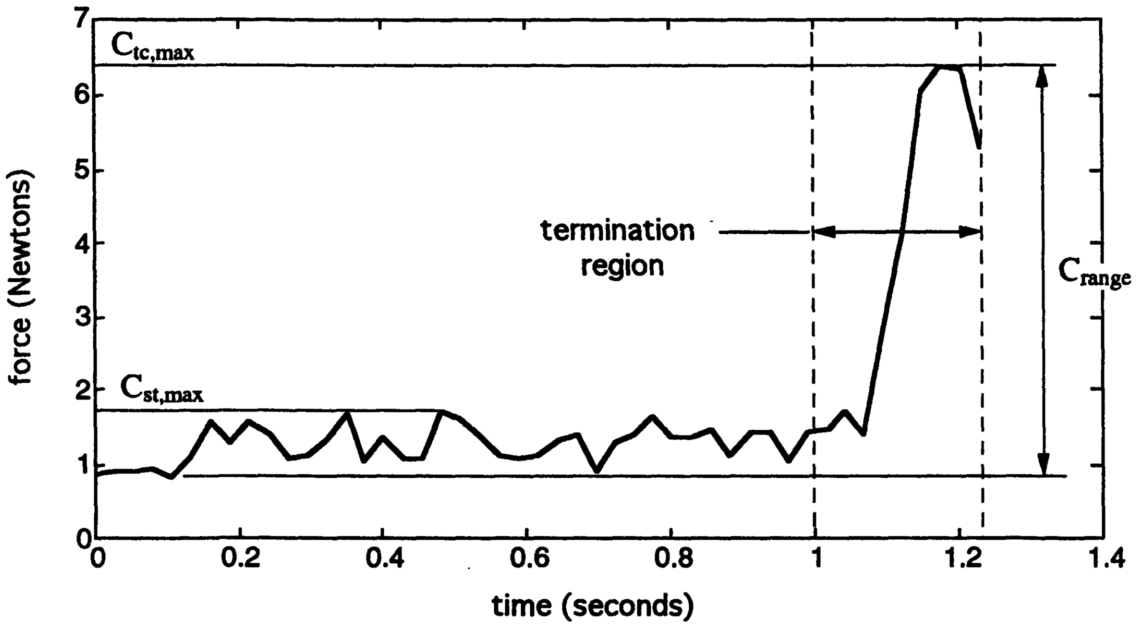


Figure 5.2: Termination Condition

If none of the potential termination conditions are valid, then the termination condition is of the second type. In this case, the demonstrator decides to complete the subtask without detecting a change in contact state. Instead the termination condition corresponds to when the part enters a target region. The shapes of the demonstrated trajectories are compared to each other relative to the detected workpiece surfaces, using the transformation shown in Equation 3.15. (If the motion is constrained in two directions, then the straight line trajectories are simply laid on top of each other). Variation in the position as subtask completion identifies the target region, and the average ending position identifies the target position. The robot detects subtask completion when the part enters the target region, which is how a task of unconstrained motion is completed.

5.3. Ensuring Proper Causality

In model based robot programming, reaction from the environment can be clearly distinguished from a programmed robot action. However, when interpreting human demonstration data for PHD, the potential exists for confusion between cause and effect. For example, Figure 5.3 shows the third contact state transition of the task portrayed in Figure 5.1. Here, the proper termination condition corresponds to an increase of the force in the 'x' direction as the part contacts a new surface. After the demonstrator detects the transition, they begin to move along the next contact state, and thus generate both motion and force (to overcome friction) in the 'y' direction. The force in the 'x' direction is the proper termination condition, which *causes* the demonstrator to switch to the next subtask. On the other hand, the force and motion in the 'y' direction correspond to the beginning of motion in the following subtask, and is the *effect* of the subtask transition.

Sensor measurements that correspond to both cause and effect occur within a short period of time at the completion of a subtask. Accordingly, the algorithm presented in Section 5.2 could mistakenly identify as a termination condition, a sensor measurement that corresponds to the beginning of motion in the next subtask. If the robot program was based on an inappropriate termination condition, then the robot would not detect subtask completion. For example, if the robot waited for motion to occur in the 'y' direction to detect the completion of the subtask in Figure 5.3, then that sensor measurement would never occur.

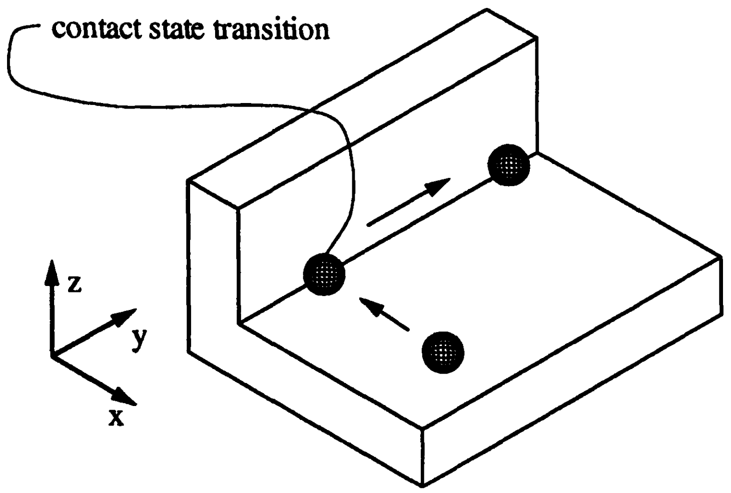


Figure 5.3: Fifth Subtask Transition

The method used to ensure proper causality, is to identify the point in time when the demonstrator changes their action and begins the next subtask. The proper termination condition will always occur prior to the change in action. The algorithm to segment the task into subtasks, presented in Chapter 4, identifies the part position where a contact state transition occurs, but not an accurate time of transition.

To identify a change in human action, a model of human manipulation is used. Up till this point in the analysis, attempts have been made to avoid relying on a model of the internal control method that the human uses, which may not be valid throughout the task. Here a human model is used only for the short period of time during which a termination condition is detected.

The model used for human motion is compliance control. Hogan [1984] presented biomechanic experiments that indicate that human motion, for relatively slow motion, can be interpreted in terms of compliance control. The model is similar to robot compliance control, but provides the relationship between human force **f**(t) and position **x**(t), which is given by:

$$\mathbf{f}(t) = \mathbf{K}\left(\mathbf{x}_{ref}(t) - \mathbf{x}(t)\right) \qquad (5.2)$$

This model is applied for a small distance of motion at the end of a subtask. For this small distance it is assumed that the reference trajectory follows a straight line at a constant speed. The use of a straight line reference trajectory corresponds to experimental results by Russell [1990] that indicate that individual segments of constrained human motion can be interpreted as compliant motion with a straight line reference trajectory. In the section of motion prior to a change in contact state, the demonstrator proceeds anticipating a gain or loss of contact at any point, but not knowing exactly where that will occur. Accordingly, it is assumed that the human reference trajectory proceeds at a constant speed, which is given by:

$$\mathbf{x}_{ref}(t) = \hat{\mathbf{v}}_{ref}\, t + \mathbf{x}_{ref,0} \qquad (5.3)$$

The objective of using the human model is to identify the point at which the demonstrator detects subtask completion and changes their action. Constant human action is identified during a period where the human motion and force can be accurately modeled in terms of compliance control with a constant **K** and $\hat{\mathbf{v}}_{ref}$.

A period of time defined as the termination region, begins just prior to the completion of a subtask, and ends at the time at which the human responds to the completion. The segmentation algorithm in Chapter 4 identifies a position, $\mathbf{x}(s_{st})$, at which a subtask is completed. The beginning of the termination region is selected as the time a which the part is a small distance, δd_{tr}, prior to $\mathbf{x}(s_{st})$ in the demonstrated trajectory. The end of the termination region is calculated by identifying the time during which the model of human motion, with a constant **K** and $\hat{\mathbf{v}}_{ref}$, remains valid. A least squares method is used to fit the model to the data in a fashion similar to identifying a region of straight line motion. The details of the analysis are shown in Appendix V.

An example of a termination region is shown in Figure 5.4. The forces in the 'x' and 'y' directions, f_x and f_y, are plotted as a function of time. The termination region begins prior to contact with the new surface. Contact with the surface can be seen in the plot at the time where the force f_x begining to increase. A reference trajectory is calculated using the values of **K** and $\hat{\mathbf{v}}_{ref}$ identified from the least squares fit. The termination region identifies where the valid termination condition, f_x, occurs, but stops prior to the increase in the force f_y or motion in the 'y' direction. Thus, use of the termination region prevents causality errors in identifying the termination region.

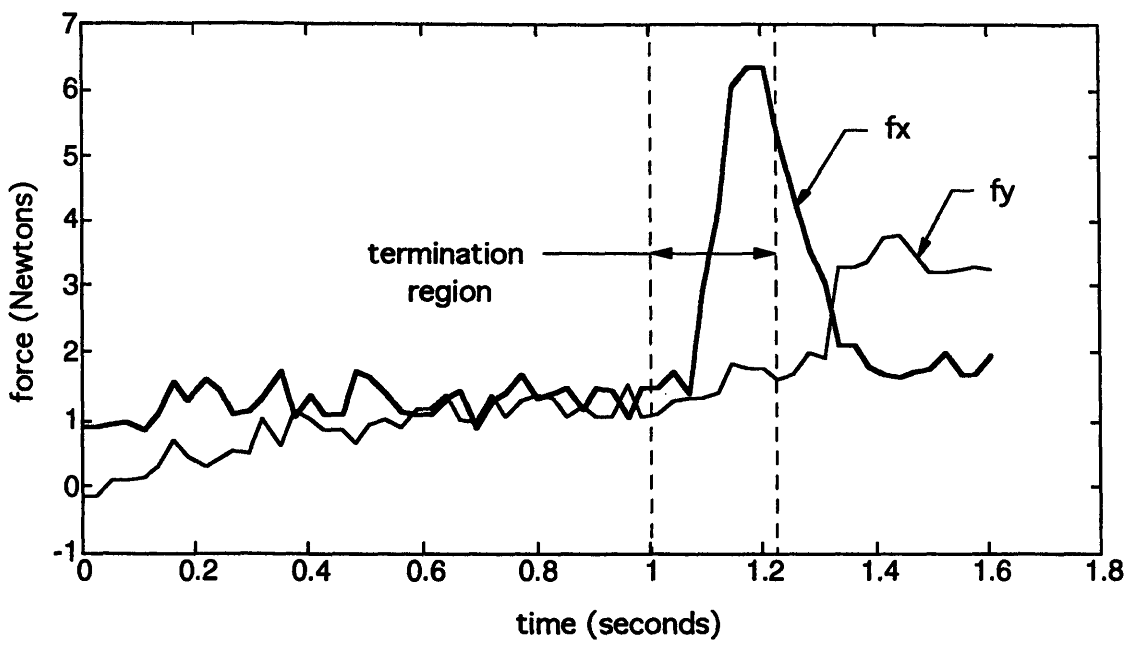


Figure 5.4: Termination Region

It should be noted that when the subtask termination condition is of the second type, the demonstrator does not rely on a change in contact state to indicate subtask completion. Here too it is necessary to identify an appropriate termination region; otherwise initial motion from the following contact state could be misinterpreted as a valid termination condition. In the case of a type two termination condition, the model of human motion is not completely accurate. The speed of the demonstrator's reference trajectory will drop rather than remain constant, as the part approaches the desired position. Nevertheless, the model provides appropriate results for identifying a valid termination condition. The inaccuracy in the model, results in a shorter period in which the model matches the human motion. The shorter termination region prevents sensory measurements from the following subtask to be considered as potential termination conditions. Moreover, a shorter termination region does not effect the accuracy of the valid termination conditions, which in this case is the average position at subtask completion.

The method presented in this Section identifies when the human changes their action. This approach is based on a model of human motion over a short distance of motion. The analysis, enables the algorithm for identifying termination conditions, presented in Section 5.2, to be implemented without causality errors. As in previous analysis in this thesis, an underlying theme is that human adaptation to the environment occurs in response on detectable variations in the environment. The causality criterion ensures that the detected variations must occur prior to motion that is interpreted as adaptation.

5.4. Summary

Identifying subtask termination conditions enables the robot to switch between subtasks. Sensory information that indicates subtask completion occurs at the end of each subtask, and such sensory signals are identified from the delectability criterion defined in Section 5.2. However, it is also necessary to ensure that a termination condition is not selected as a sensor signal that corresponds to subtask completion only because it is a result of motion in the following subtask. Proper causality is ensured, in Section 5.3, by identifying the time at which the human changes their action. Accordingly, the termination condition is selected to correspond to a sensory signal that occurs prior to the human switching to the next subtask.

