Virtual Display Aids for Teleoperation

by

Ravi K. Chiruvolu

Submitted to the Department of Mechanical Engineering
in partial fulfillment of the requirements for the degrees of

Master of Science in Mechanical Engineering

and

Bachelor of Science in Mechanical Engineering

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

May 1991

Author .....
Department of Mechanical Engineering
May 10, 1991

Certified by .....
Thomas Sheridan
Professor of Mechanical Engineering
Thesis Supervisor

Accepted by .....
Ain A. Sonin
Chairman, Departmental Committee on Graduate Students

Virtual Display Aids for Teleoperation

by

Ravi K. Chiruvolu

Submitted to the Department of Mechanical Engineering
on May 10, 1991, in partial fulfillment of the
requirements for the degrees of
Master of Science in Mechanical Engineering
and
Bachelor of Science in Mechanical Engineering

Abstract

This thesis introduces a concept called a virtual image and discusses its applications to telerobotics. The virtual image is a graphical tool which allows the operator to see the environment from any perspective that he wants. This concept proves critical in two situations: 1) when a sensor (i.e. wrist camera) from which the operator needs information is obscured, and the information is not directly provided by any sensor, 2) when the task requires moving a selected object relative to another object which is itself moving. These two problems will be discussed in detail along with the method by which the virtual image concept offers the solution. A simulation was created in which the virtual image concept was tested under a varying set of parameters for a teleoperational task. The parameter categories are camera/graphical views, graphical display content, written display content, and operator interface. The simulation will be described in detail along with the experimental results and conclusions. In addition, the possibility of introducing automation as an aid for teleoperation will be discussed and a relevant simulation scenario written by the author will be analyzed. Throughout the paper, the real-world applicability of these ideas will be emphasized.

Thesis Supervisor: Thomas Sheridan
Title: Professor of Mechanical Engineering

Acknowledgments

I would first like to thank my parents for having me and for showing me so much love and support throughout the years. I love you, mom and dad! Next, I would like to express my thanks to Tom Sheridan who introduced me to the field of the telerobotics and human factors and really spent the time to structure my research into a successful thesis. Then, to everyone at the Mitre Corporation, especially Vincent Hwang, thank you for all your help and the opportunity to work on this exciting research. I would also like to thank Eric Huber and Marc Slack for not only providing helpful insights in my research, but for allowing me to crush them in racquetball. Finally, I would like to thank all my friends at ZBT, my hometown buddies in Edison, and just everyone that I have had such great laughs with over these last five years; for without you, college would have never been as fun.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

This thesis addresses the general problem of applying robotics in space. In particular, the main issue that will be explored is how to use computer graphics to aid an operator perform a teleoperated task. The concept introduced to aid the operator is called a virtual image. A virtual image is a graphical representation of the environment displayed in a computer simulation. It is derived from the raw data taken in by the robot's sensors which become inputs into the simulation program. In other words, instead of viewing a video monitor which displays the world as seen directly by a camera, an operator can view a 3D graphics display with the view transformed to any view the operator desires. This helps the operator in two situations: 1) when a needed view of a sensor (eg wrist camera) is obscured, and 2) when the task involves working with a moving object and this motion hinders the operator from performing his task. This proves especially valuable because, in any telerobotic system, the number of sensors is limited, and thus, there will always be information that the operator does not have. Even if there are a number of moveable cameras in the environment (i.e. attached to a robot's wrist), there will be many positions in space that are blocked from view by the robot and other objects. The critical idea behind the virtual image is that it does not require any additional sensor hardware; it uses the already acquired data. In addition, the increasing power and affordability of computers makes teleoperation using graphics displays practical today. One critical question addressed in this research is: what feedback will best allow the operator to

perform the task successfully?

Also discussed in this thesis is the idea of introducing automation to aid the operator perform teleoperation. In particular, in a situation where there is a moving object which must be manipulated, employing some mixture of human control and autonomous control could prove useful. The use of automation in this situation is explored, and the virtual image concept is integrated to help aid the operator. Specifically, this is done by attaching a coordinate system to the moving object, and then displaying the static image of the object as if it were not moving. This proved to be a powerful tool.

The rest of this chapter provides some basic definitions, discusses traditional approaches, and gives an overview of the simulation program.

Chapter two discusses the problems involved with the traditional appoaches to visual displays and identifies three main parameters for graphical displays.

Chapter three covers the experiment performed for the graphical display. The scenario, assumptions, and experimental parameters are presented. The experimental procedures are discussed and the results and conclusions are given.

Chapter four discusses the idea of introducing automation. Supervisory control and working with time delays are discussed. A scenario is posed that was tested in a simulation environment, and the qualitative results are given.

Chapter five discusses the real-world applicability of the ideas presented. The limitations of the simulation, sources of error, and the current state of tracking technology are all presented.

Chapter six provides a summary of the major contributions achieved by this research and recommends areas for further work.

1.1 Definitions

A manipulator is the physical machinery (electromechanical arm and gripper) which is used to perform a task.

The terms teleoperator, robot, and telerobot all refer to a manipulator. A teleop-

erator must be continuously manually controlled by a human who is some distance away. A robot is programmed by a human and later operates autonomously. A telerobot is a combination of a teleoperator and robot, and can behave as either one at any given time. In addition,

A local site is where the human operator operates or monitors the system. A remote site is where the manipulator operates. The distance between these can vary from a few feet to many thousands of miles.

1.2 Need for Robotics in Space

At this time, NASA has three major goals: (1) to monitor changes in the Earth's environment, (2) to establish permanent presence of man in space with the space station, (3) to explore the solar system. Because of the expense and danger of keeping men out in space, and also because there are some tasks (such as picking up large objects) that a human can not physically perform, automation must play a critical role in achieving the goals of NASA. The problem is that a space environment often involves change and uncertainty, whereas traditional automation is preprogrammed and inflexible. We can no longer employ robots that perform only repetitive tasks. In this spirit of flexiblity, some combination of human actions and automation must be found which maximizes the abilities of each. A human is good at things like goal setting, complex decision-making, perceptual sensing, and discerning relevant information. A robot excells at precision, repeatability, data processing, lifting heavy payloads, and working in hazardous areas. Efficiently combining the resources of each would provide the best system. That is the philosophy behind telerobotics and supervisory control. These concepts are discussed in more detail in chapter four.

1.3 Traditional Approaches

1.3.1 Visual Displays

The traditional approach to visual displays for teleoperation has been a live video display. This involves receiving direct feedback from a video camera and displaying it on a monitor. There are two ways of displaying this direct camera feedback: 1) individually, either on separate monitors, or on one monitor with switching between views, or 2) as a stereo image produced by fusing the views from two cameras into one.

Displaying the views individually provides depth information through showing several views of the workplace to the operator. The problem is that all of this information may overwhelm the operator. It is crucial that the operator clearly know from which camera the view is shown and where that camera is in space. Things can get quite confusing with a moving camera, especially since it is impossible to tell if the environment is moving or the camera. The problem compounds when both the environment and the camera are moving, and there is relative motion between the two. This is a real problem when using a two-armed robot, like the Flight Telerobotic Servicer, by attaching the camera to one moving arm and performing the task with the other. In addition, a moving camera has the problem of image jittering while trying to follow the end-effector of a robot [5].

A stereo vision system typically uses two cameras and fuses the views into one view. Two two-dimensional images can be merged into one three-dimensional one. A problem with stereo camera systems is that there is a trade-off between depth resolution and depth distortion. In addition, extra sensor processing is required to combine the multiple views into one. It is still an active area of research though and work is currently being done at JPL to refine this approach[5].

Having multiple, individual camera views is the most popular approach. For instance, for NASA's Flight Telerobotic Servicer Project, there will be no graphics displays or stereo vision; instead four separate cameras will be used to sense the environment[5]. In fact, research at NASA's Marshall Space Flight Center revealed

that two orthogonal camera views were sufficient to perform typical module replacement tasks[14][11]. The two orthogonal views employed were a lateral view showing the task board and manipulator, and an overhead view from a wrist camera showing the face of the task board. A major disadvantage with multiple cameras is that the different views may confuse an operator. It is necessary for the operator to know the location of each camera. If there are several moving cameras, the operator's workload will be increased.

Another area of research is in use of graphic overlays. This involves synchronizing the computer's graphics output with the incoming video camera signal, and then placing both images on the video monitor screen[2].

1.3.2 Control Interface

Other problems involved in aiding the operator concern the operator control interface. This is important because, in teleoperation, it is not enough for the operator to get a view of the environment; he must be able easily manipulate the environment through some interface. Two commonly used hand controllers are three and six degree-of-freedom joysticks. Two three DOF joysticks are typically used (one for translation, one for rotation), and tests indicate, for tasks having high workloads, this type of interface is preferable[14].

The Space Shuttle Remote Manipulator System (RMS), employs such rate control joystick type devices. They provide cartesian control of the manipulator's tip and employ one joystick for x, y, z motions and another for the rotational roll, pitch, and yaw. Rate control devices have now been developed that allow the operator to control all six DOF with one controller. The six DOF controller is effective in moving in multiple axes simultaneously, but it does produce unwanted cross coupling problems[4].

A position control joystick is often prefered when dealing with smaller, more dexterous and quickly moved manipulators because it simulates more closely the natural hand-eye coordinated arm motions of the human operator. There are two main types of position hand controllers: the replica master and the mini-master. The replica

master is a kinematic replica of the slave manipulator, and thus the operator gets a joint for joint coorespondence between his motions and the manipulator's. This is advantageous because it is an intuitively easy control for the operator and no coordinate transformations are required between the controller and the manipulator. The major disadvantage is that the controller must be equal in size to the manipulator, and thus may require a prohibitively large working area. The mini-master is much smaller, but incurs the problem of having computational overhead in requiring transformations between the controller's motion and the manipulator's.

1.4 Simulation Program

The tool utilized in this research to demonstrate the discussed ideas and perform the experiments is called IGRIP. It stands for Interactive Graphics Robot Instructional Program and is marketed by Deneb Robotics. The system runs on a Silicon Graphics 4D 70GT Workstation. IGRIP is a simulation tool which displays interactive 3-D graphics. Some of the program's features include:

- A library of robots containing the geometry and inverse kinematics for each.
- A CAD system which allows the user to build complex objects out of more primative ones. Object relationships and relative and absolute coordinate systems can be defined.
- A mechanism for specifying the kinematics and dynamics for all objects.
- A specialized programming language called Graphic Simulation Language (GSL).

When objects are created and assigned kinematic and dynamic characteristics, they are called devices. When these devices are brought together and their relative interactions are defined, this is referred to as a workcell. The workcell is the basic unit in which the simulation takes places and each experimental trial corresponds to a different workcell. The power of IGRIP results from the ability to program each device within the workcells. By keeping the same devices in the workcell and changing only

the programs, a different workcell can be created. The Graphic Simulation Language includes robot-specific primatives which simplify tasks such as checking for collisions and providing updates of the world.