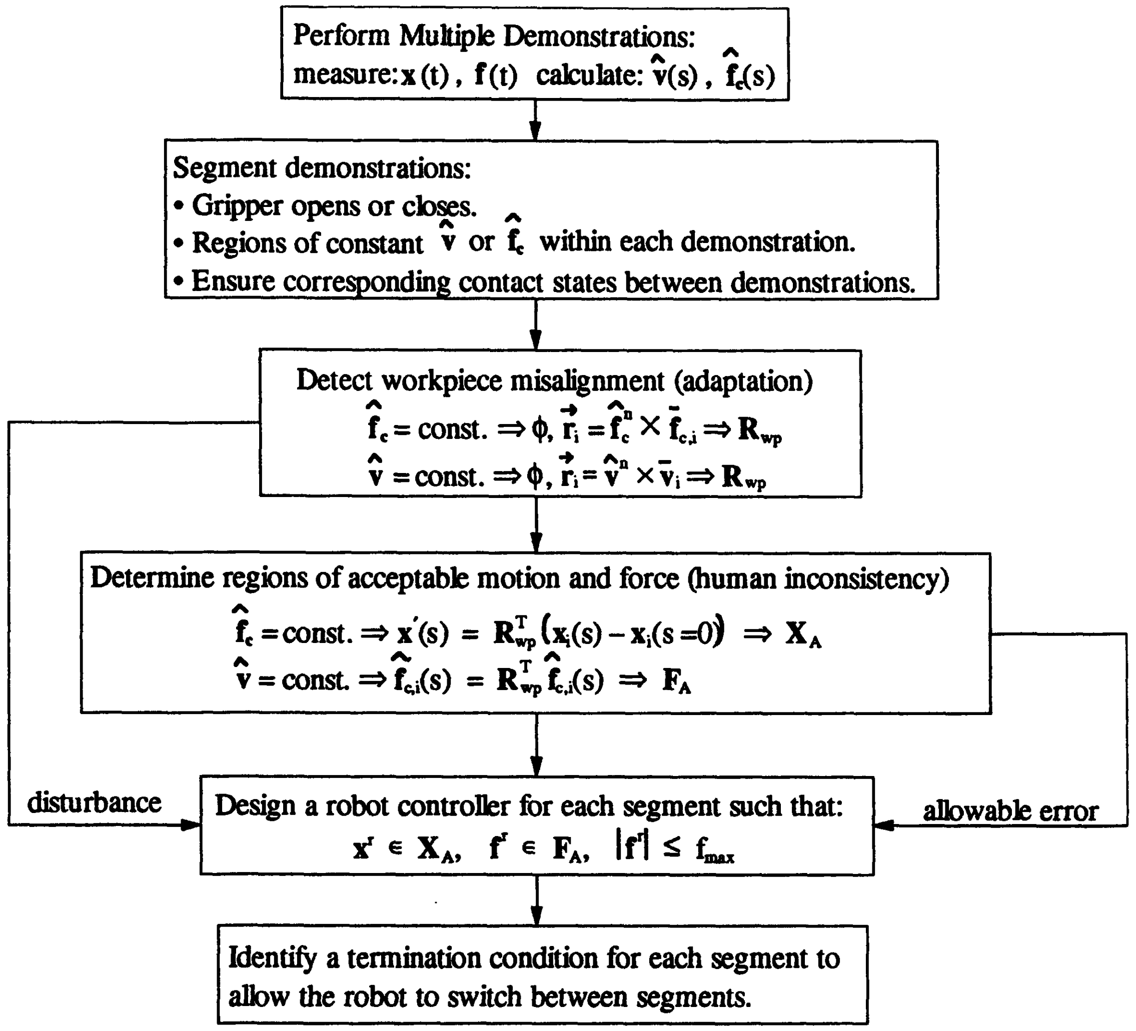
CHAPTER 6

Conclusion

A method has been presented to implement PHD for pick and place, and contact tasks. The robot program generated from the demonstration data is not a simple duplication of human motion. Rather, the robot has the ability to adapt to the environment. In addition, unnecessary human motion is removed from the robot trajectory.

A complete robot program is generated form the demonstration data, including trajectory, compliance, and subtask termination conditions. Throughout the analysis, no used is made of a geometric model task geometry, which further simplifies programming and prevents difficulties that could be caused by model inaccuracies. To ensure that all the information necessary to adapt to the environment is contained in the demonstration data, the demonstrator is restricted to using only the sensory information measured by the teaching gripper, i.e. force and position, to perform the task. By using this information a robot program is generated that can successfully perform the task as long as variations in the environment are not larger than those encountered during the demonstrations.

A summary of the different components of the analysis used to specify a robot controller for a contact task is presented in the flow chart in Figure 6.1. The chart shows how both human inconsistency and adaptation are integrated together to specify the robot performance requirements. Workpiece misalignment results a disturbance in the robot motion, and human inconsistency provides for an allowable margin of error. Thus, the method presented in Chapter 3 specifies robot compliance to adapt to workpiece misalignment while staying within the regions of acceptable motion and force.



Perform Multiple Demonstrations:
measure: $\mathbf{x}(t)$, $\mathbf{f}(t)$ calculate: $\hat{\mathbf{v}}(s)$, $\hat{\mathbf{f}}_c(s)$
Segment demonstrations:
• Gripper opens or closes.
• Regions of constant $\hat{\mathbf{v}}$ or $\hat{\mathbf{f}}_c$ within each demonstration.
• Ensure corresponding contact states between demonstrations.
Detect workpiece misalignment (adaptation)
$\hat{\mathbf{f}}_c = \text{const.} \Rightarrow \phi, \vec{\mathbf{r}}_i = \hat{\mathbf{f}}_c^n \times \bar{\mathbf{f}}_{c,i} \Rightarrow \mathbf{R}_{wp}$
$\hat{\mathbf{v}} = \text{const.} \Rightarrow \phi, \vec{\mathbf{r}}_i = \hat{\mathbf{v}}^n \times \bar{\mathbf{v}}_i \Rightarrow \mathbf{R}_{wp}$
Determine regions of acceptable motion and force (human inconsistency)
$\hat{\mathbf{f}}_c = \text{const.} \Rightarrow \mathbf{x}'(s) = \mathbf{R}_{wp}^T(\mathbf{x}_i(s) - \mathbf{x}_i(s=0)) \Rightarrow \mathbf{X}_A$
$\hat{\mathbf{v}} = \text{const.} \Rightarrow \hat{\tilde{\mathbf{f}}}_{c,i}(s) = \mathbf{R}_{wp}^T \hat{\mathbf{f}}_{c,i}(s) \Rightarrow \mathbf{F}_A$
disturbance
Design a robot controller for each segment such that:
$\mathbf{x}^r \in \mathbf{X}_A$, $\mathbf{f}^r \in \mathbf{F}_A$, $|\mathbf{f}^r| \le f_{max}$
allowable error
Identify a termination condition for each segment to allow the robot to switch between segments.

Figure 6.1: Flow Chart for Generating a Robot Program for a Contact Task

The use of human inconsistency is a consistent theme throughout the thesis. In Chapter 2, human inconsistency is used to identify an obstacle free region for pick and place tasks. A buffer between the robot and regions boundaries ensures obstacle avoidance, even in the presence of robot errors. This approach is extended in Chapter 3 to include regions of acceptable force direction and force magnitude.

Another theme is the use of information from the environment to adapt to workpiece misalignment. Robot adaptation is performed by switching between subtasks, and through use of a compliance controller within each subtask. Segmenting the task into subtasks, shown in the second block of the flow chart, relies on changes in information from the environment. Indeed, any section in which no new information is acquired from the environment can be performed as a single subtask. The subtask termination conditions are identified by changes in sensory information that correspond to subtask completion. Finally, the robot compliance controller for motion within a single subtask, is specified to adapt to workpiece orientation misalignment that is detectable from the sensory information.

Future Research

The methods presented in this thesis can be extended to include a larger scope of tasks. Regardless of the task, human inconsistency is an indicator of task accuracy requirements, and human adaptation occurs in response to detected variations in the environment.

To apply PHD to a larger range of tasks, one of the most useful extensions of the approach would be consider tasks that include part rotation as well as translation. An additional extension would be to allow the human to use information accumulated from multiple contact states, which as indicated in Chapter 3, is assumed not to occur. Removing the restriction that the sequence of contact state is constant, would also increase the scope of application. However, as indicated in Appendix I, this limitation does not preclude numerous useful tasks.

APPENDICES

Appendix I: Tasks With The Same Sequence of Contact States

The scope of this thesis is limited in application to simple assembly tasks referred to as contact tasks. The task is performed with only part translation, but no rotation. In addition, it is assumed that the demonstrator performs the task with the same sequence of contact states in each demonstration.

To extend this research to more useful assembly tasks it is necessary to extend the analysis to include part rotation. However, many useful tasks can still be performed with the same sequence of contact states, even in the presence of workpiece misalignment. Villarreal and Asada [1991] present the example task of assembling a lid onto a box. As shown in Figure AI.1, this task can be performed with a sequence of three contact states: the lid is moved into contact with the side of the box, the lid slides up the side, and finally the lid is rotated into place. Even with significant misalignment in box location and orientation, this strategy will succeed.

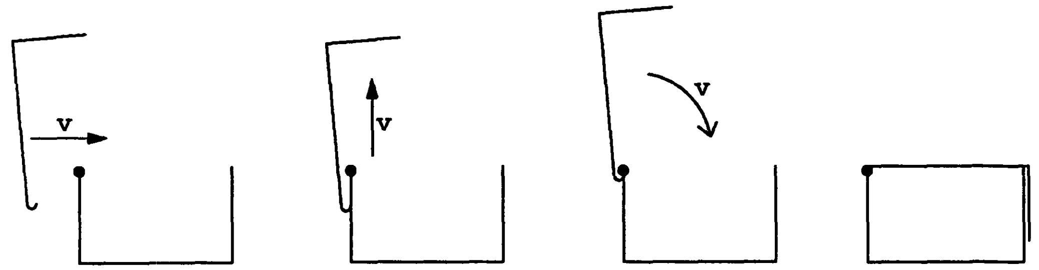


Figure AI.1: Assembly of Lid Onto Box

Another example task which has been frequently addressed in robotics, is the task of inserting a peg into a hole, as shown in Figure AI.2. There are two approaches for performing this task. The approach shown in Figure AI.2a, is to align the peg with the nominal hole orientation and attempt a direct insertion. If there exists misalignment in the hole location, the peg could contact either side of the hole, and thus the contact state sequence will vary between demonstrations. Whitney [1982] presents a robot compliance controller that can successfully implement this task for a range of misalignments, regardless of which side of the hole the peg contacts first.

An alternative approach for performing the peg in hole task is presented by Inoue [1974], and is shown in Figure AI.2b. Here the peg is purposely tilted in one direction so that the initial contact between the peg and the hole will always occur at the same side. Thus, the task can be performed with the sequence of contact states for a significant range of workpiece misalignment. In addition, this approach does not require chamfers on the hole or peg.

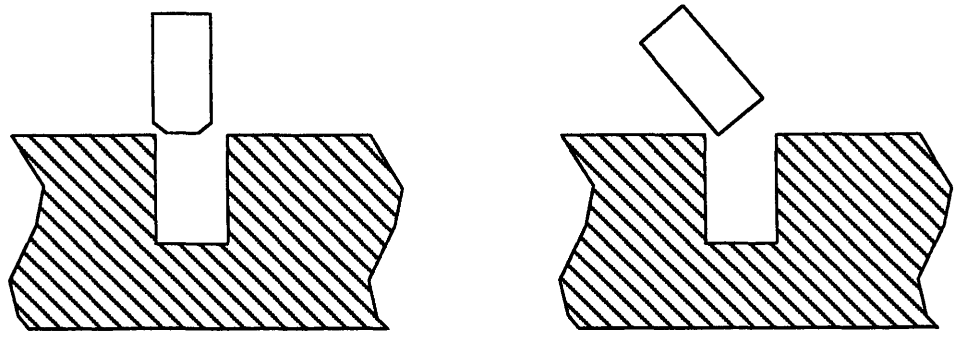

a. peg nominally aligned with hole
b. peg purposefully tilted

Figure AI.2: Strategies For Peg In Hole Task

Certain tasks may always result in a variety of contact state sequences, and as workpiece misalignment increases inevitability the contact state sequence could vary with almost any task. However, there exist a wide range of useful tasks, which can be performed with the same sequence of contact states.

Appendix II: Piecewise Linear Approximation

A trajectory can be approximated with a sequence of straight line segments, through piecewise linear approximation. Piecewise linear approximation was initially applied in the PHD research, to approximate a demonstrated position trajectory. However, this same method can be used when segmenting a task into subtasks. In the segmentation process it is necessary to evaluate whether a segment of motion has a constant direction of motion and thus lies on a straight line. The approach presented here applies a statistical hypothesis test to evaluate whether deviation from a straight line is likely due to noise in the sensor measurement, or actual change in trajectory. The approach is presented in terms of approximating a demonstrated trajectory, but can be directly applied to the segmentation algorithm.

The algorithm is applied to the initial points in the trajectory, by performing a least squares fit to a straight line. The number of points approximated by a single straight line is increased until an error criterion is violated. With such an approach there is a tradeoff between the number of linear segments and the level of accuracy. A high level of accuracy in duplicating the trajectory can be achieved, yet at the expense of a large number of segments. Thus, to implement this algorithm a criterion is established to determine when to end one straight line segment and begin the next.

Methods have been developed to approximate measured data by piecewise linear segments [Abdelmalek 1990 and Dunham 1986]. However these methods are primarily applied to pattern recognition and are used with vision systems. This analysis specifically addresses human position trajectories. In the case of task segmentation, each segment of motion constrained in two directions, is actually in a straight line and the deviations from the straight line are due to sensor noise. This model is used to determine when a segment of motion can be considered to be in a straight line.

A direct approach for evaluating when to end a segment is to compare the least squares error to a constant threshold, as done by Abdelmalek [1990]. However, with this method, the longer the trajectory continues the more the total error will increase, and eventually the cumulative error of any measured trajectory will exceed the constant threshold. Thus, a long subtask consisting of straight motion, will be mistakenly divided into multiple segments. A simple approach would be to divide the cumulative error by the number of points in the trajectory. However, the analysis presented here shows that this approach is not statistically valid. The appropriate criterion is determined by a hypothesis test at a given confidence level, which evaluates whether the data in a segment corresponds to the model of straight line motion with the addition of random noise. The resulting threshold criterion is presented in Equations AII.13 and AII.14, depending on whether the noise is Gaussian or uniform.

The human position trajectory is given by the vector $\mathbf{x}^h(t)$, and the steps to the algorithm are shown below.

Step 1: Select a starting point, $\mathbf{x}_{beg}$, for the current straight line segment. For the first segment set the starting point equal to the initial human point $\mathbf{x}^h(0)$. Otherwise set $\mathbf{x}_{beg}$ to the end of the previous straight line segment.

Step 2: Select a segment of the demonstration data to be approximated by a straight line. The trajectory $\mathbf{x}^h$ is sampled with a period of T, and the segment to be evaluated, $\mathbf{x}$, is the discrete series given by:

$$\mathbf{x}_i = \mathbf{x}^h\,(iT + t_{beg}) \quad i = 1,2,3 \ldots N \qquad \text{(AII.1)}$$

The beginning time, t_{beg}, is the starting point of the segment. For the first segment t_{beg} is set to zero, otherwise t_{beg} is set to correspond to the last point of the previous segment. The number of points in the segment is given by N. When beginning a segment N is set to three; the lowest number of points for which a straight line approximation is necessary. In step 5 the approximation is evaluated, and if it is valid, then N is increased by one.

Step 3: Utilize the gripper status information. Wherever the demonstrator opens or closes the gripper, it indicates an important and necessary point in the task. Accordingly, whenever, the gripper opens or closes, a new straight line segment is started, regardless of straight line approximation technique. Return to step 1.

Step 4: Perform a straight line fit on the data $\mathbf{x}$ using the least squares method, in which the straight line is restricted to passing through $\mathbf{x}_{beg}$. Select the coordinate of $\mathbf{x}$ with the largest variation as an independent parameter. For the purposes of illustration, it is assumed that for the current segment x is the independent parameter. For a two dimensional line in the xy plane, the equation for a straight line starting at the point (x_{beg},y_{beg}) is given by $(y-y_{beg})=(x-x_{beg})m_y$, where m_y is the slope of the line. The least squares method is used to find the slope of the line that best fits the data **x**. The problem is formulated into the form of the linear equation **b=Am** as follows.

$$\mathbf{b}=\begin{bmatrix}(y_1-y_{beg})\\(y_2-y_{beg})\\\vdots\\(y_N-y_{beg})\end{bmatrix}\quad \mathbf{A}=\begin{bmatrix}(x_1-x_{beg})\\(x_2-x_{beg})\\\vdots\\(x_N-x_{beg})\end{bmatrix}\quad \mathbf{m}=[m_y] \qquad \text{(AII.2)}$$

The solution to **m** that minimizes the error $|\mathbf{b}-\mathbf{Am}|^2$ is given by $\mathbf{m}=(\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\mathbf{b}$ [Strang 1986 Sec. 1.4]. The solution reduces to.

$$m_y=\frac{\sum_{i=1}^{N}(x_i-x_{beg})(y_i-y_{beg})}{\sum^{N}(x_i-x_{beg})^2} \qquad \text{(AII.3)}$$

The value of the error reduces to:

$$e_y=\sum_{i=1}^{N}(y_i-y_{beg})^2-\frac{\left[\sum_{i=1}^{N}(x_i-x_{beg})(y_i-y_{beg})\right]^2}{\sum^{N}(x_i-x_{beg})^2} \qquad \text{(AII.4)}$$

An efficient way to calculate the above two equations is to store the results of the summations. When an additional point is added to **x**, only the contribution due to the new point needs to be calculated.

This method of straight line approximation can be directly extended to trajectories with a dimension higher than two. The minimization of error in the one coordinate is independent of minimization of error in other coordinates. Accordingly, the slope in each dependent coordinate is calculated separately. For a three dimensional trajectory, the slope in the z direction is calculated by replacing the y coordinates with the z coordinates in the above equations. The straight line error, e_{sl}, is the sum of the error squared in each dependent coordinate, and for the three dimensional case is:

$$e=e_y+e_z \qquad \text{(AII.5)}$$

Step 5: Evaluate whether the least squares straight line calculated in the previous step results in an acceptable level of approximation, or whether it is necessary to begin a new straight line segment. A statistical test is performed to identify if the approximation is within a specified confidence level.

It is assumed that the human motion can be modeled as piecewise linear with the addition of random noise. While this assumption cannot be guaranteed, it has proven useful for a variety of tasks. The statistical analysis is presented for the case of a two dimensional trajectory where x is the independent variable and y is the dependent variable. Higher dimensions are addressed at the end of this step. For a segment of motion in which the human is in a linear segment, the measured trajectory is given by:

$$n_i + (y_i - y_{beg}) = (x_i - x_{beg})\, m_y \qquad \text{(AII.6)}$$

where n_i is noise contributing to the ith measurement. Within a segment consisting of a single subtask the only error is due to the noise and the error for the current segment of motion, e, is given by:

$$e = \sum_{i=1}^{N} n_i^2 \qquad \text{(AII.7)}$$

It is assumed that the noise values are random and independent from each other. Thus, as the value of N increases the Central Limit Theorem applies [Drake 1967] and the distribution of e can be approximated by a Gaussian distribution. When the segment of motion, **x**, corresponds to a single section of straight line human motion (eq. AII.6), the error distribution is Gaussian, and is larger otherwise. Accordingly, a hypothesis test can be performed, at a desired significance level, to determine if a straight line approximation of a segment should be accepted or rejected. A threshold level for the error, $e_{threshold}$, is determined for which a straight line approximation should be accepted as long as e is less than $e_{threshold}$. Selecting a significance level of 95%, $e_{threshold}$ is given by:

$$e_{threshold} = E(e) + 2\,\sigma_e \qquad \text{(AII.8)}$$

where E(e) and σ_e are the expected value and standard deviation of the error, e. To evaluate the above equation it is necessary to determine the values of E(e) and σ_e.

The value of e is equal to a summation (eq. AII.7) of independent variables, and thus its expected value and standard deviation are given by:

$$E(e) = N\,E(n^2) \quad \text{and} \quad \sigma_e^2 = N\left(\sigma_{n^2}\right)^2 \tag{AII.9}$$

The expected value of any variable squared is given by [Drake 1967 (Section 2.6)]:

$$E(n^2) = \sigma_n^2 + [E(n)]^2 \tag{AII.10}$$

It is assumed that the noise is unbiased, and thus E(n)=0. Incorporating E(n) into the above equation and combining it with the equation AII.9, results in:

$$E(e) = N\left(\sigma_n\right)^2 \tag{AII.11}$$

An expression for σ_{n^2} is found by substituting n with n^2 in equation AII.10, resulting in:

$$\left(\sigma_{n^2}\right)^2 = E(n^4) - \left[E(n^2)\right]^2 \tag{AII.12}$$

The value of σ_{n^2} depends on the distribution of the noise. A common assumption is that the noise has a Gaussian distribution with a standard deviation, σ_n, centered about zero. To solve equation AII.12, the expected value of n^4 is calculated by using the equation describing the Gaussian distribution and taking its fourth moment. The value of σ_{n^2} is calculated to be:

$$\sigma_{n^2} = \sqrt{2}\left(\sigma_n\right)^2 \quad \text{for Gaussian noise} \tag{AII.13}$$

An alternate assumption is that the noise has a uniform distribution centered about zero with a width of 2a. For this case the value of σ_{n^2} is calculated to be:

$$\sigma_{n^2} = 2a^2/\sqrt{45} \quad \text{for uniform noise} \tag{AII.14}$$

The error threshold level, for the case of a Gaussian distribution, can be calculated by substituting equations AII.11 and AII.13 into equations AII.8 and AII.9.

$$e_{threshold} = \sigma_n^2 [N + 2\sqrt{2N}] \quad \text{for Gaussian noise} \tag{AII.15}$$

The same approach is used for the uniform distribution, using equation AII.14 instead of AII.13. The standard deviation for a uniform distribution is, $\sigma_n^2 = a^2/3$, and the threshold level is given by:

$$e_{threshold} = \frac{a^2}{3} [N + 2\sqrt{4N/5}] \quad \text{for uniform noise} \tag{AII.16}$$

If for the current segment, the error is less than $e_{threshold}$, then accept the segment, add a new point to the segment and return to step 2. Otherwise, define the previous point as the end of a straight line segment. Using the last value of the independent parameter identify the endpoint of the straight line segment, which becomes the beginning point of the next segment., and return to step 1 to begin a new segment. The algorithm is completed when the last point in the demonstrated trajectory is reached.

The decision criterion whether to start a new straight line segment or not has been is directly related to the noise level in the system. As new points are added to the segment being evaluated, the magnitude of N and $e_{threshold}$ change to reflect the change in the statistical distribution of e.

The above analysis was presented for the case of a two dimensional trajectory. For a three dimensional trajectory the least squares error is given in terms of the error in the two independent direction, which here is taken to be the y and z directions. Assuming that the noise in each axes has the same statistical distribution, a single variable, n, can be used to represent the noise, and the error is given by:

$$\begin{aligned} e &= \sum_{i=1}^{N} n_{y,i}^2 + \sum_{i=1}^{N} n_{z,i}^2 \\ &= \sum^{2N} n_i^2 \end{aligned} \tag{AII.17}$$

The only difference between the above equation and equation AII.7 is that the summation is over 2N variables instead of N. Accordingly, the only modification necessary in calculating the error threshold in equations AII.15 and AII.16 is to replace N with 2N.

Example

A pick and place task was demonstrated on the Twin Arm robot at the Daikin facilities, and is shown on the video tape. Figure AII.1 shows the piecewise linear approximation of the demonstrated trajectory, along the y and z axes. The sampled points are indicated by the "x" marks, and the approximation is shown in a solid line. Text indicates the start and stop points, and word "grip" indicates locations where the gripper was open or closed.

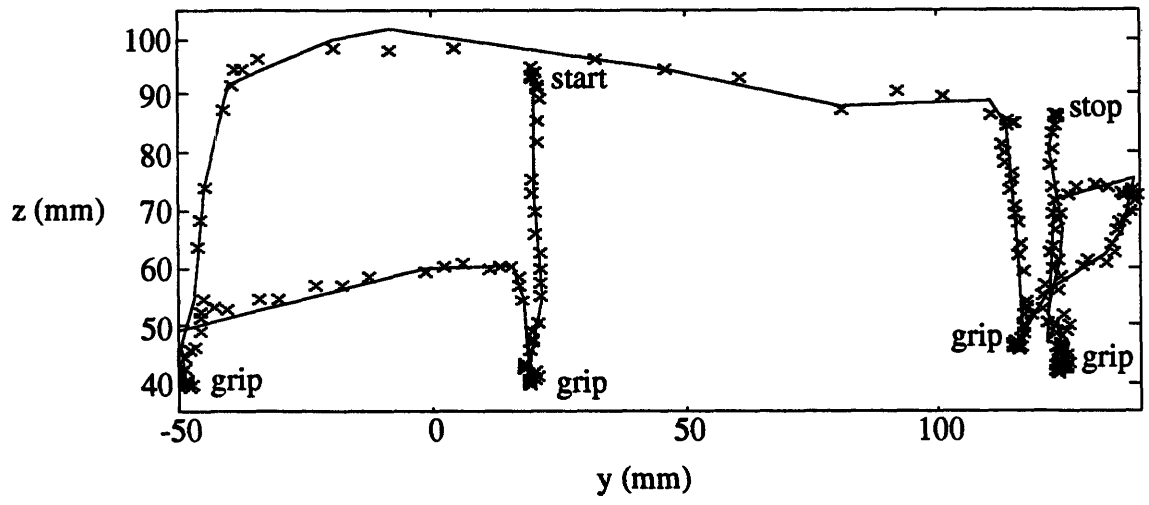


Figure AII.1: Piecewise linear approximation of the demonstration trajectory used to program the Daikin Twin Arm Robot for a pick and place task.

For the trajectory shown in Figure AII.1, it was assumed that the noise was Gaussian with a standard deviation of 1.0 mm. A 95% confidence test was used to determine when a new segment should be started (eq. AII.15). In addition, whenever the gripper opened or closed, a new straight line segment was started exactly at the point where the gripper is open or closed, which was implemented by turning on the fixed_edge_exact flag in the source code.

Appendix III: 3D Obstacle Free Trajectories

The method presented in Chapter 2 generates obstacle free robot trajectories for pick and place tasks. The region between the demonstrations was shown to correspond to human inconsistency, and a robot trajectory is guaranteed to be obstacle free as long as it remains within that region. However, the algorithm presented in Chapter 2 can only be applied to 2D translation. The Chapter 2 method identifies the boundaries of the obstacle free region, and then synthesizes the shortest robot path within this region. However, this method can not be directly extended to 3D. This appendix presents an alternate method which is first applied to 2D motion and then extended to 3D. This method synthesizes an obstacle free robot trajectory without identifying the boundaries of the obstacle free region. In addition the algorithm incorporates an optional buffer between the robot trajectory and the boundary of the obstacle free region, to allow for robot position errors. The shortest path within a 2D region is synthesized and the same method is extended to 3D, resulting in satisfactory 3D robot trajectories that are not necessarily the shortest.

2D Motion

The dimension of both a trajectory and a boundary of a 2D region are of order one. Thus, the boundary of the 2D obstacle free region is composed of segments of demonstrated trajectories. However, in 3D the boundary of a region becomes a two dimensional surface, and the boundary of a region cannot be composed of segments of individual trajectories. Thus, the original method used to analyze 2D cannot be directly extended to 3D.

An alternate 2D approach which can be extended to 3D is outlined here. This method evaluates whether a proposed robot trajectory is within the obstacle free region, without defining the boundaries of that region. A valid robot trajectory stays within the obstacle free region, and at each point along the trajectory there exists a demonstrated trajectory on the right and left side. Figure AIII.1 shows a straight line segment of a proposed robot trajectory. The proposed trajectory remains within the obstacle free region up until the point x_{valid}, after which there are no demonstrated trajectories on the right side. Thus by finding where the demonstrated trajectories intersect the proposed trajectory and keeping track of which side they are on, it is possible to evaluate whether a proposed trajectory is in the obstacle free region. The original method was based on identifying intersections between demonstrations, whereas in this method the intersections are found between individual demonstrations and the proposed trajectory. It is not necessary to define Jordan curves in this method, although the intersections between the proposed and demonstrated trajectories generate Jordan curves on either side of the proposed trajectory.