Chapter 2

Virtual Imagery

When performing pure teleoperation, the operator has the responsibility of controlling the robot. He must manually control the robot iteratively, based on some feedback as to the state of the environment. In most cases, this feedback is not direct; the operator relies on sensor feedback to perform the task. For this reason, it is critical that the operator receives visual feedback that he can quickly interpret.

2.1 Problems with Traditional Approaches

Unfortunately, direct camera images are often inadequate. Here are some of the shortcomings:

1. A live camera view is two-dimensional representation of a three-dimensional world. Even by adding additional cameras or incorporating stereo vision, accurate depth representation is a problem.

2. Every camera has a set focal length, and this fixed length limits the operating range of the camera. This problem can be avoided by having a automatic focus camera, but this results in having to calibrate the distance to the size of the image every time the focusing changes.

3. Video cameras have settings such as zoom, tilt, and pan which the operator must vary based on each situation. The fine-tune adjustments of the view are

not done automatically.

4. Environmental conditions such as poor lighting and constrast may leave the image blurred or unclear.

5. No matter how many cameras there are in the environment, the operator may not be able to see what he needs to see. This is because it is impossible to anticipate beforehand which viewing locations will be valuable, and even if the cameras could be moved, the view might be obscured.

6. Each additional camera adds the overhead of extra sensor processing and overall sensor fusion. The operator load increases as well if he is responsible for keeping track of multiple camera locations and views.

7. Adding cameras reduces the overall robustness of the system because since fixed cameras are placed before the task begins, there will be many situations where task critical views will not be available. This is an important consideration in the unpredictable environment of space[13].

2.2 Virtual Image Example

As a consequence of the shortcomings of direct video images, enhancing visual feedback using a graphics workstation is explored. By using graphics, not only can the user view the environment from any perspective he wants, but the environment can actually be changed to suit his needs. For instance, if the operator needed to see the back side of an object, the object could be made transparent. In addition, the images are sharp and clear since all the colors and contrasts are selected by the operator. By using additional strategies such as visually monitoring a part of an object (the one most convenient), and then transforming it into a graphical view for all other parts, accurate feedback can be maintained with sensor hardware kept to a minimum. To illustrate this concept, an example scenario is presented.

2.2.1 Scenario

An operator is teleoperating a manipulator to fit a peg in a hole on a table. The operator receives direct video feedback from one of the video cameras, but unfortunately it only reveals the edge of the table. The hole on the table is completely out of the sight of this camera and is possibly out of viewing range altogether. In a situation such as this, the virtual image concept can be fully exploited. If the relative location of the part of the table that can be seen is known, the rest of the table can be derived. It is a matter of matching it up to a CAD model and performing the necessary coordinate transformations. The location of the hole is determined and is displayed to the operator. The operator can now perform the task by viewing the graphics workstation instead of the direct video image.

2.3 Graphical Display Categories

Given that graphical views are more appealing to operators, the question then becomes: what constitutes a good graphical view? In order to answer this question, parameters involved with visual displays are first identified and discussed. There are three main parameter categories: camera views, graphical display content, and alphanumeric display content.

A. Camera Views. Though there are an infinite number of views that can be shown from a camera, there are three main types. It is important to note that each of these views come directly from a camera or can be derived from other camera views.

   1. World Camera View. This view appears as if it comes from a static camera in a fixed position in the world. It allows the operator to get a global perspective on the environment. This camera's position can be moved to anywhere the operator wants.

   2. Wrist Camera View. This view appears as if it comes from a camera which is attached to the wrist of the robot. The major advantage with this is

that as the robot's tool tip translates and rotates, the camera moves with it, and it always appears that the tool tip is orthogonal to the operator. A disadvantage is that it can be confusing to decide from this view how to control the hand, since the world seems to move in a direction opposite to the control action.

3. Object View. This view appears as it is coming from a camera attached to the object. It proves especially useful with avoiding collisions with specific objects, and in tasks where moving to objects are necessary. When the object is stationary, this view is essentially the same as the world camera view. In the case of a moving object, the view appears as if the camera were moving with the object, and thus the object appears static. The motion of the object gets abstracted out and transferred into the environment as if it were the environment that was moving.

B. Graphical Display Content. There are many different ways of graphically representing raw camera data.

1. Object Modelling

   a. Solids. Objects are shown as solids with opaque faces unless there is a specific reason not to because solids are the clearest to see. In addition, modelling objects as solids most closely simulates the real-world. In figure 2-1, the table is modelled as a solid.

   b. Transparent. Some objects are displayed as transparent because it necessary to see through them. This mode is valuable because though it allows the operator to see behind an object, it yet maintains the image of the object so that the operator is still aware of its location. It is important to give the operator feedback most closely resembling the real environment because oversimplification can lead to uncertainty and unwanted collisions. The box is modelled as a transparent object in figure 2-1.

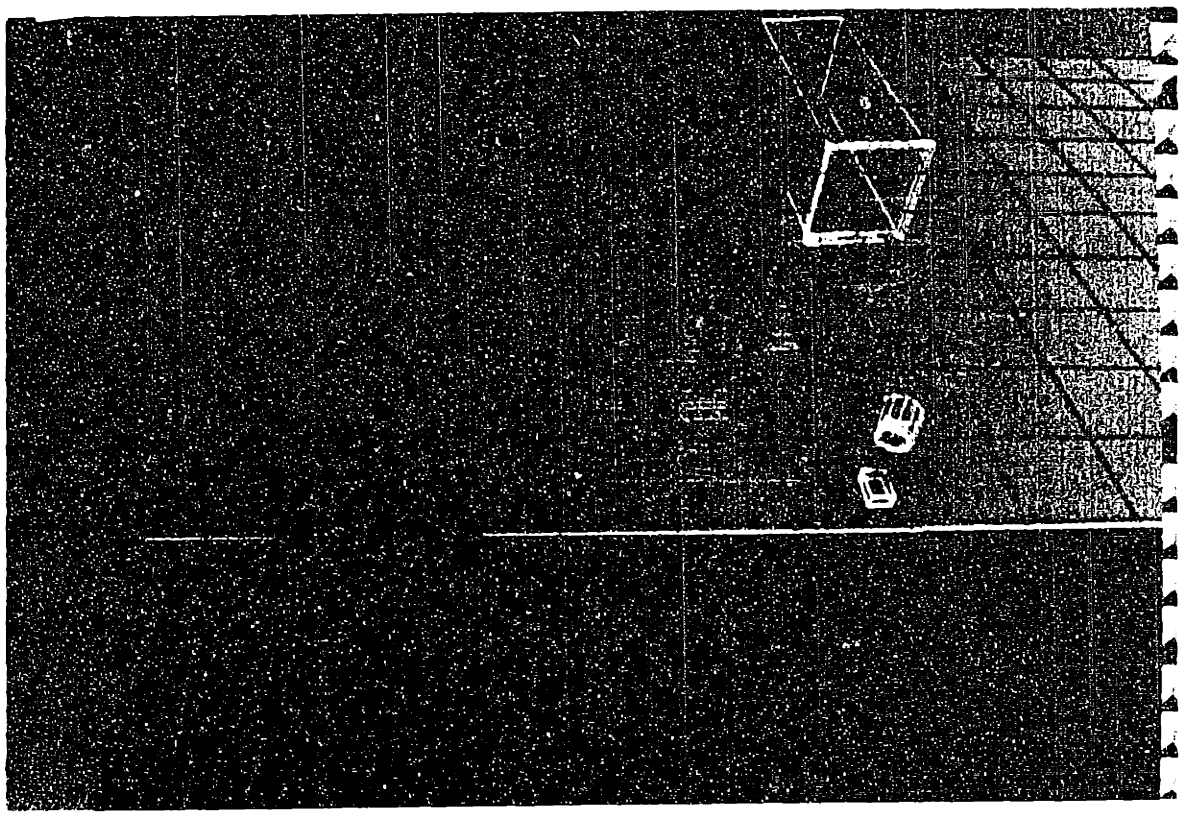

Figure 2-1: Wireframe robot, transparent box, and solid table.

c. Wireframe. This display style outlines the edges of the objects and displays the rest as invisible. The operator can get a better sense of depth perception than in the transparent case. The robot is modelled as a wireframe object in figure 2-1.

d. Invisible. Making an occluding object invisible can help make the view behind it clearer (ie by removing obstructions), but the operator does not get a realistic sense of the environment. For example, if a part is made invisible, the operator will not know it is there and may collide with it. Invisibility should be used only with parts that are completely extraneous to the operation.

2. Graphical Aides

a. Graphics Window. It is a window which can graphically depict relative transformations between the robot and an object. This can be done like many Flight Simulator Programs by showing a horizon in the center of the screen and having a moving background.

b. Changing Color. Color can be used as an effective guide for an operation. For example, if a collision is about to occur, the colors of the colliding objects can be changed to warn the operator.

c. Changing Size. As the robot moves close to objects, the view can be enlarged; as the robot moves away, a more global view can be shown.

C. Alphanumeric Display Content. There are many things that are most appropriately displayed to an operator quantitatively. Some of them are:

1. Distance to Target. For example, the straight line distance from the robot's tool tip to a specified target location may be useful.

2. Distance to Collision. This is the closest straight line distance from the robot to an object.

3. Absolute Position. The translational and rotational coordinates in the X, Y, and Z directions are displayed from some fixed, global reference frame.

4. Relative Position. This message provides the relative translational and rotational coordinates in the X, Y, and Z directions between two entities in the environment.

Chapter 3

Simulation and Experimentation Employing Graphic Displays

In order to study how best to improve operator feedback, a simulation was developed. Because this question is task dependant, the simulation has been limited to one particular task: putting a peg in a hole on a moving object. Though there are many other tasks that could have been chosen, six degree of freedom mating is an especially representative task because it is difficult yet practical. It is generic enough to allow conclusions to be drawn about other specific tasks. Another important consideration is the validity of the simulation. This is critical because if the simulation's results do not apply in the real world, all of the experimentation and results are meaningless. Thus, great care was taken to explictly state all of the assumptions made when developing the simulation and ensuring that these assumptions are realistic.

In this chapter, a scenario is described that was modelled in IGRIP. The relevant assumptions are listed. The parameters identified as affecting the operator's ability to perform the task are provided along with descriptions of each. The experiement itself is described, and the results are listed. The experimental conclusions are then presented.