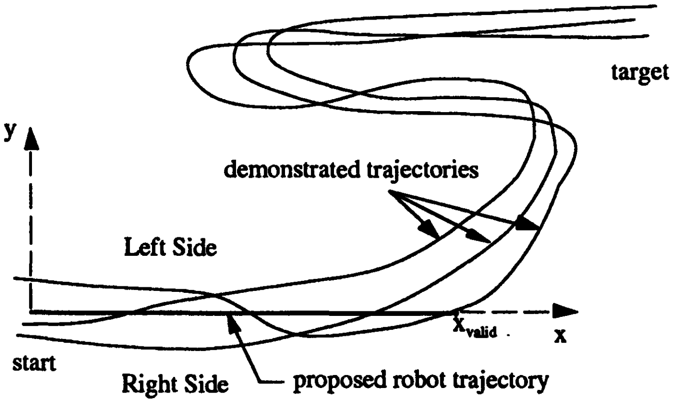


Figure AIII.1: Evaluation of a Proposed Robot Trajectory
The segment of proposed robot trajectory is within the obstacle free region up until x_{valid}, after which there is no longer a demonstration on the right side.

A modification to the approach outlined above incorporates a buffer between the robot trajectory and the boundary of the obstacle free region, which increases the distance between the robot and the obstacles. The buffer is implemented by requiring that a demonstration be the width of the buffer away from the proposed trajectory before it can be designated as being on the right or left side. The details of the method follow.

A coordinates system is defined that is aligned with the proposed robot trajectory, as shown in Figure AIII.1. Each human trajectory is transformed to this coordinate system and represented by, $\mathbf{h}_i(s)$, where the subscript "i" indicates the demonstration number, "s" is the distance traveled, and each point in the trajectory has an "x" and "y" value. The side of the demonstration is a function of "s" and is given by:

$$\text{side}(\mathbf{h}_i(s)) = \left\{ \begin{array}{lll} 1 & y \geq +\text{buffer} & \text{(Left side)} \\ 2 & y \leq -\text{buffer} & \text{(Right side)} \\ 0 & \text{Otherwise} & \text{(In buffer)} \end{array} \right\} \qquad \text{(AIII.1)}$$

For each demonstration the sets R_i and L_i are found which are the regions where the demonstrations is on the right and left of the proposed trajectory. The locations where $\mathbf{h}_i(s)$ intersect the proposed trajectory are found when side($\mathbf{h}_i(s)$) changes value and $x \geq 0$, and are designated by s_j, where $j=1,2,\ldots,J$. (To simplify notation the subscript "i" is not used with the variable s_j with the understanding that the values are recalculated for each demonstration). The actual values of s_j are found through linear interpolation between the sampled data points. The region where the demonstrations are on the right, between intersections 1 and J is given by:

$$R_i = \bigcup_{j=1}^{J-1} \left[x(s_j),\ x(s_{j+1})\right] \quad \text{for side}\left(\mathbf{h}_i(s_j) + \varepsilon\right) = 2 \qquad \text{(AIII.2)}$$

where ε is a small number used to detect the side following intersection "j". The region on the left is given by:

$$L_i = \bigcup_{j=1}^{J-1} \left[x(s_j),\ x(s_{j+1})\right] \quad \text{for side}\left(\mathbf{h}_i(s_j) + \varepsilon\right) = 1 \qquad \text{(AIII.3)}$$

If the side prior to $x(s_1)$ is not in the buffer, then the region $[0,\ x((s_1)]$ is appended to either R_i or L_i according to the side of $x(s_1)$. In the same fashion, if the side after to $x(s_J)$ is not in the buffer, then the region $[x((s_J),\ +\infty)$ and is appended to either R_i or L_i according to the side of $x(s_J)$.

The length of the proposed trajectory which remains inside the obstacle free region is determined by taking the union of R_i and L_i for all N demonstrations and then finding the maximum value of x_{valid} which satisfies the following condition.

$$[0,\ x_{valid}] \in \bigcup_{i=1}^{N} R_i \quad \text{and} \quad [0,\ x_{valid}] \in \bigcup_{i=1}^{N} L_i \qquad \text{(AIII.4)}$$

Figure AIII.1 shows a single segment of the proposed robot trajectory beginning at the starting location. However, a complete robot trajectory is synthesized from a sequence of straight line segments using a method described in the following section. In this process, it becomes necessary to evaluate segments that are continuations of previous segments. The same analytical technique is applied as in the case of the first segment, yet the proposed straight line segment begins at the end of the previous segment. The coordinate system used in the analysis is always aligned with the current segment of proposed robot trajectory and the origin is at the beginning of that segment.

Synthesizing a 2D Robot Trajectory

Now that a method has been formulated to evaluate whether a proposed trajectory is within the obstacle free region, it is necessary to synthesize an appropriate robot trajectory. Human inconsistency provides a range of acceptable motion, and selecting the appropriate trajectory depends on one's application. The method presented here determines the shortest 2D robot path from the start to target positions when the buffer is set to zero. With a non zero buffer the robot distance is no longer minimized, yet the distance between the robot and the obstacles is increased. Alternate performance criteria include maximizing the distance between the robot trajectory and the obstacles, or finding the minimum time trajectory by incorporating robot dynamics. Regardless of the criterion used, the range of human inconsistency allows one to improve robot performance above that of duplicating individual demonstrations.

The method used to select the shortest path presented in Chpater 2, is applied here with small modifications since the the boundaries of the region are not availaible.

1. Define the start and target positions, $\mathbf{p}_{start}$ and $\mathbf{p}_{target}$, as the average of the demonstration starting and ending positions. Set the beginning position of the current segment, $\mathbf{p}_{beg}$, to $\mathbf{p}_{start}$.
2. Guess an initial direction of a straight line that departs the starting point.
3. Evaluate the proposed straight line segment. Find the length, x_{valid}, and the side where lack of demonstrations limits the length of the line (for example the right side in Figure AIII.1).
4. Propose a new line, by either incrementing or decrementing the slope of the previous line in the direction towards the opposite side found in step 3. Evaluate this line and continue modifying the slope until a line is found that is limited by the side opposite that of step 3. Eventually both "right" and "left" lines are found.
5. Select a new line between the right and left by dividing the angle between them. Evaluate this line, and replace the previous "right" or "left" line according to the side of the new line.

6. Repeat the previous step until both the right and left lines converge, which is indicated when the angle between them is less than a desired accuracy threshold. A robot path segment is defined in the direction of convergence with a length of the shorter of the right and left lines.
7. Define a new beginning point, $\mathbf{p}_{\text{beg}}$, at the end of the previous robot segment. Repeat the above steps to find the next robot segment, starting at step 2. The shortest path is completed when the target can be reached with an obstacle free straight line from the starting point of the previous segment.

The procedure for converging on a line that intersects both the right and left side identifies the only direction that is not excluded from being the shortest path. After the first intersection between the selected line and the boundary, a new direction for the shortest path is possible. Thus, a new segment of the robot path is started and the convergence procedure is repeated. To implement this method one need only determine what side of the boundary a proposed line intersects, which is given by the side limited first by the lack of demonstrations (Figure AIII.1). Thus, the shortest path within the obstacle free region is generated without specifying the boundaries of the region.

3D Motion

Obstacle avoidance in 3D is in general more difficult than in 2D, and methods that work in 2D cannot always be extended to 3D [Latombe 1991]. In this section the 2D motion analysis is extended to 3D with appropriate modifications.

In 2D a proposed robot trajectory has a right and left side, and as long as demonstrated trajectories are present in both sides the proposed trajectory is obstacle free [Figure AIII.1]. This approach can be extended to 3D by defining a coordinate system with the "x" axis aligned with a segment of the proposed robot trajectory, as shown in Figure AIII.2. However, in 3D there exist four quadrants in which demonstrated trajectories exist.

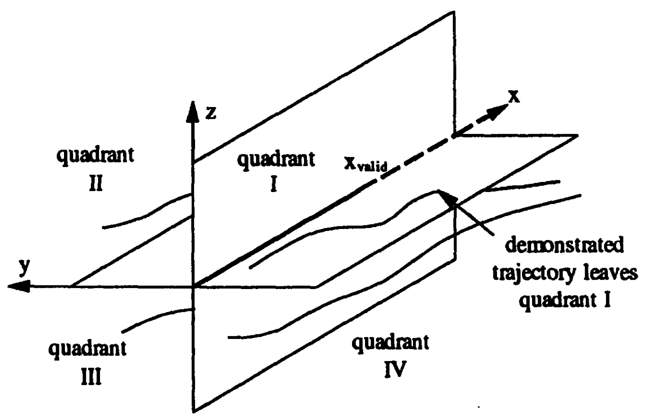


Figure AIII.2: Coordinate System for 3D Motion
The proposed robot trajectory is aligned with the "x" axis. As long as there is a demonstration in each quadrant, the robot trajectory is within the obstacle free region.

A proposed trajectory is obstacle free if there exist at least one demonstration in all of its quadrants from start to the target. Figure AIII.3 shows a cross section of demonstrated trajectories and the obstacle free region they define. The cross section of the total obstacle free region is defined by enclosing the demonstrated trajectories with straight line interpolations between the demonstrations. As in the 2D case it is assumed that the human uses the same obstacle avoidance strategy in all demonstrations, thus ensuring that the region between demonstrated trajectories is obstacle free. In 3D a specific requirement is that the cross section of an obstacle does not intersect any of the straight lines between the demonstrations at the same cross section.

The advantage of the quadrant method is that it simplifies the analysis, since boundaries of the 3D region do not need to be calculated. However, as a trade off, only a subset of the possible obstacle free region is identified as valid robot trajectories, as shown in Figure AIII.3.

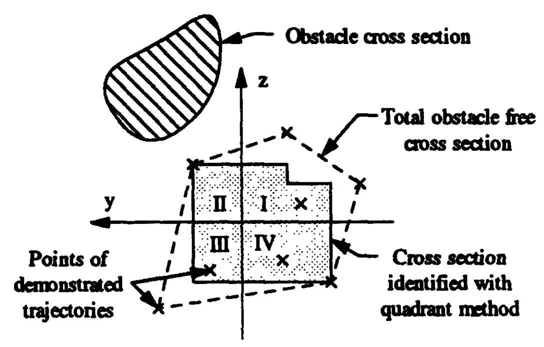

Obstacle cross section
z
Total obstacle free cross section
y
II
I
III
IV
Points of demonstrated trajectories
Cross section identified with quadrant method

Figure AIII.3: Cross Section of the Obstacle Free Region

The quadrant method identifies a subset of the total obstacle free region defined by the demonstrations.

To determine the length, x_{valid}, of a proposed robot trajectory that is within the obstacle free region, the same steps are followed as in the 2D analysis. However, Equation AIII.1 is replaced by:

$$\text{side}(\mathbf{h}_i(s)) = \left\{ \begin{array}{ll} 1 & y \leq -\text{buffer} \;\&\; z \geq +\text{buffer} \\ 2 & y \geq +\text{buffer} \;\&\; z \geq +\text{buffer} \\ 3 & y \geq +\text{buffer} \;\&\; z \leq -\text{buffer} \\ 4 & y \leq -\text{buffer} \;\&\; z \leq -\text{buffer} \\ 0 & \text{Otherwise} \end{array} \right\} \qquad \text{(AIII.5)}$$

In Equations AIII.2, AIII.3, and AIII.4 references to the right and left sides are replaced by the quadrants I, II, III, and IV. The overall approach for evaluating a single segment of proposed robot trajectory does not change between 2D and 3D.

Synthesizing a 3D Robot Trajectory

Selecting an appropriate 3D trajectory is done by following the same steps used in the 2D algorithm, yet the steps that converge on the desired robot trajectory require modification. Moreover, the resulting robot trajectory is not guaranteed to be the shortest as in the case with 2D motion.

As in the 2D case, the desired robot trajectory is identified by converging on a line that intersects the boundary of the obstacle free region on two sides, where a small change in slope will change the side of the boundary that is intersected. In the 3D case the quadrant method of evaluating trajectories identifies four possible sides by which a proposed line can exit the obstacle free region. An example is shown in Figure AIII.2 where the length of the proposed trajectory is limited when no demonstrations remain in quadrant I, and is interpreted as intersecting the boundary on side 1. In 3D a range of line directions exists where a small change in slope will the change the side of the boundary intersected. Thus, the convergence process does not result in a unique line direction that is the shortest robot path, but does identify a satisfactory trajectory that incorporates a buffer and removes much of the unnecessary human motion.

Converging on a desired line is performed by stepping diagonally into the interior of the region. Thus, if the first guess results in a line that exits the region on the side of quadrant I, then the slope of the line is stepped towards quadrant III, and if a line exits on quadrant II, the slope of the line is stepped towards quadrant IV, and so on. Each step is recorded and when the proposed line is has already been tried, the step size is reduced by half. This process is repeated until the step size is less than the desired accuracy threshold. The result converges to a line direction that intersects two boundaries, yet the solution is not unique and thus depends on the initial guess.

Experimental Results for 3D

An example 3D translational task of moving a sphere around a single obstacle was performed. Figure AIII.4 shows the demonstrated trajectories in dotted lines, and the synthesized robot trajectory in bold. A total of fifteen demonstrations were performed with a sample frequency of 20Hz. The average distance of the human demonstrations was 528mm and the minimum was 490mm. A robot trajectory was synthesized using a buffer of 2.0mm, and with a distance of 479mm.

Here the robot path is almost 10% shorter than the average demonstrations, and 2% shorter than the minimum trajectory. The decrease in the robot travel distance is less in the 3D case than the 2D case, which may be partially due to the amount of variation in the demonstrations. In addition, the 3D analysis robot path is not the shortest possible because the effective obstacle free region is reduced by using the quadrant approach, and the convergence method used does not identify the shortest path within this reduced obstacle free region. Overall, the approach presented in this article generates satisfactory robot trajectories using a simple algorithm. Similar tradeoffs between robot performance and complexity of the analysis are common in model based obstacle avoidance analysis [Latombe 1991].

The robot trajectory is improved over any single demonstrated trajectory both by the buffer which increases distance from the obstacles and by the removal of unnecessary human motion. The front view in Figure AIII.4 shows how the robot trajectory remains close to the inside curve of the demonstrated trajectories, and avoids unnecessary motion. In order to maintain a demonstrated trajectory in each quadrant, however, the distance cannot be minimized in all 2D projections, as shown in the top and side views.

If there are too few demonstrations, the quadrant approach can reduced the effective cross section of the obstacle free region to single point or line. Under these circumstance, the convergence of an obstacle free robot trajectory becomes numerically impractical. This problem is solved by increasing the number of demonstrations, which provides additional information regarding the obstacle free region and the task accuracy requirements.

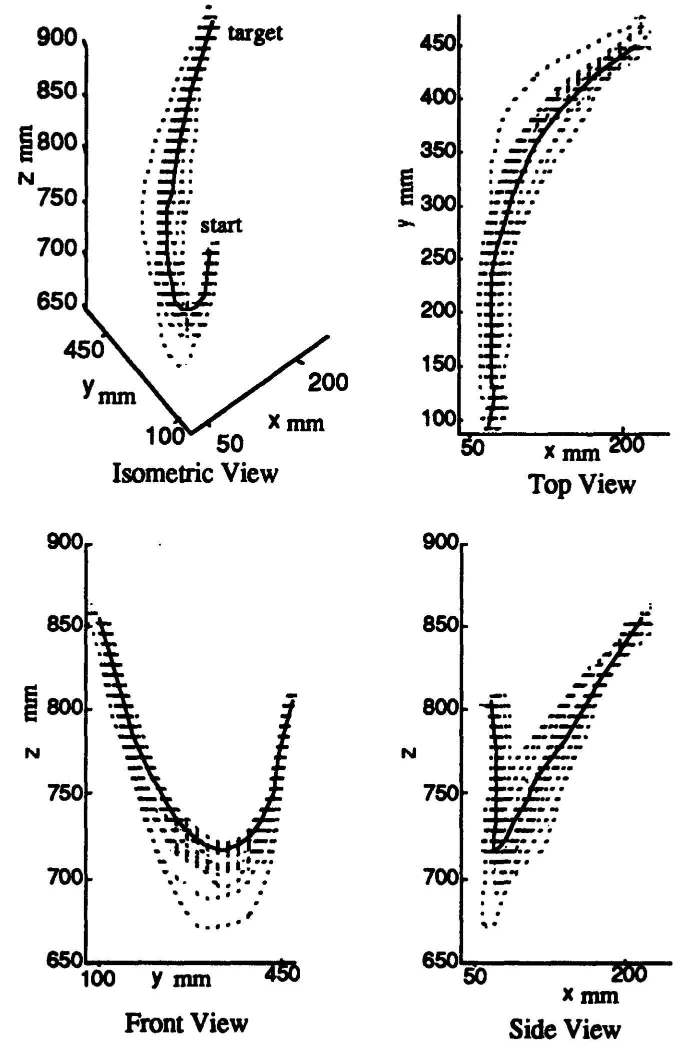


Figure AIII.4: 3D Experimental Results
The demonstrations are dotted lines, and the robot trajectory is a solid line.

Summary

An alternate algorithm is presented for generating obstacle free rbot trajectories within the obstacle free region defined from human inconsistency. For the case of 2D translation the algorithm generates a robot trajectory that is the shortest path within the identified obstacle free region. For 3D translation a robot trajectory is generated that also avoids obstacles, yet is not necessarily the shortest path within the region. Experimentally robot paths are generated for both 2D and 3D tasks that are shorter than any of the demonstrations, and do not contain the "wiggles" present in some demonstrations.

The range of human inconsistency can also be used to incorporate a buffer between the robot and the obstacles. The buffer can guarantee obstacle avoidance even if there are errors in the robot position controller, as long as the buffer is larger than the maximum robot error.

Appendix IV: Compliance Controller Performance

When performing assembly tasks it is necessary to avoid excessive forces that may damage the parts. Specifying the robot arm impedance is an effective method of limiting contact forces. Otherwise, even a small workpiece misalignment could result in large contact forces, if the manipulator is infinitely stiff. The approach selected is to control the robot arm so that it reacts with the environment as a linear spring, which is referred to as compliance control. Compliance control is specified such that the force exerted by the robot, $\mathbf{f}^r$, is given by:

$$\mathbf{f}^r = \mathbf{K}(\mathbf{x}_{ref} - \mathbf{x}^r) \qquad \text{(AIV.1)}$$

where $\mathbf{x}^r$ is the robot gripper position, $\mathbf{x}_{ref}$ is the reference position, and $\mathbf{K}$ is a positive definite stiffness matrix. If the robot stiffness is too high, then workpiece misalignment could result in excessive contact forces. However, if the robot stiffness is too low than excessive position errors could occur.

Hogan [1988] demonstrated that a compliant controller is stable when the manipulator is in contact with a passive environment, which is the typical case for assembly operations. It was further shown by Whitney [1982] that by selecting an appropriate robot compliance, the robot could adapt to workpiece misalignment for the peg in hole task. Schimmels and Peshkin [1992] presented a method of specifying a robot control law in which the robot is programmed to model a damper. Both the work by Whitney, and Schimmels and Peshkin, identified a robot controller which could implement motion consisting of a sequence of contact states. They also considered tasks which required both part translation and rotation. However, they assumed that the orientation misalignment of the workpiece was infinitesimal, and thus did not change the direction of contact forces.

The analysis presented here is applied to motion within a single contact state of a contact task, in which the part motion consists of only translation. The analysis here, however, does consider the effect of workpiece orientation misalignment, and addresses all possible constraint configurations in a contact task. Furthermore, robot performance is evaluated in terms of position and force errors. Xiao [1991] does address finite orientation misalignments when the evaluating the performance of damping control. However, Xiao only considers force errors caused by orientation misalignment, and neglects position errors which is an essential performance criteria for assembly operations. The analysis presented here is for use with the PHD approach, however the results are also valid for modeling compliance control in model based programming methods.

Problem Definition

Robot performance is evaluated in term of position and force errors due to orientation misalignment of the workpiece, for motion within a single contact state. The overall robot controller detects contact with a surface using a subtask termination condition. The reference trajectory for motion within the contact sate is then specified from the point of contact. However, the reference trajectory is specified assuming that the workpiece is in the nominal orientation. In this appendix the actual robot and forces are calculated when there is misalignment in workpiece orientation. The analysis is performed first assuming that there is no friction, and the effect of friction is considered. It is assumed that the robot motion is quasi-static and dynamic forces can be neglected.

The desired robot position trajectory when the workpiece is in the nominal orientation is given by $\mathbf{x}^n(s)$, and the desired force trajectory without friction is given by $\mathbf{f}_c^n(s)$. The coefficient of friction can vary for each workpiece, and therefore friction is treated as a disturbance. The compliance controller reference trajectory is calculated by substituting the nominal trajectories for the robot trajectories in Equation AIV.1. Thus, when the robot is in the nominal location the desired trajectories will be achieved. The reference trajectory is given by:

$$\mathbf{x}_{ref} = \mathbf{K}^{-1}\,\mathbf{f}^n + \mathbf{x}^n \qquad \text{(AIV.2)}$$

The desired robot trajectory is to maintain the same relative motion between the part and the workpiece that occurs when the workpiece is in the nominal position. In the case where the orientation of the workpiece is rotated, the desired force and position trajectories are calculated by rotating the nominal trajectories. The rotation of the workpiece is specified by a 3x3 rotation matrix **R**, which is nonsingular. The robot position is measured in a coordinate system with its origin at point where initial contact with the surface occurs. Accordingly, the transformation from a nominal position to a desired position is a pure rotation. The superscript "d" designates the desired trajectories, and the desired position trajectory, x^d, is given by:

$$x^d(s) = R\, x^n(s) \qquad \text{(AIV.3)}$$

and the desired constraint force, f_c^d, is given by:

$$f_c^d(s) = R\, f_c^n(s) \qquad \text{(AIV.4)}$$

The robot performance is quantified by comparing the actual robot positions and forces to the desired values.

Constrained Motion

In the assembly process when there is contact between the part and the workpiece, the motion of the part is constrained and the directions of admissible motions and admissible forces are restricted. The nature of constraints were formalized by Mason [1981], and the geometric constraints can be expressed in terms of an admissible motion matrix A_p, and admissible force matrix, A_f. The directions of admissible motion are tangent to the constraining surface. The admissible force is the component of the contact force that is not due to friction, and its possible directions are normal to the constraining surfaces. The columns of A_p and A_f form a basis for the vector spaces of admissible motion and admissible forces, and are selected to be orthonormal. The direction of admissible force at the constraint is normal to the direction of motion and produces no work, which is represented by:

$$(A_f)^T A_p = [0] \qquad \text{(AIV.5)}$$

There are two possible constraint configuration for 3D translation, which are shown in Figure AIV.1. The number of constraints are designated by 'm'.

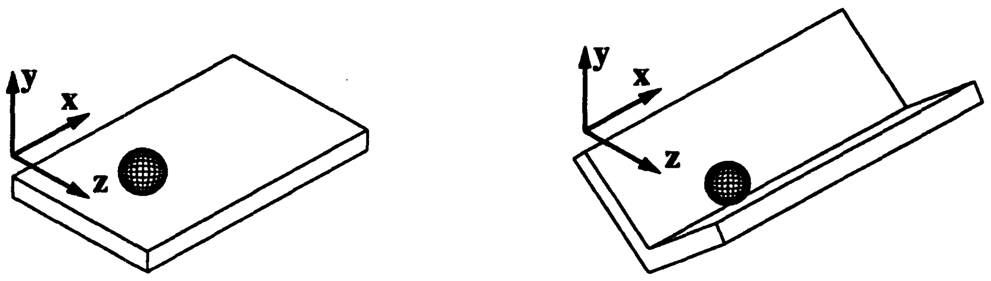

y
x
z
a. Single direction of constraint
y
x
z
b. Two directions of constraint

Figure AIV.1: Two Possible Constraint Configurations for 3D Translation

In the nominal configuration the admissible motion and force matrices are designated by the superscript "n". In the single constraint configuration (figure AIV.1a) the admissible motion and force matrices are:

$$\mathbf{A}_p^n = \begin{bmatrix} 1 & 0 \\ 0 & 0 \\ 0 & 1 \end{bmatrix} \qquad \mathbf{A}_f^n = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} \qquad \text{for } m = 1 \qquad \text{(AIV.6)}$$

In the two constraint configuration (figure AIV.1b) the admissible motion and force matrices are:

$$\mathbf{A}_p^n = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \qquad \mathbf{A}_f^n = \begin{bmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix} \qquad \text{for } m = 2 \qquad \text{(AIV.7)}$$

The constrained motion and force can be defined in terms of admissible motion and force variables, $\mathbf{a}_p$ and $\mathbf{a}_f$. The nominal position trajectory is:

$$\mathbf{x}^n = \mathbf{A}_p^n \mathbf{a}_p^n d^n = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} d^n \qquad \text{(AIV.8)}$$

where $\mathbf{a}_p^n$ is the unit vector of nominal admissible motion variables, and d^n is the nominal distance from the origin.

The nominal constraint force, $\mathbf{f}_c^n$, is given by:

$$\mathbf{f}_c^n = \mathbf{A}_f^n \mathbf{a}_f^n f^n = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} f^n \qquad \text{(AIV.9)}$$

where $\mathbf{a}_f^n$ is the unit vector of nominal admissible force variables, and f^n is the nominal force magnitude. In the nominal single constraint configuration the unit vector admissible motion and force variables are:

$$\mathbf{a}_p^n = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \qquad \mathbf{a}_f^n = [1] \qquad \text{for } m = 1 \qquad \text{(AIV.10)}$$

In the nominal two constraint configuration the admissible unit vector motion and force variables are:

$$\mathbf{a}_p^n = [1] \qquad \mathbf{a}_f^n = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \qquad \text{for } m = 2 \qquad \text{(AIV.11)}$$

Performance With Zero Friction

When there is no friction, then the force exerted by the environment onto the robot lies normal to the constraint surfaces, and thus is in the admissible force space. As long as motion remains in the same contact state and the friction force is zero, then energy in the system is conserved. In a conservative system the robot position is independent of the trajectory history [Crandall et al 1985]. Therefore, the robot position can be calculated as a function of the nominal distance from the origin, 'd^n', without regard to the shape of the trajectory $\mathbf{x}^n(s)$.

The rotation of the workpiece varies the directions of admissible motion and force. The admissible motion matrix encountered by the robot, $\mathbf{A}_p^r$, is given by rotating the columns of the nominal admissible motion matrix.

$$\mathbf{A}_p^r = \mathbf{R}\ \mathbf{A}_p^n \qquad \text{(AIV.12)}$$

The admissible force matrix encountered by the robot, $\mathbf{A}_f^r$, is given by rotating the columns of the nominal admissible force matrix.

$$\mathbf{A}_f^r = \mathbf{R}\ \mathbf{A}_f^n \qquad \text{(AIV.13)}$$

It is assumed that the contact points on the part do not change throughout the task, and that the workpiece surface is flat (i.e. polyhedral shaped). Thus for translational motion the admissible motion and force spaces remain constant throughout the task.