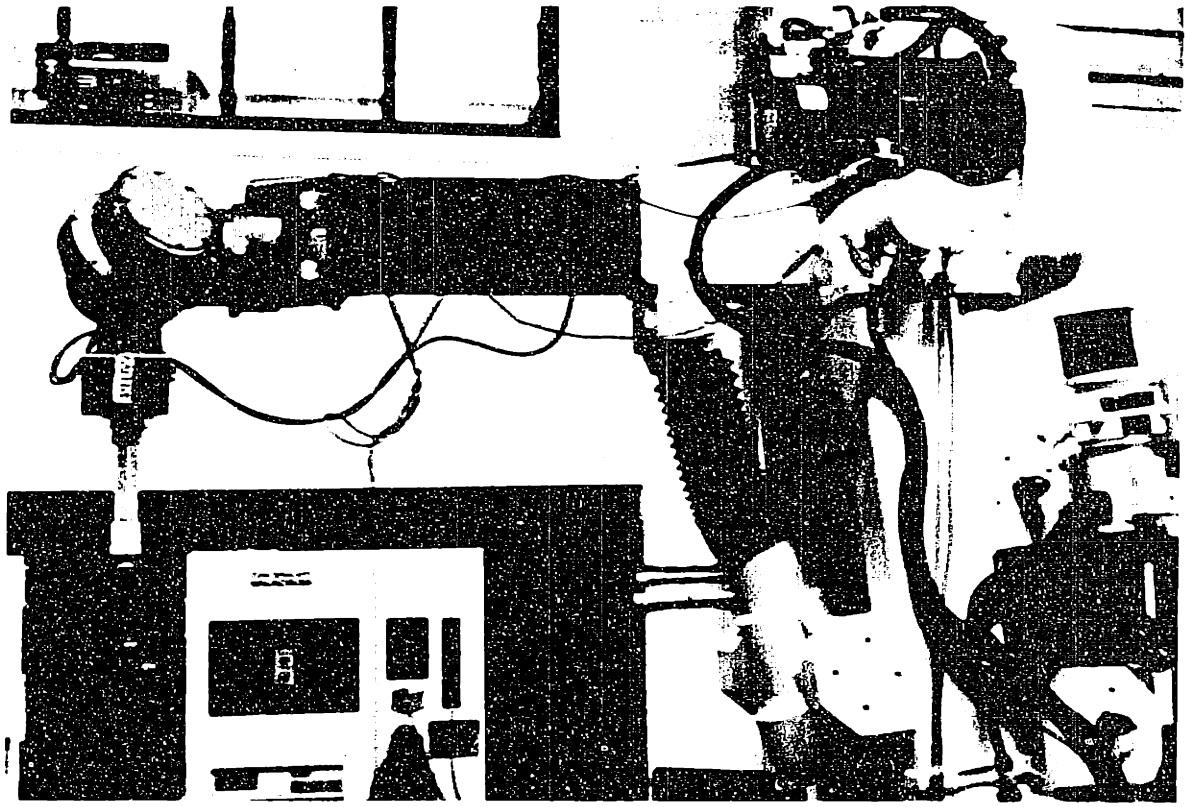

Figure 3-1: Side View of the Cincinnati Milacron T3-786 Robot.

3.1 Scenario

An object is floating slowly through space. It is rotating and translating. There is a hole on one of the faces of the object. There exists a Cincinnati Milacron, T3-786 robot with a peg in its gripper which the operator can move around via some operator interface. The task is for the operator to maneuver the peg into the hole of the object, within some tolerance, without any collisions. Figures 3-1 and 3-2 show the actual T3-786 robot as taken at NASA's Goddard Flight Center in Greenbelt, Maryland.

Note: The above scenario could have been posed using any generic robot. The T3-786 was selected and the accompanying pictures included, to provide the reader an example of a particular robot that could be utilized.

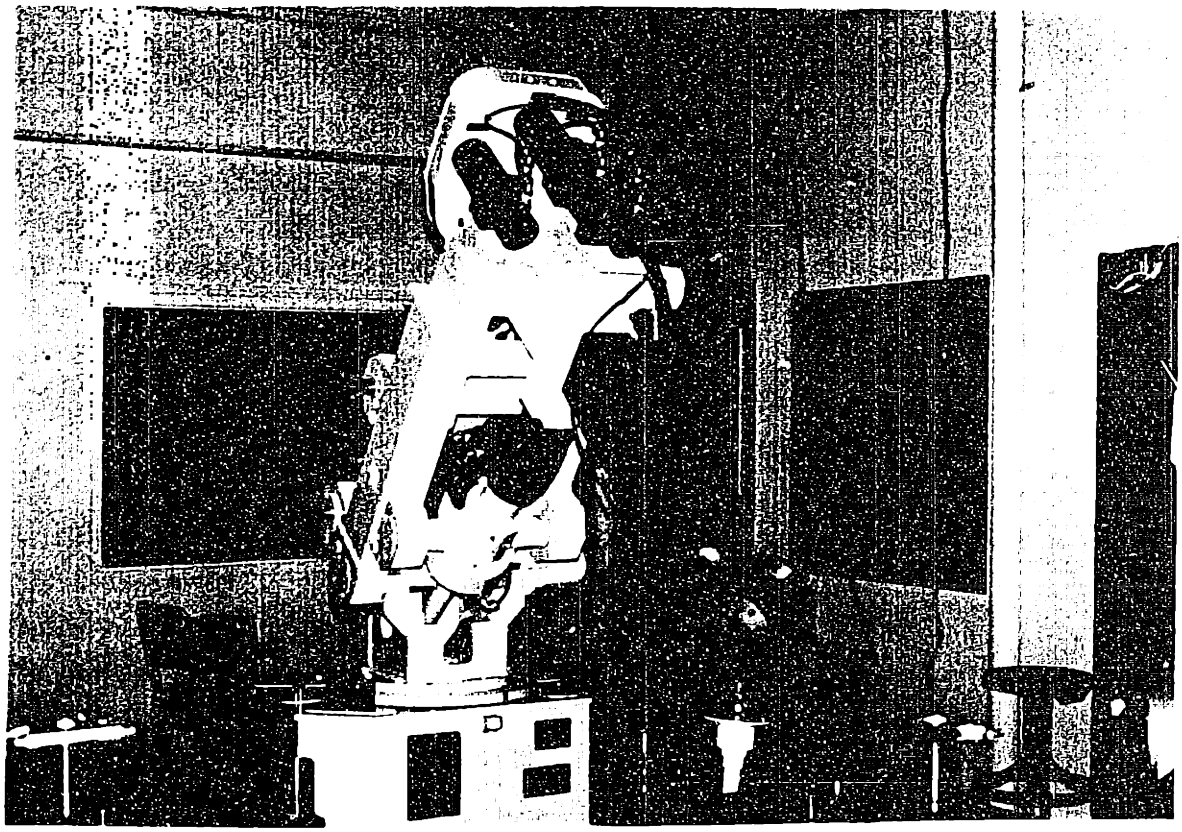

Figure 3-2: T3-786 Robot bent downward.

3.2 Assumptions

A number of assumptions were made to derive a simulation from the scenario described above:

1. The robot base coordinate system is fixed.

2. There is a precise CAD model of every object in the environment. This serves as the basis for the model-based object recognition algorithms.

3. The coordinate transformations from the camera to the robot's tool tip and from the tool tip to the robot's base are known.

4. The robot can output its joint angles in real-time, and thus the position of the whole robot with respect to its base coordinate system is known.

5. The object has already been sighted and is within the view of the world camera at the outset.

6. Transmission time delays are negligible - ie. the operator is working from a nearby space station control room. With round-trip delays, for example, of three seconds for communications from the earth to the moon, forward simulation could be used. For a complete discussion of this, refer to[6][10].

For the vision system, assume:

1. A digitizer for each video camera which continuously processes frames.

2. A low-level vision module that can extract feature points from the digitized image. These feature points are then matched to the CAD model of the object and the locations of these points are derived with respect to the local coordinate system of the object. This matching process can be done in a man-assisted mode where the operator interactively assists by indicating which features in the image (e.g. corners, edges) correspond to those in the CAD model. It is assumed here that the feature matching is done automatically by the system. Two popular matching strategies are: tree searching and clustering[12].

3. A higher-level vision module which, given the feature points, finds the location and orientation of the object in the camera coordinate system (pose estimation).

4. Another module transforms the six-dimensional position of the object into the robot base coordinate system. This is possible because, since the past camera trajectory is known, the position of the camera in the robot base coordinate system when the frame was grabbed is known.

5. The vision system can do all of this fast enough so that the information is not obsolete by the time it is processed. This assumption requires some combination of an efficient vision system and a slowly changing environment.

Because the positions of all of the objects seen by the camera are known with respect to the robot base coordinate system and also because all parts of the robot are located with respect to this same system, the camera picture can now be translated onto a graphics model (simulation). Because this model gets periodically updated by

the camera, it shows essentially the same view that the camera sees. The practical difference is that the objects in the simulation come from a CAD model, not the real world.

3.3 Experimental Parameters

The scenario described above was modelled in a simulation environment. An experiment was performed to determine which factors best aided an operator in performing a task. Four major display/control categories were tested in the simulation.

A. Camera Views (VIEW). Three were used in the experiment.

1. World Camera View (woc). One camera was placed in a fixed location in the environment. This camera view was available for all the trials. For this task, it was useful in performing gross motions and viewing relative transformations. The problem with this view was that it was not effective for performing fine motions. This view is depicted in figure 3-3.
2. Wrist Camera View (wc). The one problem is that there always is an offset between the tool tip and the camera, sometimes causing the camera view to be obscured. This view is shown in figure 3-4. The robot was displayed as invisible to achieve a better view of the peg.
3. Object View (ob). Because of the infeasibility of actually placing cameras in space that move with objects, this view, unlike the previous two, was merely a transformation of the wrist camera view. The advantage with using this view was that the problem gets transformed from moving a peg into a moving hole to that of moving a peg into a static hole. One drawback was that the motion of the object became abstracted into the motion of the robot (ie. the robot continuously moved instead of the object). For this task, because the goal was to move the robot into an object's hole, there were two appropriate object views:

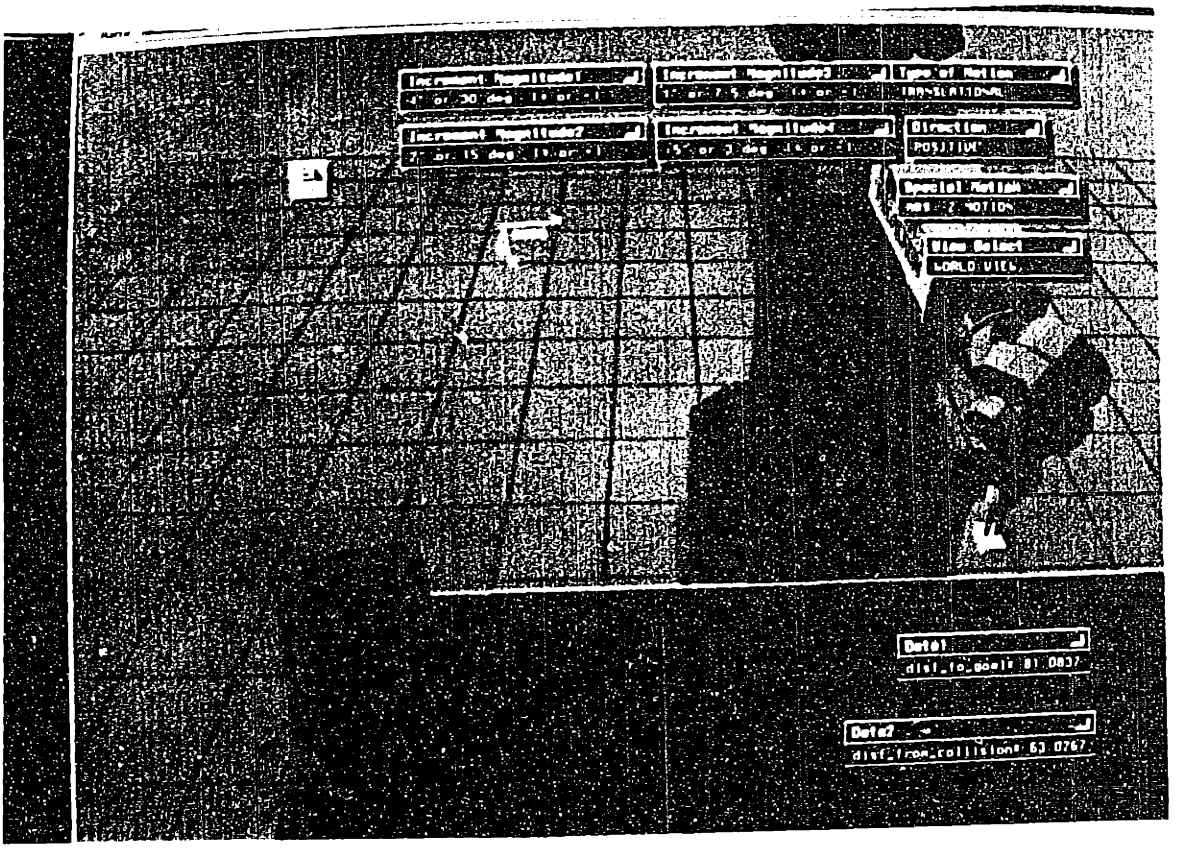

Figure 3-3: World Camera View

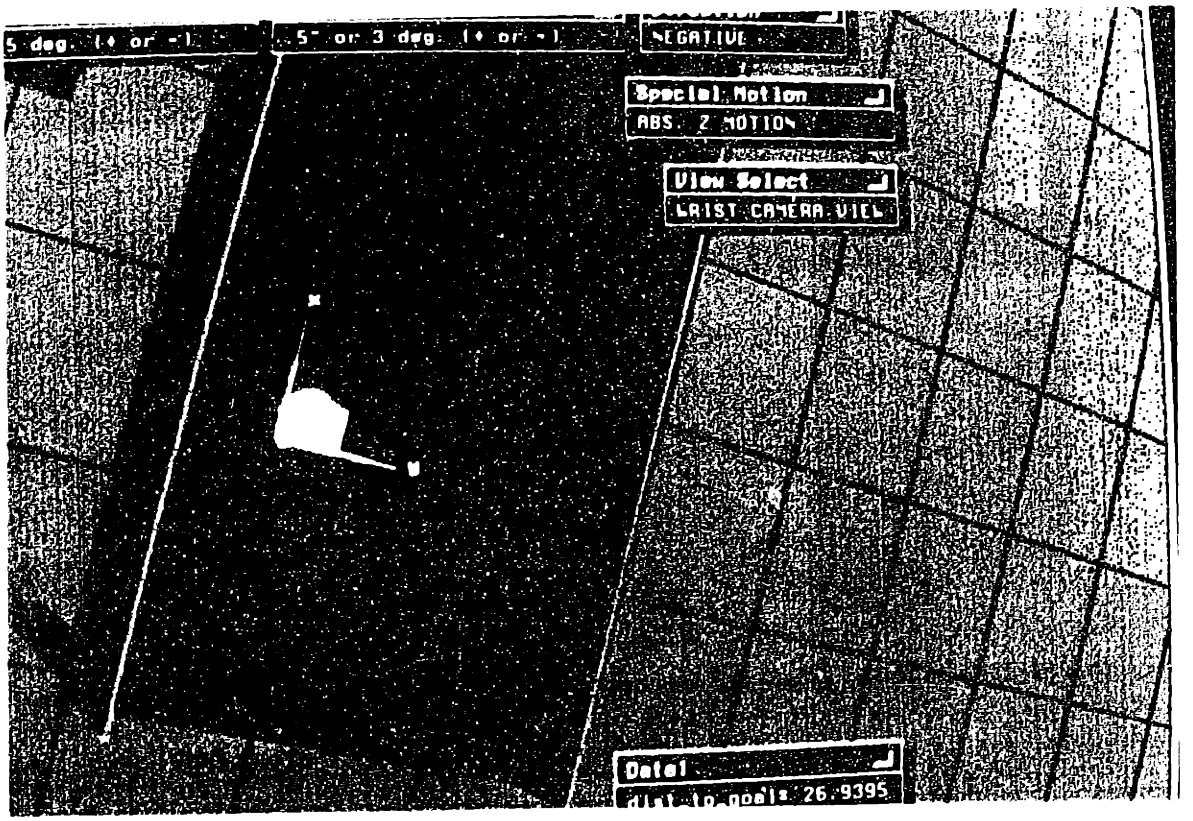

Figure 3-4: Wrist Camera View

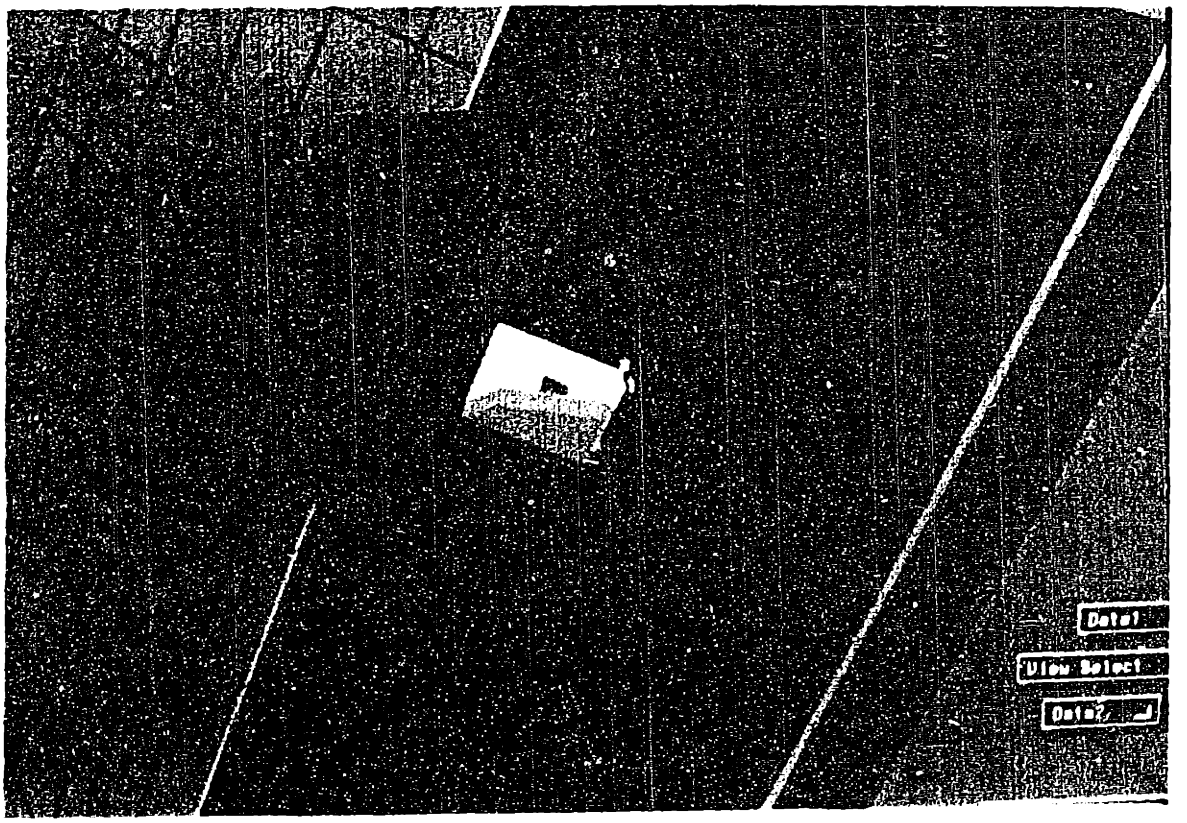

Figure 3-5: Object View 1: Looking down onto the box.

a. Object View 1 (ob1). This view involved looking down onto the target hole. This view is shown in figure 3-5. The peg was modelled as a solid, and the robot was modelled as transparent.

b. Object View 2 (ob2). This view involved looking up from the target hole. This view is shown in figure 3-6. Again, the peg was displayed as a solid and the robot as transparent. A solid object could be seen behind a transparent one, but a second transparent object could not. Figure 3-7 shows the same view, but became enlarged as the peg got closer to the hole.