The robot position trajectory is defined in terms of the admissible motion vector, $\mathbf{a}_p^r$, and the distance from the origin, d^r, by:

$$\mathbf{x}^r = \mathbf{A}_p^r\, \mathbf{a}_p^r\, d^r \qquad \text{(AIV.14)}$$

The robot constraint force is defined in terms of the admissible force vector, $\mathbf{a}_f^r$, and the magnitude of the force , f^r, by:

$$\mathbf{f}_c^r = \mathbf{A}_f^r\, \mathbf{a}_f^r\, f^r \qquad \text{(AIV.15)}$$

The robot controller is specified by the compliance controller (eq. AIV.1) and the reference trajectory (eq. AIV.2). The resulting motion is calculated by imposing the constraint that the actual motion lie in the admissible motion space (eq. AIV.14), and that the force lie in the admissible force space (eq. AIV.15). At this point of the analysis it is assumed that there is no friction, and the affect of friction is incorporated in the following section. Thus substituting equations AIV.8, AIV.9, AIV.2, AIV.14, and AIV.15, into equation AIV.1 results in:

$$\mathbf{A}_f^r \mathbf{a}_f^r f^r - \mathbf{A}_f^n \mathbf{a}_f^n f^n = \mathbf{K}\left[\mathbf{A}_p^n \mathbf{a}_p^n d^n - \mathbf{A}_p^r \mathbf{a}_p^r d^r\right] \qquad \text{(AIV.16)}$$

The left hand side of equation AIV.16 is the difference between the actual force and the nominal force, and the right side is the difference between the actual position and the nominal position multiplied by the stiffness matrix **K**. The robot motion is defined in terms of the admissible motion variables, $\mathbf{a}_p^r$, and the distance from the origin, d^r, which are calculated by premultiplying equation AIV.16 by $(\mathbf{A}_p^r)^T$. The product $(\mathbf{A}_p^r)^T\mathbf{A}_f^r$ is equal to zero (eq. AIV.5). The direction of the constraining surface is given in terms of **R** by substituting in equations AIV.12 and AIV.13, to give:

$$\mathbf{a}_p^r\, d^r = \left(\mathbf{A}_p^{n^T} \mathbf{R}^T \mathbf{K}\, \mathbf{R}\, \mathbf{A}_p^n\right)^{-1}\left[\mathbf{A}_p^{n^T} \mathbf{R}^T \left(\mathbf{f}_c^n + \mathbf{K}\, \mathbf{x}^n\right)\right] \qquad \text{(AIV.17)}$$

The robot contact force is defined in terms of the admissible force variables, $\mathbf{a}_f^r$, and the force magnitude f^r, and is calculated in a similar manner. Equation AIV.16 is premultiplied by $(\mathbf{A}_f^r)^T\mathbf{K}^{-1}$ and the null term $(\mathbf{A}_f^r)^T\mathbf{A}_p^r$ is removed, resulting in:

$$\mathbf{a}_f^r\, f^r = \left(\mathbf{A}_f^{n^T} \mathbf{R}^T \mathbf{K}^{-1}\, \mathbf{R}\, \mathbf{A}_f^n\right)^{-1}\left[\mathbf{A}_f^{n^T} \mathbf{R}^T \left(\mathbf{K}^{-1}\mathbf{f}_c^n + \mathbf{x}^n\right)\right] \qquad \text{(AIV.18)}$$

The matrices that are inverted in equations AIV.17 and AIV.18 are indeed nonsingular, since the matrix **K** is positive definite, and **R**, $\mathbf{A}_p$, and $\mathbf{A}_f$ are of full column rank [Strang 1986, sec. 1.4].

Thus, the robot position and forces in the presence of workpiece orientation misalignment have been identified. The robot trajectories can be calculated for an arbitrary **R** and **K**. The robot position is given by incorporating the result of equation AIV.17 into equation AIV.14. The robot force is given by incorporating the result of equation AIV.18 into equation AIV.15.

The performance of the robot can be calculated for an arbitrary stiffness matrix, yet for the purposes of controller design it is possible to restrict ourselves to a stiffness matrix that will simplify the analysis. The robot stiffness matrix is selected such that its axes are aligned with the nominal coordinate system, and **K** is given by:

$$\mathbf{K} = \begin{bmatrix} k_x & 0 & 0 \\ 0 & k_y & 0 \\ 0 & 0 & k_z \end{bmatrix} \qquad \text{(AIV.19)}$$

In addition, for small angles of workpiece misalignment, it is possible to consider the effect of rotations about the x, y, and z axes independently. Rotation matrices about the three axes are presented by Crandall et al [1985], and are designated by $\mathbf{R}_x$, $\mathbf{R}_y$, and $\mathbf{R}_z$.

The robot force and position trajectories are presented in Figures AIV.2 and AIV.3 for the case of small angles of misalignments and for a stiffness matrix aligned with the nominal coordinate system.

In the single constraint configuration the 'x' axis is aligned with the vector from the origin to the nominal part position. Rotation about the x axis results in a position error in the 'z' direction, and an error in the force magnitude; rotation about the y axis does not change the robots position or force; and rotation about the z axis results in a position error in the 'x' direction and an error in the force magnitude.

In the two constraint configuration the 'y' axis is aligned with the nominal force direction. Since there is only one direction of admissible motion, no errors in the direction of motion occur, and errors in the distance traveled do not effect performance. Accordingly, position errors do not occur when the part is constrained in two directions. Rotation about the 'x' axis does not change the robot position or force; rotations about the 'y' axis results in an error in the force magnitude and direction; and rotation about the 'z' axis results in an error in the force magnitude.

In Chapter 3, Section 3.4.1, directions of detectable workpiece misalignment are identified. In the single constraint configuration, rotation about the 'y' is undetectable, and in the two constraint configuration rotation about the 'x' axis is undetectable. In both of these undetectable directions, the robot forces and positions did not vary in response to workpiece misalignment. Indeed any such variation would be impossible, since it would indicate that the compliance controller 'detected' these misalignments.

In Section 3.5.1, the robot performance equations are further simplified by requiring that the components of the stiffness matrix, k_x, k_y, and k_z are all within the same order of magnitude. Under these circumstances and using the small angle approximation, the following approximation for an expression in Figure AIV.2 can be made:

$$k_y \sin^2\phi + k_x \cos^2\phi \approx k_x \qquad \text{(AIV.20)}$$

Similar approximation in Figures AIV.2 and AIV.3 are possible, which is how the results in Figures 3.14 and 3.15 are generated.

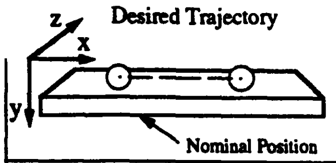
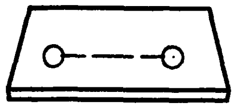
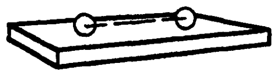
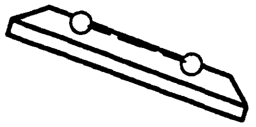
	Axis of workpiece rotation	Robot Postion and Force with Compliant Controller
	ϕ_x	$x^r = x^d + \frac{f^n \sin\phi}{k_y \sin^2\phi + k_z \cos^2\phi}\begin{bmatrix} 0 \\ \sin\phi \\ -\cos\phi \end{bmatrix}$ $f_c^r = f_c^d \left\{ \frac{k_z \cos\phi}{k_y \sin^2\phi + k_z \cos^2\phi} \right\}$
	ϕ_y	$x^r = x^n = \begin{bmatrix} d^n \\ 0 \\ 0 \end{bmatrix} \neq x^d$ $f_c^r = f_c^n = f_c^d$
	ϕ_z	$x^r = x^d \left\{ \frac{\left(k_x \cos\phi + \frac{f^n \sin\phi}{d^n} \right)}{k_y \sin^2\phi + k_x \cos^2\phi} \right\}$ $f_c^r = f_c^d \left\{ \frac{k_x k_y \left(\frac{\cos\phi}{k_y} - \frac{d^n \sin\phi}{f^n} \right)}{k_y \sin^2\phi + k_x \cos^2\phi} \right\}$

Figure AIV.2: Compliance Controller Performance with a Single Direction of Constraint

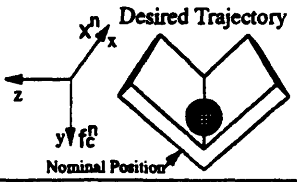
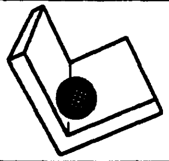
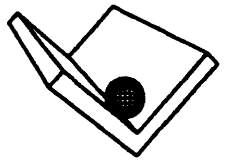
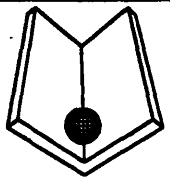
	Axis of workpiece rotation	Robot Postion and Force with Compliant Controller
	ϕ_x	$x^r = x^n = x^d$ $f_c^r = f_c^n = \begin{bmatrix} 0 \\ f^n \\ 0 \end{bmatrix} \neq f_c^d$
	ϕ_y	$x^r = x^d$ $f_c^r = f_c^d + \frac{k_x k_z d^n \sin\phi}{k_z \sin^2\phi + k_x \cos^2\phi}\begin{bmatrix} \sin\phi \\ 0 \\ \cos\phi \end{bmatrix}$
	ϕ_z	$x^r = x^d$ $f_c^r = f_c^d \left\{ \frac{k_x k_y \left(\frac{\cos\phi}{k_y} - \frac{d^n \sin\phi}{f^n} \right)}{k_y \sin^2\phi + k_x \cos^2\phi} \right\}$

Figure AIV.3: Compliance Controller Performance with Two Directions of Constraint

Affect of Friction on Robot Performance

When friction is present, the system is no longer conservative, and the robot position depends on the trajectory history. Since the trajectory history depends on the desired robot trajectory it can be arbitrary. Accordingly, a simplified robot trajectory is evaluated, and the results are the applied to other trajectories with a worst case analysis. It is assumed that the desired robot trajectory is straight line motion from the origin. The trajectory can be generated by monotonically increasing d^n in Equation AIV.8.

The first section of the friction analysis identifies the sticking condition and determines under what conditions friction will prevent the desired sliding between the part and the workpiece. The second section determines the effect of friction on the direction of motion when motion does occur.

To simplify notation, the superscript "r" is dropped in reference to robot motion, in the friction analysis section. The total force applied by the robot, **f**, in the case of quasi-static motion is given by:

$$\mathbf{f} = \mathbf{f}_c + \mathbf{f}_f \qquad \text{(AIV.21)}$$

where $\mathbf{f}_c$ is the constraint force normal to the workpiece surface, and $\mathbf{f}_f$ is the friction component and is tangential to the surface. It is assumed that the friction can be modeled as Coulomb friction with an isotropic coefficient of friction, μ. Accordingly, the maximum magnitude of the friction force, $f_{f,max}$, is:

$$f_{f,max} = \mu N \qquad \text{(AIV.22)}$$

where N is the magnitude of the normal force.

Avoiding Sticking

Sticking has been identified as a potential problem in assembly [Lozano-Pérez et al. 1984, Whitney 1982, and Ohwovoriole 1980], and it occurs when the friction force is greater than or equal to the force in the direction of possible motion. The direction of the friction force is tangential to the constraint surface and lies in an admissible motion direction. Accordingly, $\mathbf{f}_f$ can be defined in terms of the admissible motion matrix, and the unit vector $\mathbf{a}_\mu$ which defines the direction of the friction force. For the purposes of analyzing sticking it is assumed that the coefficient of friction is equal to the sticking value, μ^*, which is just large enough to create sticking. Accordingly, the friction force is equal to the maximum friction force and is given by:

$$\mathbf{f_f} = \mu^* \, f \, \mathbf{A_p} \, \mathbf{a_\mu} \qquad \text{for } |\mathbf{v}| = 0 \qquad \text{(AIV.23)}$$

where f is the magnitude of the constraint force (eq. AIV.15). The compliance control equation including friction is given by incorporating eq. AIV.21 into eq. AIV.1.

$$\mathbf{f_c} + \mathbf{f_f} = \mathbf{K} \, (\mathbf{x_{ref}} - \mathbf{x}) \qquad \text{(AIV.24)}$$

In the case of sticking, the robot position does not move from the origin, and **x** is equal to the null vector. Equation AIV.24 is expanded by incorporating the reference trajectory (eq. AIV.2), and by expressing $\mathbf{f_c}$ and $\mathbf{f_f}$ in terms of admissible force and motion matrices (eq. AIV.15 and AIV.23) to give:

$$\mathbf{A_f} \, \mathbf{a_f} \, f + \mu^* \, f \, \mathbf{A_p} \, \mathbf{a_\mu} = \mathbf{f_c^n} + \mathbf{K} \, \mathbf{x}^n \qquad \text{(AIV.25)}$$

The magnitude of the constraint force, f, is calculated by multiplying equation AIV.25 by $(\mathbf{A_f})^T$. The product $(\mathbf{A_f})^T\mathbf{A_p}$ is equal to zero (eq. AIV.5), and the product $(\mathbf{A_f})^T\mathbf{A_f}$ is equal to the identity matrix because columns of $\mathbf{A_f}$ are orthonormal vectors. The vector $\mathbf{a_f}$ has unit length, and thus f is given by:

$$f = \left| \, (\mathbf{A_f})^T \, \mathbf{f_c^n} + (\mathbf{A_f})^T \, \mathbf{K} \, \mathbf{x}^n \, \right| \qquad \text{(AIV.26)}$$

The coefficient of friction for sticking, μ^*, is calculated by multiplying equation AIV.25 by $(\mathbf{A_p})^T$. The product $(\mathbf{A_p})^T\mathbf{A_f}$ is equal to zero, and the product $(\mathbf{A_p})^T\mathbf{A_p}$ is equal to the identity matrix. The vector $\mathbf{a_\mu}$ has unit length, and thus the coefficient μ^* in terms of f is given by:

$$\mu^* = \frac{1}{f} \left| (\mathbf{A_p})^T \, \mathbf{f_c^n} + (\mathbf{A_p})^T \, \mathbf{K} \, \mathbf{x}^n \right| \qquad \text{(AIV.27)}$$

Incorporating values for $\mathbf{x}^n$ (eq. AIV.8), $\mathbf{f_c^n}$ (eq. AIV.9), and **K** (eq. AIV.19) and combining the previous two equations results in the following expression for μ^*.

$$\mu^* = \left| \frac{f^n \, (\mathbf{A_p})^T \, \hat{\jmath} + d^n \, k_x \, (\mathbf{A_p})^T \, \hat{\imath}}{f^n \, (\mathbf{A_f})^T \, \hat{\jmath} + d^n \, k_x \, (\mathbf{A_f})^T \, \hat{\imath}} \right| \qquad \text{(AIV.28)}$$

where $\hat{\imath}$ and $\hat{\jmath}$ are unit vectors along the x and y axis respectively. In the above equation, the first component in both the numerator and denominator remains constant while the second component varies with d^n

which increase monotonically as the reference trajectory advances. In the case of sticking, the part remains stationary as d^n increases. The coefficient of friction necessary to keep the part stationary, μ^*, is calculated by taking the limit as d^n approaches infinity.

$$\lim_{d^n \to \infty} \mu^* = \lim_{d^n \to \infty} \left| \frac{d^n k_x (A_p)^T \hat{i}}{d^n k_x (A_f)^T \hat{i}} \right| = \left| \frac{(A_p)^T \hat{i}}{(A_f)^T \hat{i}} \right| \qquad \text{(AIV.29)}$$

When the workpiece is in the nominal configuration (eq. AIV.8 and AIV.9), the numerator of AIV.29 is equal to one, and the denominator is equal to zero, resulting in an infinite value of μ necessary to maintain sticking. Thus for a finite μ, sticking will not occur in the nominal configuration. Figure AIV.4 shows the nominal configuration, the friction cone, and how the reference trajectory advances to eventually generate a force large enough to start motion.

The amount of variation in workpiece orientation that can occur without resulting in sticking is calculated be identifying the angle of workpiece rotation which would result in a sticking coefficient of friction, μ^*, that is equal to the actual coefficient of friction, μ. In the case of a single direction of constraint, A_f, is a column vector (eq. AIV.6) and the denominator of eq. AIV.29 is the dot product between two unit vectors that can be expressed in terms of the cosine of the angle between the vectors, β.

$$\left| (A_f)^T \hat{i} \right| = \cos(\beta) \qquad \text{(AIV.30)}$$

In the nominal configuration β is equal to $\pi/2$, and thus when the workpiece is rotated by an angle ϕ in a direction that changes the angle between the vectors, the result of eq. AIV.30 is equal to $\cos(\pi/2-\phi)$, as long as ϕ is less than π. Rotations of the workpiece in directions that do not change β, do not contribute to sticking. The numerator and denominator of eq. AIV.29 are the projection of the unit vector $\hat{i}$ into the subspace A_p and its orthogonal complement A_f, which decomposes the vector into two orthogonal components [Strang 1986 sec. 2.2]. The two components are at right angles to each other, and their magnitude can be combined using Pythagoras' theorem to give the total magnitude of the original unit vector:

$$\begin{aligned} \left|(A_p)^T \hat{i}\right|^2 &= 1 - \left|(A_f)^T \hat{i}\right|^2 \\ &= 1 - \cos^2(\beta) = \sin^2(\beta) \end{aligned} \qquad \text{(AIV.31)}$$

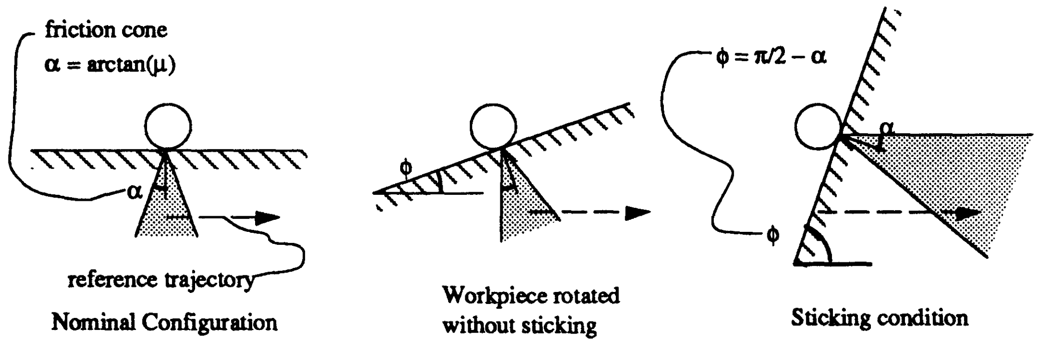


Figure AIV.4 : Sticking occurs only if the workpiece misalignment is greater than π/2-α, otherwise the reference trajectory escapes the friction cone.

The angle of workpiece rotation at which sticking occurs, ϕ_{stick}, is given by setting μ^* equal to μ, and by substituting equations AIV.30 and AIV.31 into equation AIV.29, to give:

$$\phi_{stick} = \frac{\pi}{2} - \arctan(\mu) \qquad \text{(AIV.32)}$$

The sticking angle calculated for the configuration with a single direction of constraint is the same as for the configuration with two directions of constraint. The change in the analysis is that the numerator is the dot product of $\hat{\imath}$ and A_f, which is equal to $\cos(\phi)$, and the denominator is equal to $\sin(\phi)$.

Figure AIV.4 shows how sticking can occur. The angle of the friction cone is equal to arctan(μ), and thus the workpiece can be rotated 90 degrees minus the angle of the friction cone (eq. AIV.32), before sticking will occur. Coefficients of friction are given in Marks' handbook table 3.2.1 [Avallone and Baumeister 1987], and the highest value of μ is for cast iron against cast iron, which has a static coefficient of friction 1.1, which corresponds to a friction cone of 48° and a sticking angle of 42°. Thus the analysis of a straight line task using a compliant controller indicates that sticking will not occur for small or moderate angles of variation in workpiece orientation.

Affect of Friction on the Direction of Motion

The preceding analysis established that friction will not result in sticking, yet it is also necessary to determine whether friction will contribute to position errors in robot motion. The robot position trajectories are calculated for the case where friction is present and sticking does not occur. In the case of quasi-static motion with an isotropic coefficient of friction, the direction of the friction force is opposite the direction of motion and is equal to its maximum value [Peshkin and Sanderson 1989]. Accordingly, the friction force is given in terms of the admissible motion variables, $\mathbf{a}_p$, and the dynamic coefficient of friction, μ.

$$\mathbf{f}_f = -\mu \, |\mathbf{f}_c| \, \mathbf{A}_p \, \mathbf{a}_p \qquad \text{for } |\mathbf{v}| > 0 \qquad \text{(AIV.33)}$$

The compliant control equation (AIV.24) is expanded as in the analysis of sticking while keeping the robot motion term (eq. AIV.14), and using the friction force during motion (eq. AIV.33).

$$\mathbf{A}_f \, \mathbf{a}_f \, \mathbf{f} - \mu \, |\mathbf{f}_c| \, \mathbf{A}_p \, \mathbf{a}_p = \mathbf{f}_c^n + \mathbf{K} \, [\mathbf{x}^n - \mathbf{A}_p \, \mathbf{a}_p \, d] \qquad \text{(AIV.34)}$$

The admissible motion variables, $\mathbf{a}_p$, can be calculated as a function of the distance from the origin, d, and the magnitude of the contact force, $|\mathbf{f}_c|$ by multiplying equation AIV.34 by $(\mathbf{A}_p)^T$.

$$\mathbf{a}_p = \left[(\mathbf{A}_p)^T \, \mathbf{K} \, \mathbf{A}_p \, d - \mu \, |\mathbf{f}_c| \, \mathbf{I} \, \right]^{-1} \left[(\mathbf{A}_p)^T(\mathbf{f}_c^n + \mathbf{K} \, \mathbf{x}^n)\right] \text{(AIV.35)}$$

The robot position trajectory with friction, $\mathbf{x}_{wfr}$, is given in terms of the admissible motion variable $\mathbf{a}_p$ (eq. AIV.35), and the admissible motion matrix (eq. AIV.14). The position, $\mathbf{x}_{wfr}$, can be expressed in terms of d^n by realizing that the magnitude of $\mathbf{x}_{wfr}$ is equal to d. The position trajectories for both constraint configurations corresponding to different workpiece orientations is shown in Figure AIV.5. When possible the result is expressed in terms of the position trajectory which occurs when no friction is present, designated by $\mathbf{x}_{nofr}$.

ϕ	One Direction of Constraint	Two Directions of Constraint
ϕ_x	$x_{wfr} = \left(\frac{1}{1+\frac{\mu\lvert f_c\rvert}{d\,k}}\right)\begin{bmatrix} d^n \\ 0 \\ 0 \end{bmatrix} + \frac{\mu\lvert f_c\rvert}{\frac{\mu\lvert f_c\rvert}{d} + k_y \sin^2\phi + k_z \cos^2\phi}\begin{bmatrix} 0 \\ \sin\phi \\ -\cos\phi \end{bmatrix}$	$x_{wfr} = x_{nofr}\left[1 - \frac{\mu\lvert f_c\rvert}{d^n\,k_x}\right]$
ϕ_y	$x_{wfr} = x_{nofr}\left[1 - \frac{\mu\lvert f_c\rvert}{d^n\,k_x}\right]$	$x_{wfr} = x_{nofr}\left[1 - \frac{\mu\lvert f_c\rvert}{d^n\,k_x \cos(\phi)}\right]$
ϕ_z	$x_{wfr} = x_{nofr}\left[1 - \frac{\mu\lvert f_c\rvert}{d^n\,k_x \cos(\phi) + f^n \sin(\phi)}\right]$	$x_{wfr} = x_{nofr}\left[1 - \frac{\mu\lvert f_c\rvert}{d^n\,k_x \cos(\phi) + f^n \sin(\phi)}\right]$

Figure AIV.5: Position trajectories with friction for different directions of workpiece rotation.

Friction does not change the direction of motion in five out of the six cases shown in Figure AIV.5, as indicated where x_{wfr} is equal to x_{nofr} multiplied by a scalar. In these five cases the scalar is less than unity, indicating that friction causes the robot position to be closer to the origin. However, in the single constraint configuration with rotation about the 'x' axis, the direction of motion changes. The magnitude of the position error normal to the desired direction of motion is given by:

$$x_{err,norm} = \frac{f^n \sin\phi}{k_y \sin^2\phi + k_x \cos^2\phi + \frac{\mu\lvert f_c\rvert}{d}} \quad \text{for } m=1 \text{ and } \phi=\phi_x \tag{AIV.36}$$

The magnitude of the position error with friction (eq. AIV.36) is similar to the expression for the position error without friction (Figure AIV.2, row 1), with the addition of the term ($\mu|f_c|/d$) in the denominator. This term is always positive, and thus the position error with friction is always less than the error without friction, as shown in figure AIV.6. Accordingly, friction does not increase robot motion error normal to the direction of motion.

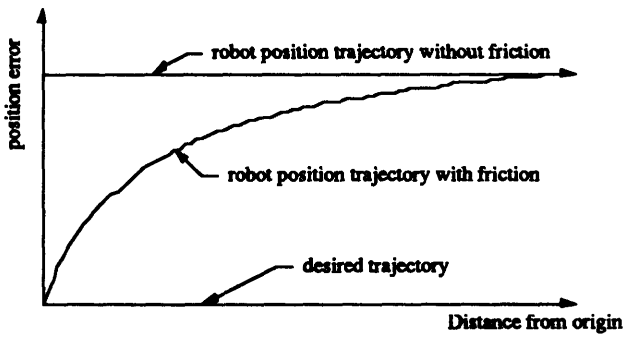


Figure AIV.6: Robot position trajectories with and without friction for the single constraint configuration with a workpiece rotated about the x axis

The amount of position error in the direction of motion is the same in all cases in Figure AIV.5, if one assumes a small angle of workpiece misalignment. The position error parallel to the direction of motion due to friction is:

$$x_{err,parallel} = \frac{\mu |f_c|}{k_x} \qquad \text{(AIV.37)}$$

If the desired trajectory is actually straight line motion, then one can neglect errors parallel to the direction of motion, since the robot will eventually reach the desired location, and therefore does not effect robot performance. However, for a general trajectory, a worst case analysis is performed and the position error due to friction is given by $x_{err,parallel}$, which for small angles of misalignment is given in eq. AIV.37.

The overall effect of friction is that it creates a position error in the direction of motion, but actually limits the position error normal to the direction of motion.

Summary

The analysis in this appendix quantifies compliance controller performance for the case of 3D translation. The results are simplified in their application in Chapter 3. Nevertheless, the complete analysis provides the verification of the simpler results, and can be used in actual applications where the small angle approximation or the simplification of **K** do not apply.

Appendix V: Termination Region Estimation

In Section 5.3 of Chapter 5, the termination region is identified by a section of the demonstration data that conforms to a model of the human motion. It is assumed that the human arm reacts as a compliant controller, and the human force $\mathbf{f}(t)$ is related to the human position, $\mathbf{x}(t)$ by the following equation.

$$\mathbf{f}(t) = \mathbf{K}\left(\mathbf{x}_{ref}(t) - \mathbf{x}(t)\right) \qquad \text{(AV.1)}$$

where $\mathbf{K}$ is a positive definite stiffness matrix. In addition, it is assumed that the reference trajectory is in the form of a straight line motion with constant speed, which is represented by:

$$\mathbf{x}_{ref}(t) = \hat{\mathbf{v}}_{ref}\, t + \mathbf{x}_{ref,0} \qquad \text{(AV.2)}$$

The model of motion within the termination region is given by substituting Equation AV.2 into Equation AV.1, to give:

$$\mathbf{f}(t) = \mathbf{K}\,\mathbf{x}(t) + \mathbf{K}\,\hat{\mathbf{v}}_{ref}\, t - \mathbf{K}\,\mathbf{x}_{ref,0} \qquad \text{(AV.3)}$$

The piecewise linear approximation algorithm presented in Appendix II, illustrates a method of defining a region of position trajectory that conforms to a straight line model. A least squares approximation of the data is performed, and the error between the model and the data is compared to a threshold. This same approach can be applied to identify the termination region.

A least squares approximation can be implemented if the data is presented in the form of the $\mathbf{A}\mathbf{y}=\mathbf{b}$, where $\mathbf{y}$ is the vectors of unknowns, $\mathbf{A}$ is a known matrix, and $\mathbf{b}$ is a known vector [Strang 1986]. In Equation AV.3, the knowns are $\mathbf{f}(t)$, $\mathbf{x}(t)$, and 't' which are measured by the teaching gripper, and the unknowns are $\mathbf{K}$, $\hat{\mathbf{v}}_{ref}$, and $\mathbf{x}_{ref,0}$. To implement a least squares approximation it is necessary to transform Equation AV.3 into the form of the linear equation $\mathbf{A}\mathbf{y}=\mathbf{b}$.