B. Graphical Display Content (GDC). There were two main modes: normal and graphical.

1. Normal (-). In this mode, all of the objects were initially modelled as solids. Then, based on the viewing situation, the objects were changed to either transparent, wireframe, or invisible. A simple algorithm determined

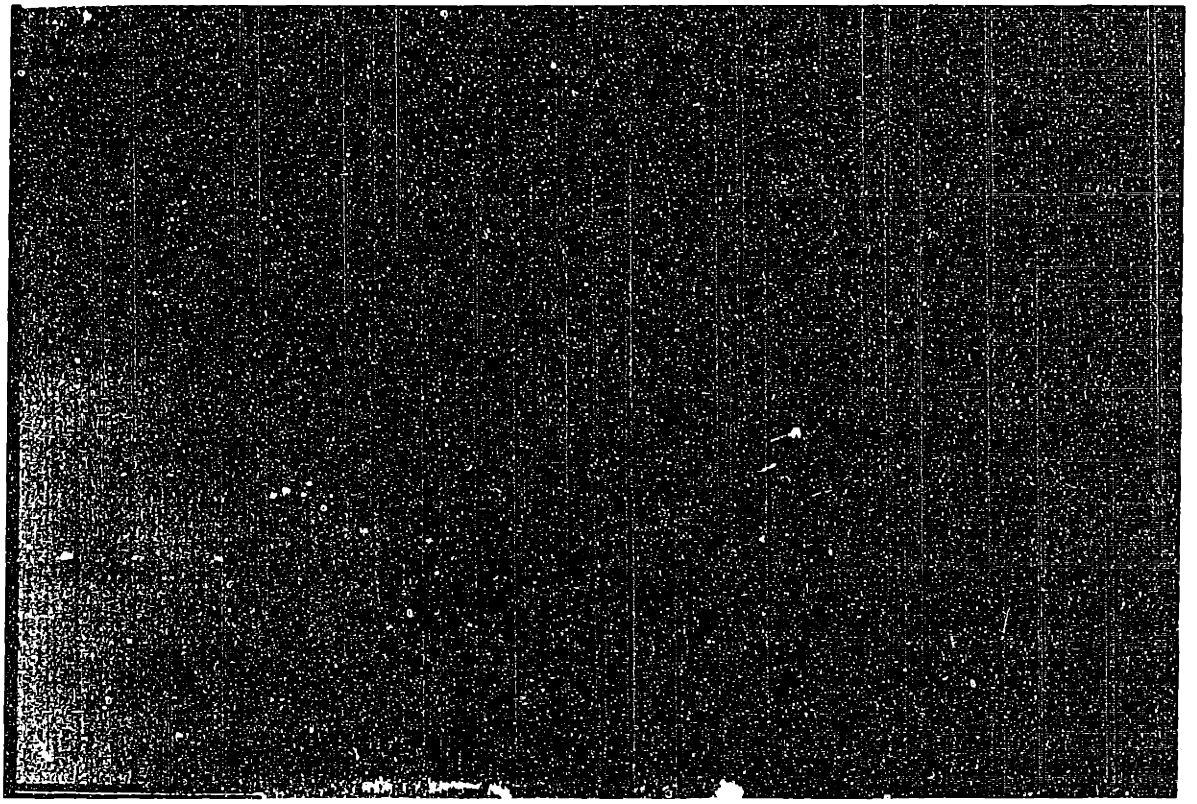

Figure 3-6: Object View 2: Looking up through the box.

Figure 3-7: Object View 2: Enlarged view through the box.

the size of the view based on how close the object was to the camera. If a collision occurred, the robot would turn red and move away from the object by five inches.

2. Graphical Mode (gra). This mode included all the features of the normal mode plus two additional ones. First, a graphics window was added. It depicted the relative yaw/pitch distances between the peg and hole by moving a background around a fixed horizon. For this window, a rotational difference in the X direction caused the background to rotate (in the plane of the screen), and a rotational difference in the Y direction caused the background to translate up and down. The other feature was that when the peg's translations are lined up within some tolerence with the hole, the peg turned red.

C. Numeric Display Content (NDC). There were two display modes here: normal and additional.

1. Normal (–). The operator received continuous feedback of two parameters: distance to target and distance to collision.

2. Additional Information (ai). In this mode, not only were the distance to target and distance to collision figures given, the relative position between the end of the peg and the center of the hole were also given.

D. Interface. The operator interface is of critical concern in a teleoperated task because it dictates the means by which the operator controls the robot. Not only must the operator receive proper visual feedback, he must also be able to easily maneuver the robot through the environment. In this simulation, all the inverse kinematics of the robot were predefined, so the operator could control the tip of the robot in cartesian space. For the operator's convenience, there were two coordinate systems in which the robot could be moved: absolute and relative. When the static, world camera was used, the robot moved in an absolute coordinate system that the operator could see on the screen. This

coordinate system did not change based on the robot's motion. When the wrist camera view (or either object view) was used, the operator moved in the relative coordinate system. This coordinate system was relative to the tool tip, and thus, when the robot rotated, the coordinate system also rotated. This relative motion was intuitively easy to control when employing the wrist camera view. The axes are also displayed on the screen so the operator always knew where the x and y axes were pointing. The positive z axis was always straight down the tool. Five operator interfaces were used:

1. Mouse (mouse). With this interface, the operator clicked one of the three buttons on a mouse which translated/rotated the robot's tool tip in the specified direction. The left, middle, and right buttons represented the X, Y, and Z directions respectively. The distance that the robot moved depended on where on the screen the operator clicked. There were explict blocks which had fixed distance values. The parameters such as positive/negative, translation/rotation, and world camera/wrist camera were set by clicking toggle buttons on the screen. This interface, along with the normal display content windows, is displayed in figure3-8.

2. Mouse with aid (auto). The display for this interface was the Additional Information Written Display Content. The operator moved the tool tip by simply clicking the mouse button on one of the distance readout windows, and the robot moved towards the target in that direction. The distance it traveled depended on which of the three mouse buttons was pressed. In other words, if the tool tip was 50 inches away from the hole in the X direction, if the operator clicked the left button in this window, the robot moved to 45 inches away. If, for some reason, the operator wanted to move away from the object in some direction, there was a toggle switch which allowed him to do so. This interface is displayed in figure 3-9.

3. Graphical Window Interface (gwi). This interface was employed only with gra, and served to supplement the standard interface used in the trial.

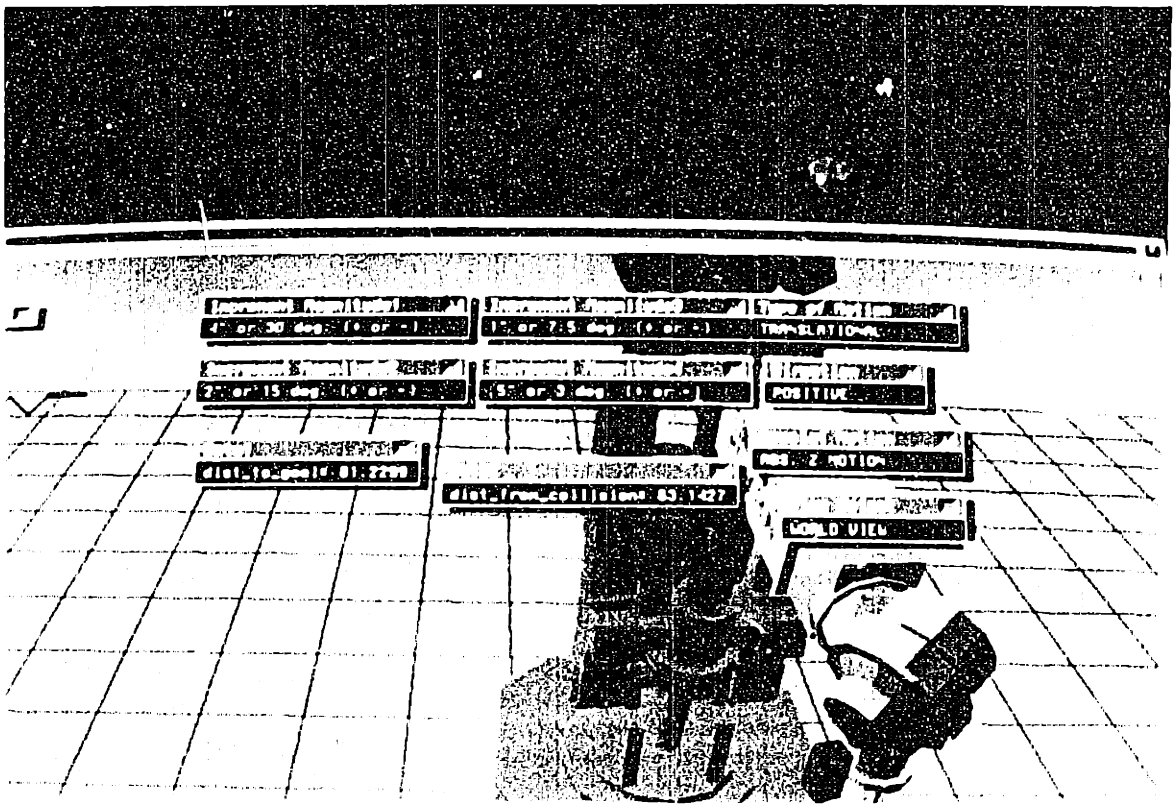

Figure 3-8: Buttons for Mouse Control Interface

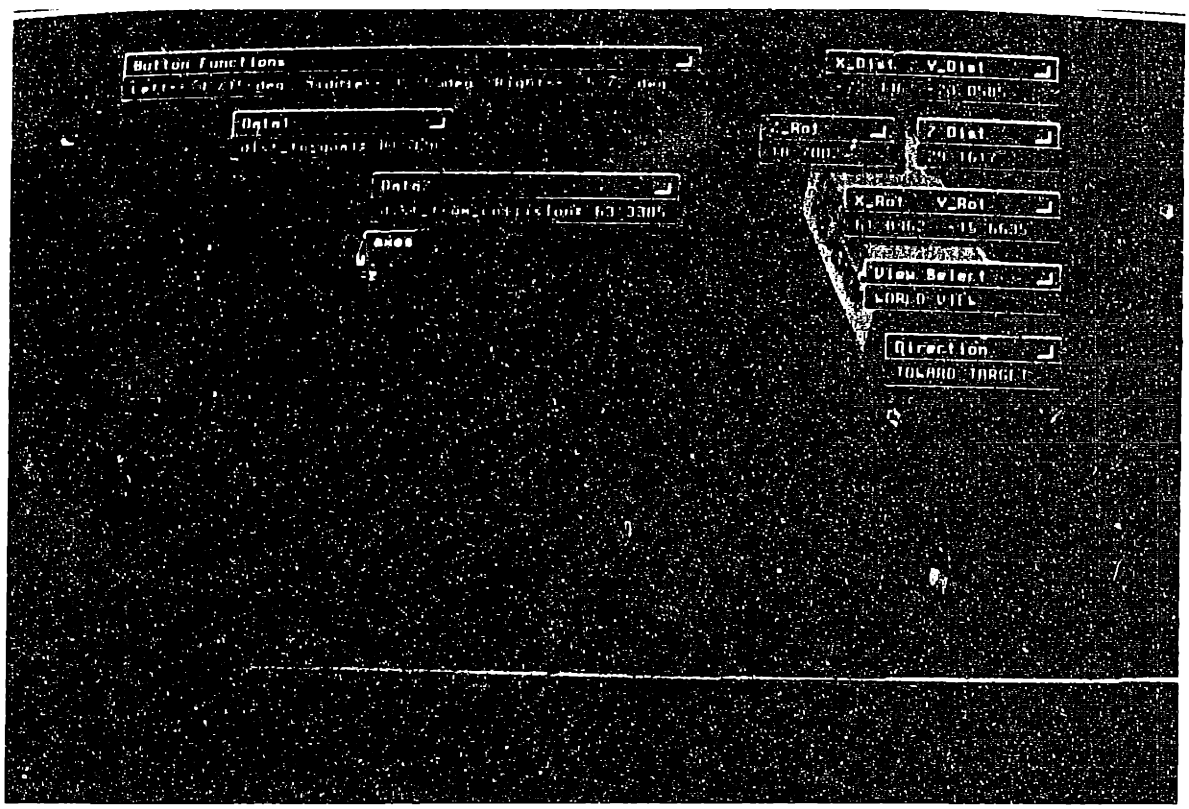

Figure 3-9: Buttons for Automatic Mouse Control Interface

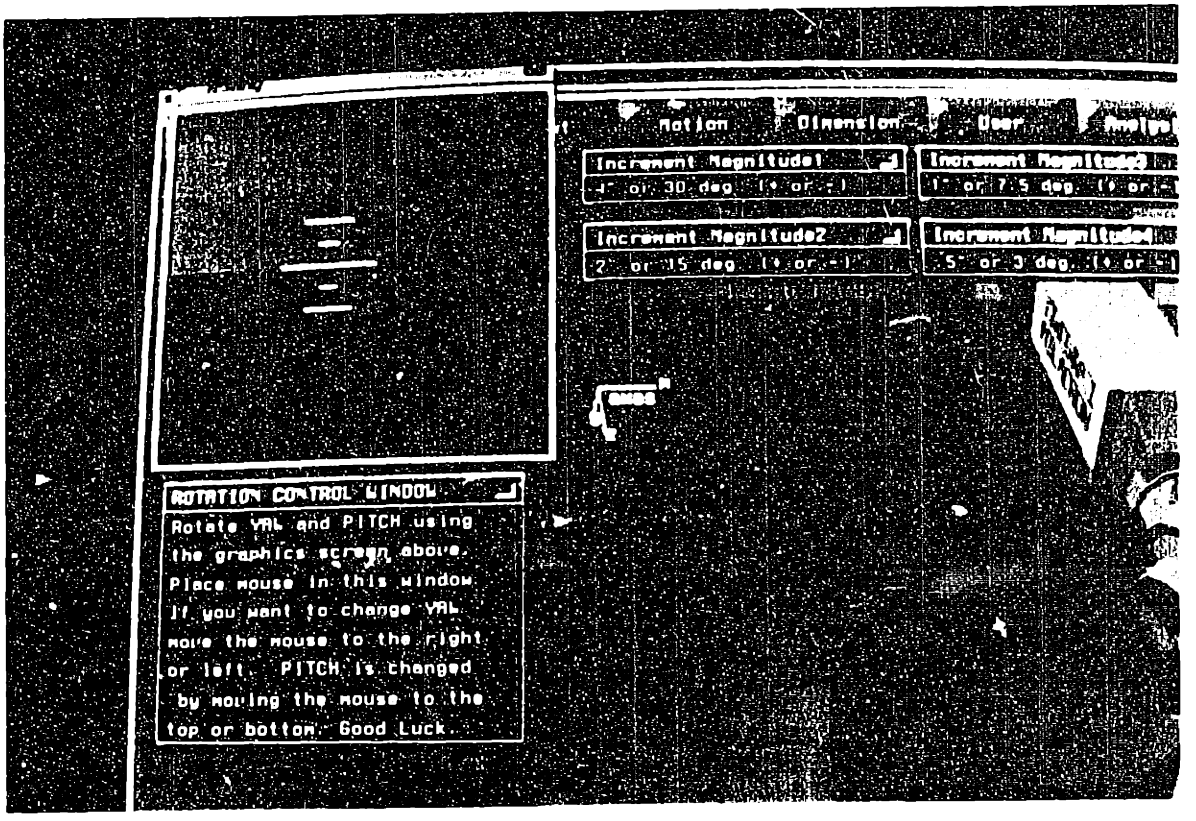

Figure 3-10: Graphical and Control Interface Window: The X and Y rotations are aligned.

Basically, this interface created an input window which cooresponded to the graphic window described above. The interface was simple. When the operator wanted to correct the Y rotation, for example, he moved the mouse to the bottom of the input window. When he wanted to change the X rotation, he moved to either the right or left of the input window cooresponding to the direction he wished to rotate. The graphical window and its cooresponding interface are displayed in figures 3-10,3-11,3-12.

4. Six degree of freedom Joystick. A six DOF joystick was connected to the serial port of the Silicon Graphics computer. After a C program read the joystick data, it passed the values onto the IGRIP program via sockets. The joystick continuously emitted values to the port (whether they are read or not), so the question of proper sampling was critical. There was always a time lag between when the operator moved the joystick, and when the IGRIP program received the data and the robot completed its motion.

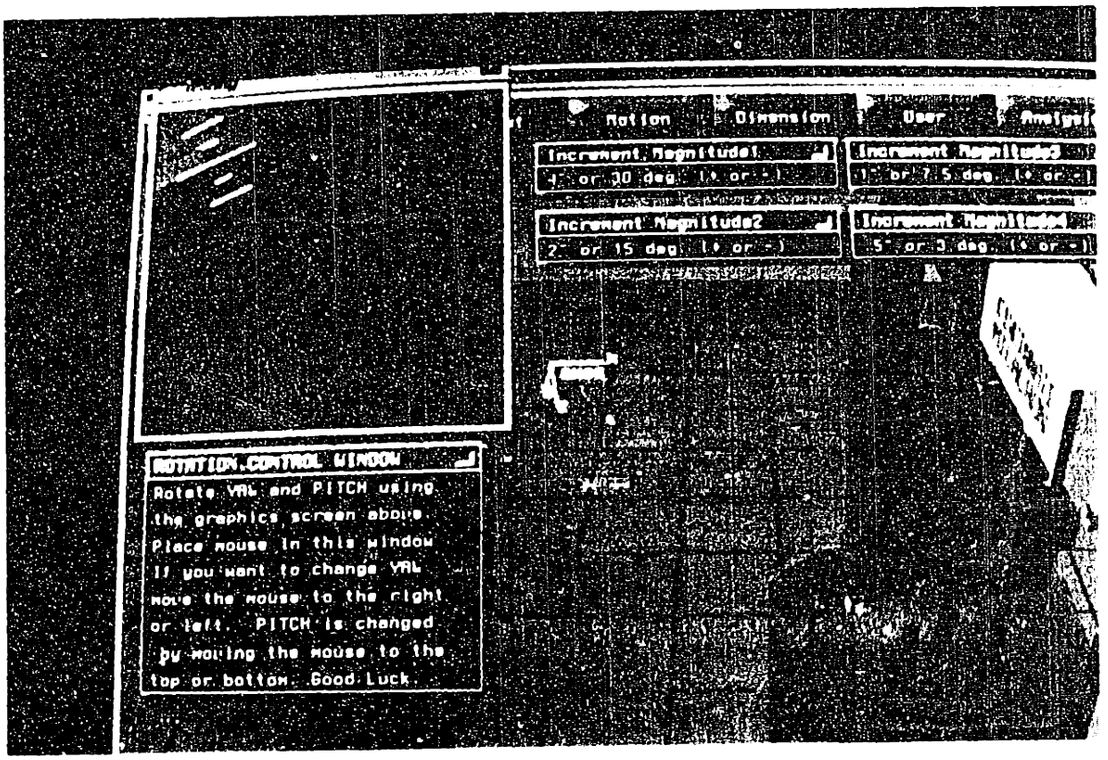

Figure 3-11: Graphical and Control Interface Window: The X rotation is not aligned.

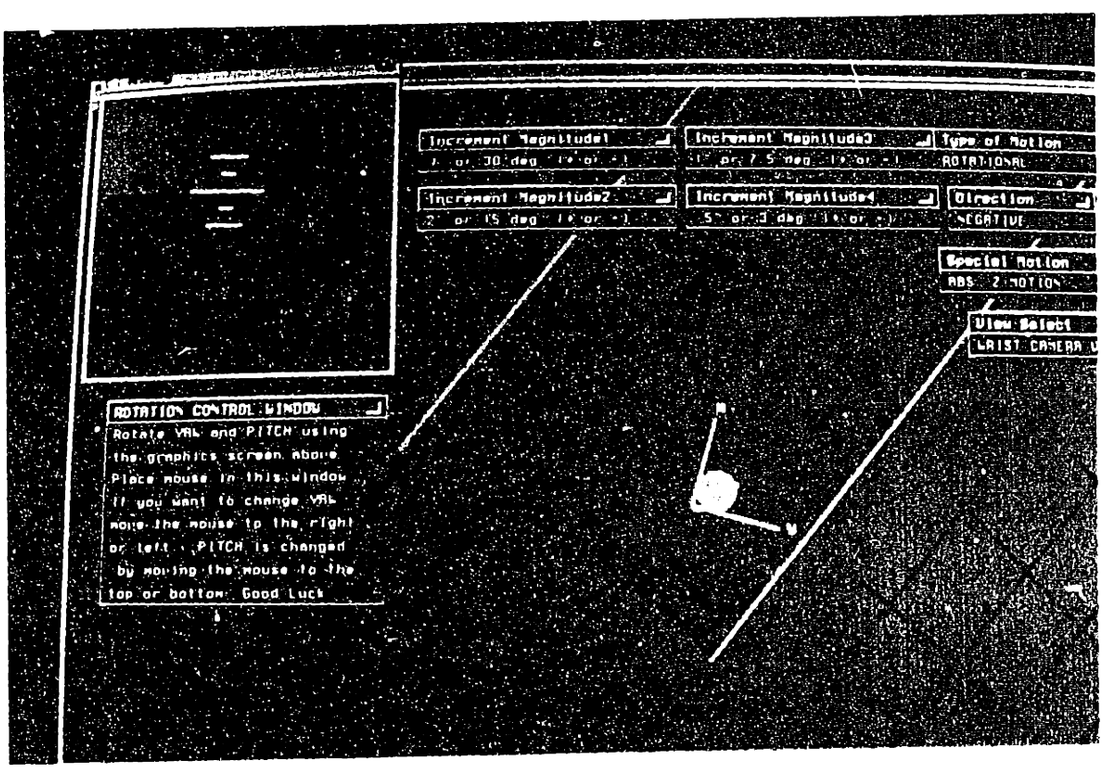

Figure 3-12: Graphical and Control Interface Window: The Y rotation is not aligned.