To get an intuitive understanding of the human model, it is useful to take the time derivative of both sides of Equation AV.3 to give:

$$\dot{\mathbf{f}}(t) = \mathbf{K}\,\dot{\mathbf{x}}(t) + \mathbf{K}\,\hat{\mathbf{v}}_{ref} \qquad \text{(AV.4)}$$

where the overscript dot indicates time derivative. If the part is moving with a constant velocity and then hits a constraint and stops, the force will continue to increase in proportion to $\mathbf{K}$. However, in practice taking the time derivative of the force and the position would increase the effect of noise in the system, and therefore Equation AV.4 is not useful for analytical purposes. Accordingly, the least squares approach is applied directly onto Equation AV.3 and where the time derivative is not used.

Another aspect of the analysis is that the matrix **K** is positive definite, and therefore symmetric. The least square approximation, should therefore maintain the symmetry of **K**.

The least square approach is applied by rewriting Equation AV.3 in the form of **Ay=b**. This approach is first applied to the two dimensional case. In 2D, the components of the symmetric matrix **K** are given by:

$$\mathbf{K} = \begin{bmatrix} k_{xx} & k_{xy} \\ k_{xy} & k_{yy} \end{bmatrix} \qquad \text{(AV.5)}$$

The quotient $\mathbf{K}\,\hat{\mathbf{v}}_{ref}$ is defined by the unknown variables:

$$\mathbf{K}\hat{\mathbf{v}}_{ref} = \begin{bmatrix} c_x \\ c_y \end{bmatrix} \qquad \text{(AV.6)}$$

The quotient $\mathbf{K}\,\mathbf{x}_{ref,0}$ is defined by the unknown variables:

$$\mathbf{K}\mathbf{x}_{ref,0} = \begin{bmatrix} d_x \\ d_y \end{bmatrix} \qquad \text{(AV.7)}$$

The least squares solution identifies the unknown variables that minimize the error between the data and the linear model. The matrices **A** and **b** contain the sampled data. For a given segment being evaluated the number of sample points is given by 'J' and the subscript 'j' indicates the sample number. Equation AV.3 is rewritten in terms of **Ay=b**, in the following equation.

$$-\begin{bmatrix} f_{x,1} \\ f_{y,1} \\ f_{x,2} \\ f_{y,2} \\ \vdots \\ f_{x,2} \\ f_{y,2} \end{bmatrix} = \begin{bmatrix} t & 0 & 1 & 0 & -x_1 & -y_1 & 0 \\ 0 & t & 0 & 1 & 0 & -x_1 & -y_1 \\ 2t & 0 & 1 & 0 & -x_2 & -y_2 & 0 \\ 0 & 2t & 0 & 1 & 0 & -x_2 & -y_2 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ Jt & 0 & 1 & 0 & -x_J & -y_J & 0 \\ 0 & Jt & 0 & 1 & 0 & -x_J & -y_J \end{bmatrix} \begin{bmatrix} c_x \\ c_y \\ d_x \\ d_y \\ k_{xx} \\ k_{xy} \\ k_{yy} \end{bmatrix} \qquad \text{(AV.8)}$$

where f_x and f_y are components of **f**, and x and y are components of **x**. The corresponding equation for the 3D case is given by:

$$-\begin{bmatrix} f_{x,1} \\ f_{y,1} \\ f_{z,1} \\ f_{x,2} \\ f_{y,2} \\ f_{z,2} \\ \vdots \\ f_{x,J} \\ f_{y,J} \\ f_{z,J} \end{bmatrix} = \begin{bmatrix} t & 0 & 0 & 1 & 0 & 0 & -x_1 & -y_1 & -z_1 & 0 & 0 & 0 \\ 0 & t & 0 & 0 & 1 & 0 & 0 & -x_1 & 0 & -y_1 & -z_1 & 0 \\ 0 & 0 & t & 0 & 0 & 1 & 0 & 0 & -x_1 & 0 & -y_1 & -z_1 \\ 2t & 0 & 0 & 1 & 0 & 0 & -x_2 & -y_2 & -z_2 & 0 & 0 & 0 \\ 0 & 2t & 0 & 0 & 1 & 0 & 0 & -x_2 & 0 & -y_2 & -z_2 & 0 \\ 0 & 0 & 2t & 0 & 0 & 1 & 0 & 0 & -x_2 & 0 & -y_2 & -z_2 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ Jt & 0 & 0 & 1 & 0 & 0 & -x_J & -y_J & -z_J & 0 & 0 & 0 \\ 0 & Jt & 0 & 0 & 1 & 0 & 0 & -x_J & 0 & -y_J & -z_J & 0 \\ 0 & 0 & Jt & 0 & 0 & 1 & 0 & 0 & -x_J & 0 & -y_J & -z_J \end{bmatrix} \begin{bmatrix} c_x \\ c_y \\ c_z \\ d_x \\ d_y \\ d_z \\ k_{xx} \\ k_{xy} \\ k_{xz} \\ k_{yy} \\ k_{yz} \\ k_{zz} \end{bmatrix} \qquad \text{(AV.9)}$$

The least squares solution is given by: $\mathbf{y}=(\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\mathbf{b}$. One could use the results to identify a human compliance, but to do this accurately one should ensure that the column rank of **A** is full. However, to identify a termination region, it is only necessary to identify the error between the data and the best linear fit. The same method presented in Appendix II is used; the least squares approximation is applied iterativly as one increases the number of points in the sample period, and a region where the model applies is identified as the termination region.

References

Asada, H., and Slotine, J.-J., 1986, Robot Analysis and Control, John Wiley and Sons

Asada, H., and Izumi, S., 1987, "Direct Teaching and Automatic Program Generation for the Hybrid Control of Robot Manipulators, "*Proc. IEEE International Conference on Robotics and Automation*, Raleigh, North Carolina, Vol. 3, pp. 1401-1406

Asada, H., and Hirai, S. 1989. "Towards a Symbolic-Level Feedback; Recognition of Assembly Process States," *Proc. 5'th Int. Symp. Robotics Research*, Tokyo 1989.

Avallone E. A. and Baumeister T., 1987, Marks' Standard Handbook for Mechanical Engineers, Ninth Edition, McGraw-Hill Book Co.

Cheng-Jung Chiu, 1994, "Use of A Redundant 3D Ultrasonic Sensor to Improve Accuracy, for Robot Programming by Human Demonstration," CIDMS Student Conference, MIT, Jan. 28-29.

Crandall S. H., Karnopp, D. C., Kurtz, E. F. Jr., Pridmore-Brown D. C., 1985, Dynamics of Mechanical and Electromechanical Systems, Robert E. Krieger Publishing Co., Malabar, Florida

Delson, N., and West, H., 1992 "Robot Programming by Human Demonstration," Japan-USA Symposium on Flexible Automation, San Francisco, July 1992, Vol. 2 pp. 1387-1397

Delson, N., and West, H., 1993a "Robot Programming by Human Demonstration: Subtask Compliance Controller Identification," Proc. of the 1993 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, pp. 33-41

Delson, N., and West, H. 1993b. "Robot Programming by Human Demonstration: Learning Straight Line Motion," *Proc. ASME Winter Annual Meeting*, New Orleans, Nov. 28 - Dec. 3, 1993, DSC-Vol. 49, pp. 351-365.

Delson, N., and West, H. 1994 "Robot Programming by Human Demonstration: The Use of Human Variation in Identifying Obstacle Free Trajectories," Proc. of the 1994 IEEE Int. Conf. on Robotics and Automation, San Diego, California, May 8-13 1994, pp. 564-571.

Desai, R., and Volz, R., 1989, "Identification and Verification of Termination Conditions in Fine Motion in Presence of Sensor Errors and Geometric Uncertainties," *1989 IEEE International Conference on Robotics and Automation*, Vol. 2 pp.800-807

Dessen F., and Balchen J. G., 1988, "A Positional Deviation Sensor for Training Robots", Modeling, Identification, and Control, Vol. 9. No. 2, pp. 99-108

Hannaford, B., and Lee, P., 1989, "Hidden Markov Model Analysis of Force/Torque Information in Telemanipulation," Proc. of the 1st Int. Symposium on Experimental Robotics, Springer Verlag, Montreal, June.

Harima T., and West, H., 1992 "Natural Robot Programming System" Proc. of the 1992 IEEE/RSJ International Conference on Intelligent Robots and Systems , July 7-10, Raleigh North Carolina pp. 750-756

Hirzinger G., and Heindl J., 1983, " Sensor programming - a new way for teaching a robot paths and force/torques simultaneously", *Intelligent Robots: Third Int. Conf. on Robot Vision and Sensory Controls*, Part II pp. 549-558.

Hogan, N., 1988, "On the Stability of Manipulators Performing Contact Tasks", *IEEE Journal of Robotics and Automation*, Vol. 4, No. 6 Dec. pp. 677-686

Ikeuchi, K. and Suehiro, T. 1992, "Towards an Assembly Plan from Observation," Proc. of the 1992 IEEE Int. Conf. on Robotics and Automation, Nice, France May 1992 pp. 2171-2177.

Ikeuchi, K., Kawade M., and Suehiro, T. 1993, "Towards an Assembly Plan from Observation Task Recognition with Planar, Curved and Mechanical Contacts," Proc. of the 1993 IEEE/RSJ International Conference on Intelligent Robots and Systems, July 26-30, Yokohama Japan, vol. 3, pp. 2294-2301.

Inoue, H. 1974, "Force Feedback In Precise Assembly Tasks." 1974 (Aug.) AIM-308. Cambridge MA: Massachusetts Institute of Technology Artificial Intelligence Laboratory. Reprinted in Winston, P. H. and Brown, R. H. eds. 1979. *Artificial Intelligence: An MIT Perspective*, Cambridge, MA, MIT Press.

Kosuge K., Fukuda T., and Asada H., 1991, "Acquisition of Human Skills for Robotic Systems," Proc. of the 1991 IEEE Int. Symposium on Intelligent Control, Aug. 13-15, Arlington, Virginia, pp. 469-474

Kuniyoshi, Y., Inaba, M., and Inoue, H., 1992, "Seeing, Understanding and Doing Human Task," Proc. of the 1992 IEEE Int. Conf. on Robotics and Automation, Nice, France May 1992 pp. 2-9

Latombe, J., 1991, Robot Motion Planning, Kluwer Academic Publishers, Boston/Dordrecht/London

Liu, S., and Asada, H., 1992, "Transferring Manipulative Skills to Robots: Representation and Acquisition of Tool Manipulative Skills Using a Process Dynamics Model," J. of Dyn. Sys., Meas. and Control, June, Vol. 114, pp. 220-228

Lozano-Pérez, T., 1983, "Robot Programming," Proc. of the IEEE, Vol. 71, No. 7, July, pp. 821-841

Lozano-Pérez, T., Mason, M. T.., and Taylor, R. H.. 1984. "Automatic Synthesis of Fine-Motion Strategies for Robots." *Int. J. Robotics Research* Vol. 3, No. 1, pp. 3-24

Lozano-Pérez, T., Jones, J. L., Mazer E., and O'Donnell P.A., 1992, HANDEY A Robot Task Planner, The MIT Press, London, England

Mason, M. T. 1981, "Compliance and Force Control for Computer Controlled Manipulators," *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-11 June 6 1981 pp. 418-432

McCarragher, B. J., and Asada H., 1993, "Qualitative Template Matching Using Dynamic Process Models for State Transition Recognition of Robotic Assembly," Transactions of the ASME. Journal of Dynamic Systems, Measurement and Control, vol. 115, no.2A, pp. 261-269

Munkres, J. R., 1975, Topology A First Course, Prentice-Hall, Inc., Englewood Cliffs, New Jersey

Ogata, H, and Takahashi, T., 1993, "A Geometric Approach to Task Understanding for Robotic Assembly Operations," IEEE Int. Conf. on Robotics and Automation, Atlanta, Georgia, May 2-6, pp.58-64

Ohwovoriole, M. S., 1980 "An Extension of Screw Theory and its Application to the Automation for Assembly Machines," Ph.D. Univ. of Michigan, Ann Arbor

Peshkin, M. A., and Sanderson, A. C., 1989 "Minimization of Energy in Quasi-Static Manipulation," *IEEE Transactions on Robotics and Automation*, Vol. 5, No. 1, pp. 53-60

Pinkney, J., 1993, "The Design of an Intuitive Teaching Interface for Robot Programming by Human Demonstration", Masters Thesis, Department of Mechanical Engineering, Massachusetts Institute of Technology, May 1993

Schimmels, J., and Peshkin, M., 1992 "Admittance Matrix Design for Force-Guided Assembly," *IEEE Transactions on Robotics and Automation*, Vol. 8, No. 2 April 1992 pp. 213-227

Strang G.,1986 Introduction To Applied Mathematics, Wellesley-Cambridge Press

Takahashi T., Ogata H., and Moto S., 1993, "Method for Analyzing Human Assembly Operations for Use in Automatically Generating Robot Commands." IEEE Int. Conf. on Robotics and Automation, Atlanta, Georgia, May 2-6, pp. 695-700

Villarreal, A., and Asada, H., 1991, "A Geometric Representation of Distributed Compliance for the Assembly of Flexible Parts," Proc. of the 1991 IEEE International Conf. on Robotics and Automation, Sacramento, California April 1991, pp. 2708-2715

West, H., 1986, "Kinematic Analysis for the Design and Control of Braced Manipulators", Ph.D. Thesis, Massachusetts Institute of Technology, Dept. of Mechanical Engineering.

Whitney, D. E., 1977, "Force feedback control of manipulator fine motions," *J. Dyn. Syst. Measurement Contr.*, June 1977 pp. 91-97

Whitney, D. E., 1982, "Quasi-static Assembly of Compliantly Supported Rigid Parts," *J. Dyn. Syst. Measurement Contr.* Vol. 104 pp. 65-77

Will, P. M., and Grossman, D. D., 1975, "An Experimental System for Computer Controller Mechanical Assembly," Trans. IEEE Computers, C-24(9), pp. 879-888

Xiao, J., 1991, "Force/Moment Constraints for Robot Compliant Motions in the Presence of Uncertainties," *Proc. of the 1991 IEEE Int. Sym. on Intelligent Control*, Aug. 13-15 Arlington Virginia pp. 122-127