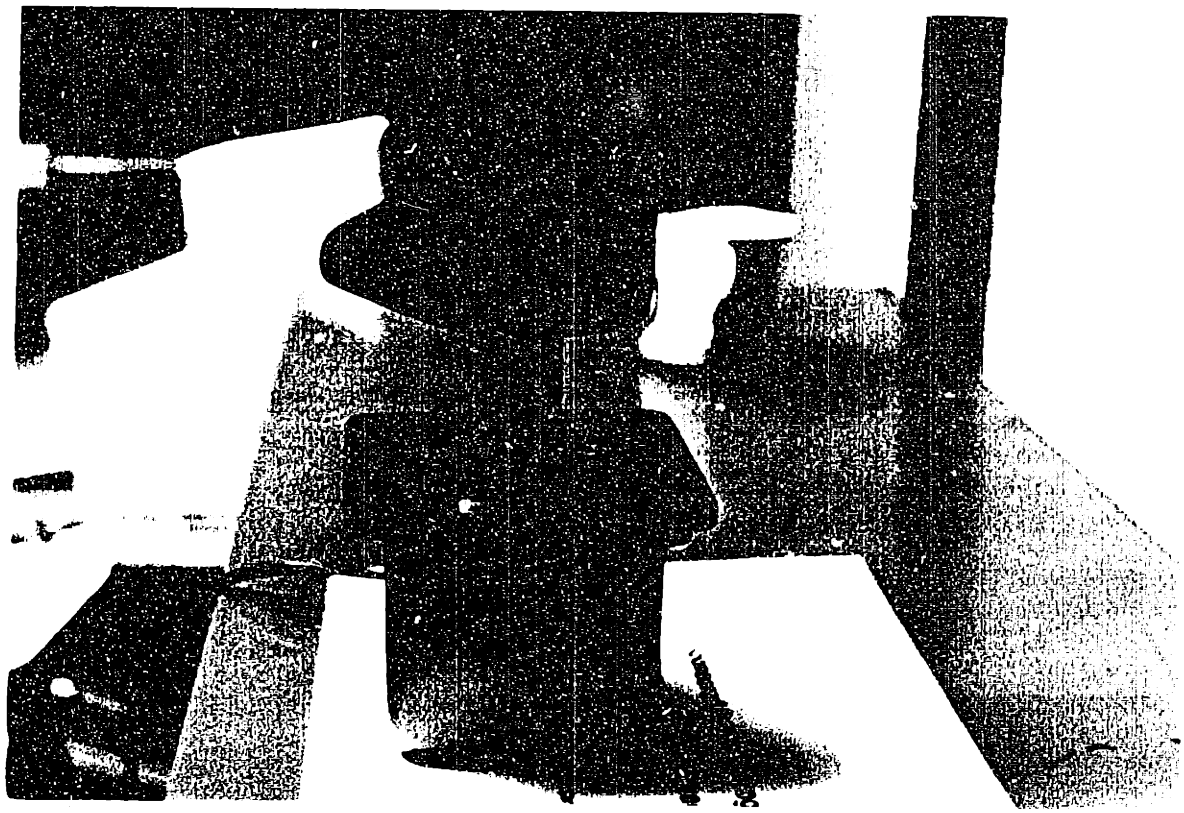

Figure 3-13: Six Degree of Freedom Joystick

Thus, when not sampled carefully, there was a queue of joystick commands that got built-up, and the lag time increased on the order of ten to twenty seconds. For this reason, the current polling scheme was to read a value, make the move, and then keep reading values until (0,0,0,0,0,0) had been detected. The next input was then processed. The joystick was run under two control schemes: variable positional and fixed positional. The joystick is shown in figure 3-13

a. Variable Positional (joy_var). When the operator exerted a force on the joystick, the robot moved a distance proportional to that force in each direction. The motion was coupled.

b. Fixed Positional (joy_fix). When the operator exerted a force on the joystick, the robot moved in the direction in which the maximum force was applied. The distance travelled was determined by the operator based on the location of the mouse. The motion is entirely de-coupled and the robot traveled fixed distances.

TRIAL	PEG	VIEW	GDC	NDC	INTERFACE
1	–	wc	–	–	mouse
2	–	ob1	–	–	mouse
3	–	ob2	–	–	mouse
4	–	wc	gra	–	mouse
5	–	wc	gra	–	mouse+gwi
6	–	ob	gra	–	mouse+gwi
7	–	wc	–	ai	mouse
8	–	wc	gra	ai	mouse+gwi
9	–	wc	–	–	auto
10	–	ob	–	–	auto
11	–	wc	–	–	joy_var
12	–	wc	–	–	joy_fix
13	–	wc	gra	–	joy_fix+gwi
14	bl	wc	–	–	mouse
15	bl	ob1	–	–	mouse
16	bl	ob2	–	–	mouse
17	bl	wc	gra	ai	mouse+gwi
18	bl	ob	gra	ai	mouse+gwi

Table 3.1: List of Trials

3.4 The Experiment

Four operator subjects were selected to perform the experiments. None had prior experience with the system. Two of the operators had extensive experience in the area of telerobotics. All the operators were trained on the system for about 8 hours through a standard set of trials before performing the actual experiments. The training period was long and rigidly monitored to ensure that each operator got equally proficient with each trial. The list of trials are given in table 3.1.

The abbreviations correspond to those listed with the descriptions of experimental parameters earlier in the paper. The second column (labeled PEG) refers to the type of peg used in the trial. The default was a small rectangular peg, whereas bl stands for a large peg which serves to block the view of the wrist camera. This can be seen in figure 3-14. The abbreviation ob refers to the object view (either ob1 or ob2) that the operator liked better. The full list of abbreviations can be found in table 3.2.

Abbreviation	What it Represents
bl	large peg
wc	wrist camera
ob1	object view 1 (looking down)
ob2	object view 2 (looking up)
ob	object view (1 or 2) that operator perferred
gra	graphical mode (includes graphical window)
ai	additional alphanumeric information
gwi	graphical window interface
auto	mouse with aid
joy_fix	six dof joystick with fixed distance movement
joy_var	six dof joystick with distance proportional to force

Table 3.2: List of Abbreviations

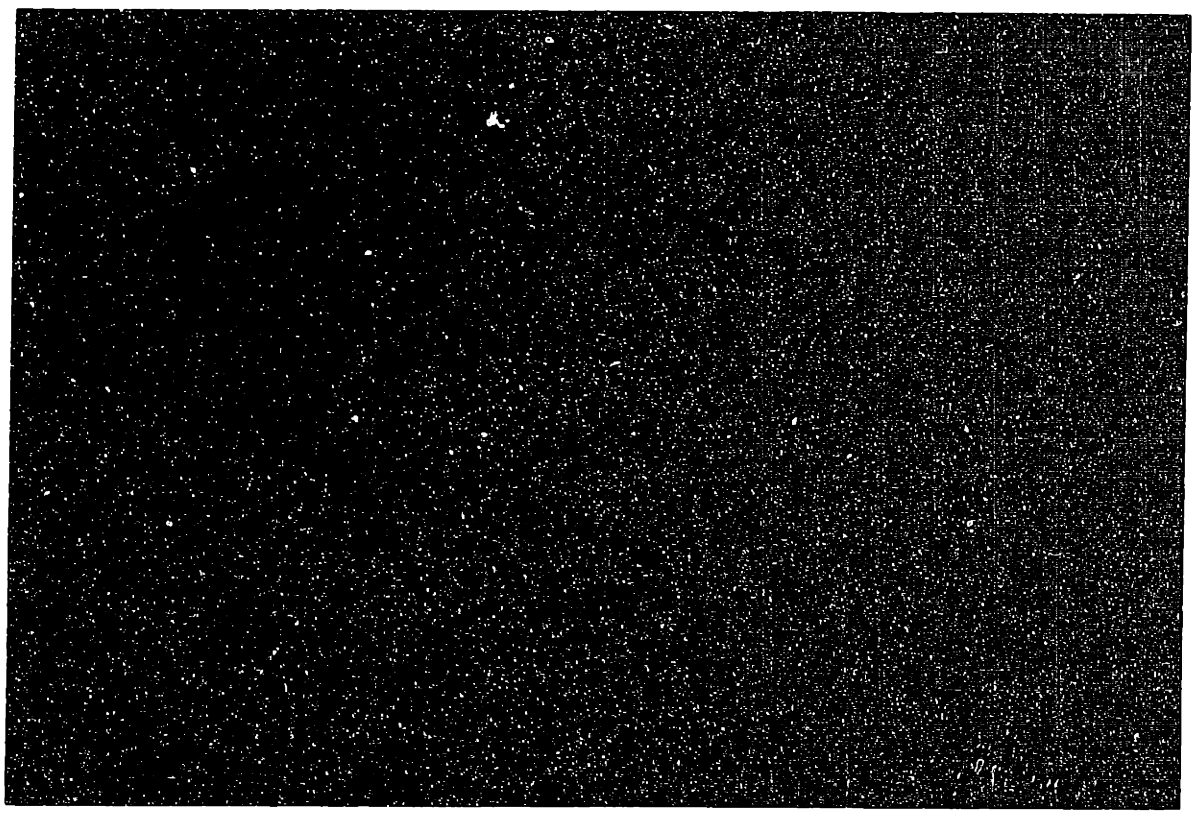

Figure 3-14: World Camera View of Large Peg

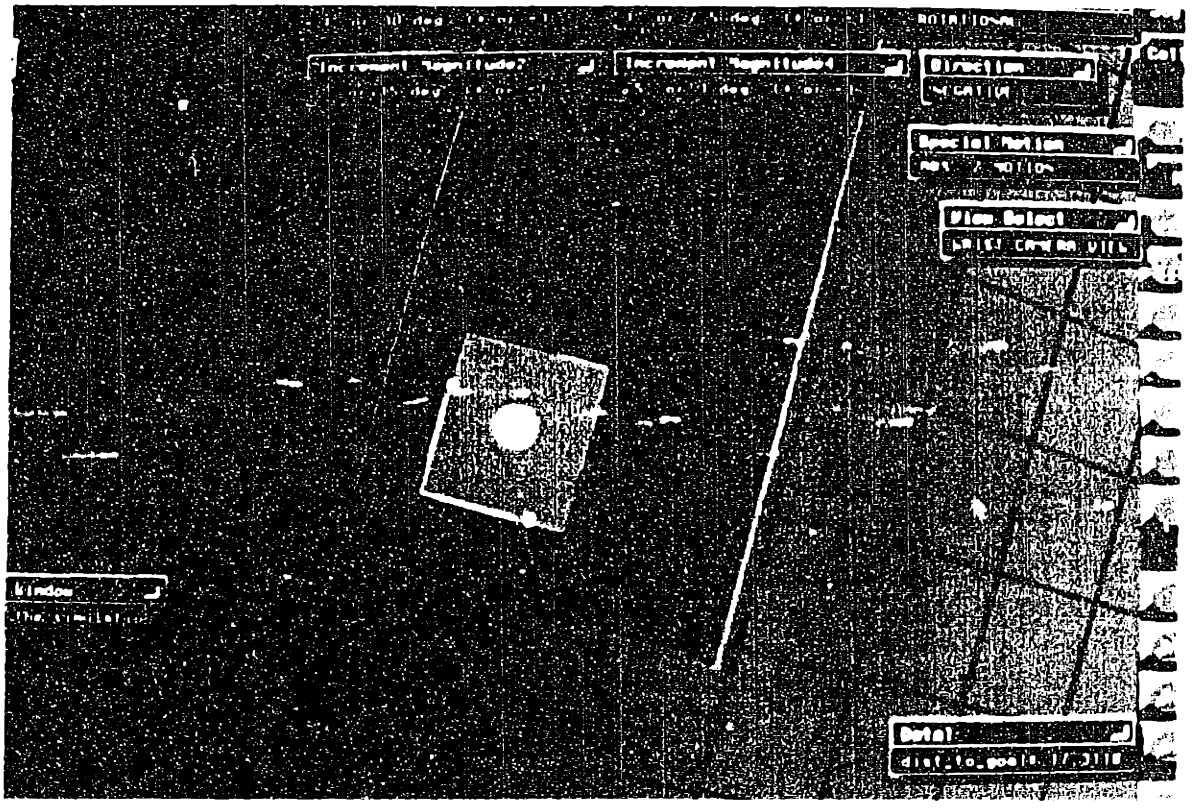

Figure 3-15: The large peg blocks the view of the Wrist Camera

Note: figures 3-15, 3-16, and 3-17 display the different views for trials 14, 15, and 16 respectively.

Each trial was run four times under different object speeds and task tolerences. The four runs for each trail are listed in table 3.3.

The object speed corresponds to how quickly the moving object translated and rotated. The task tolerance represents the level of difficulty of the task. A low task tolerance required that the peg be within .75" of the center of the hole, while a high task tolerence required the peg to come within 1.25". This task tolerance parameter

Run #	Object Speed	Task Tolerance
1	slow	low
2	slow	high
3	fast	low
4	fast	high

Table 3.3: The Four Trial Runs

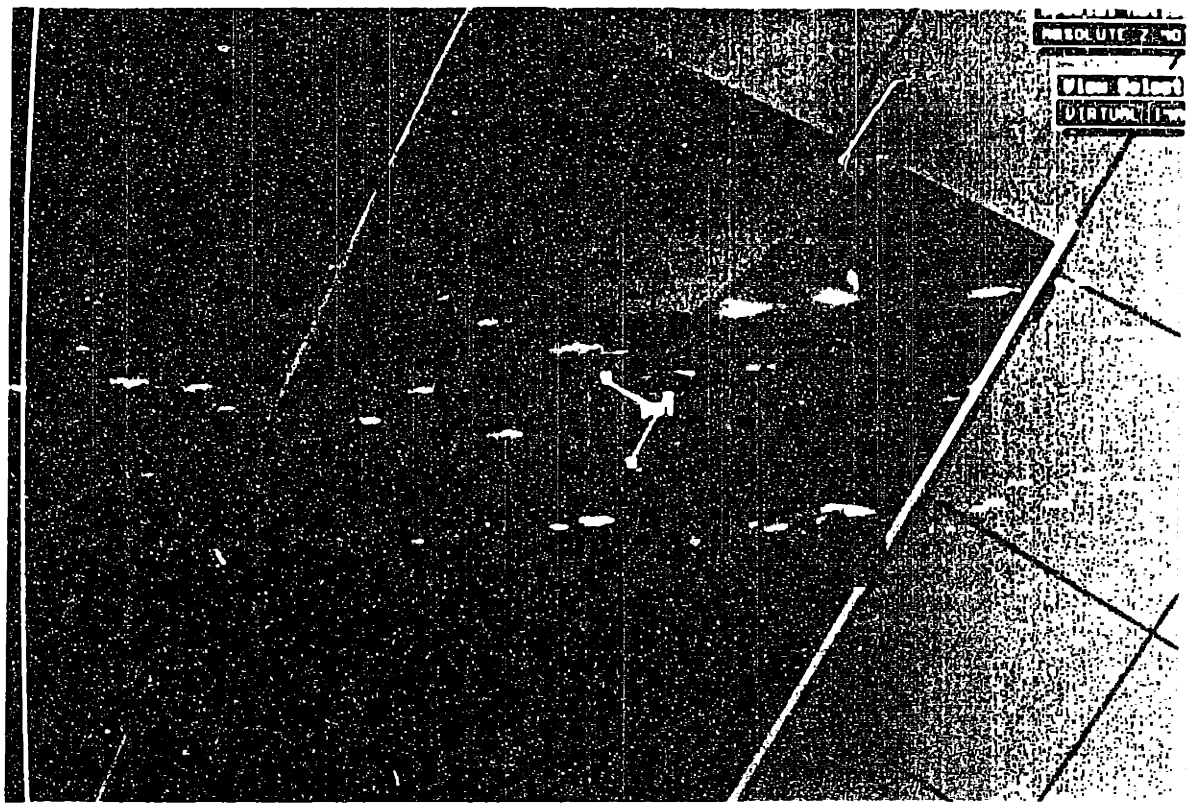

Figure 3-16: Object View 1: The large peg is transparent

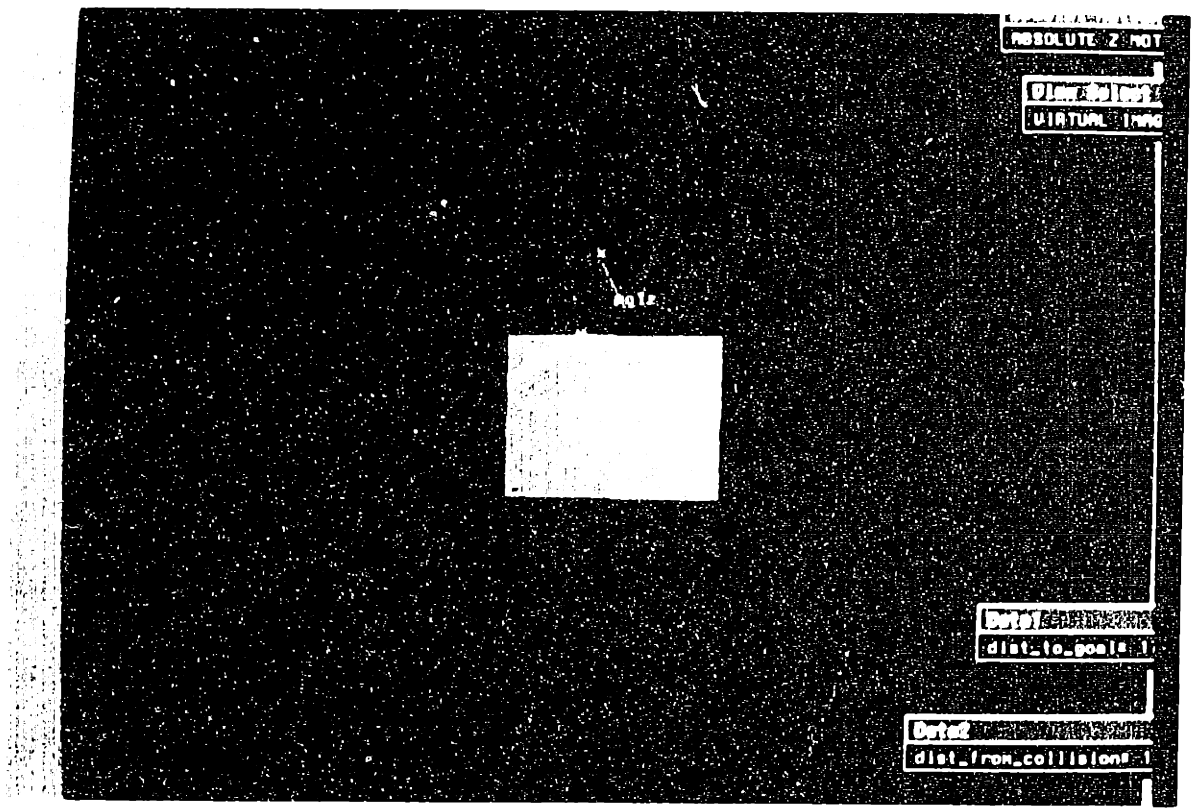

Figure 3-17: Object View 2: The box is transparent and the large peg is wireframe

was introduced to simulate the precision involved in the particular task. Some tasks required a high degree of accuracy, while others did not. This parameter can also be thought of as efffectively varying the size of the hole because the higher the tolerance, the smaller the hole that the peg had to enter.

In a typical run, the operator would first use the world camera to move near the object. He would position the robot so that the object was within sight of the wrist camera. Though this motion had inherently more error associated with it [12], it was acceptable while performing such gross motions. After getting within wrist camera range, the operator switched to the wrist camera view (or virtual image view) to maneuver the robot for the mating operation. The time limit on each trial was four and a half minutes. The data recorded for each trial was the time taken to complete the task (if completed at all), and the number of collisions that occurred.

3.5 Results

A main source of error identified before the experimentation was learning on the part of the operator. Since the entire experiment was so long, the operator's skill visibly improved as he progressed through the trials. This was one the reasons the training period was so long, in effort to begin experimentation only after the learning curve had somewhat leveled off. To further mitigate the problem, the experiment was counterbalanced by having each operator perform the trials in a different order. The same was done for each of the four runs within each trial. Thus, though the data for each operator reflected some degree of learning, overall, the learning effects averaged out.

Still, because of the length of each trial and the time constraints on the operators, it is important to emphasize that the results are not statistically significant. Since there were only four operators, and since each operator performed each run of every trial only once, there was a high degree of variability involved.

A scoring system was introduced to determine which trial produced the best overall results. The scoring system involved computing a score for each of the four runs in

each trial, and then summing the numbers over the four subjects. Thus, for each trial, there were sixteen data points. The equation used to compute the score was

$$SCORE = INT\left[((350 - TIME) - 20 \cdot COLL) \cdot \frac{TIME}{TIME + 1}\right]$$

where: $TIME$ = the time, in seconds, taken to complete the task. It was zero if the task was not completed.

$COLL$ = the numbers of collisions that occurred.

The term $\frac{TIME}{TIME+1}$ was introduced so that if $TIME$ was zero (ie the task was not completed), the $SCORE$ would be zero. The INT procedure truncated any fractions that were introduced.

With this weighting system, a collision equalled about twenty seconds in time. That was the penalty for a collision. In addition, if the task was not completed, then the score was zero regardless of the number of collisions. This was done because accomplishing the task was the top priority and, if it was not done, nothing else mattered.

The results can be found in Appendix 1. It includes data for each of the four operators for not only each trial, but the four variations within each trial. The total scores are summed at the end.

3.6 Experimental Conclusions

1. The best view was object view 1 (looking down). This is because not only did it abstract out the motion the object, but it also provided excellent depth qperception. Object view 2 (looking up) was deemed much better than the wrist camera view, but provided less clear depth perception than object view 1, because the operator was forced to look through the transparent object. Regardless, the results reveal that in a goal oriented task such as placing a peg in a hole, object views are critical. This is because by being given a clear focus on the goal, the operator has a reduced workload. He can concentrate solely on

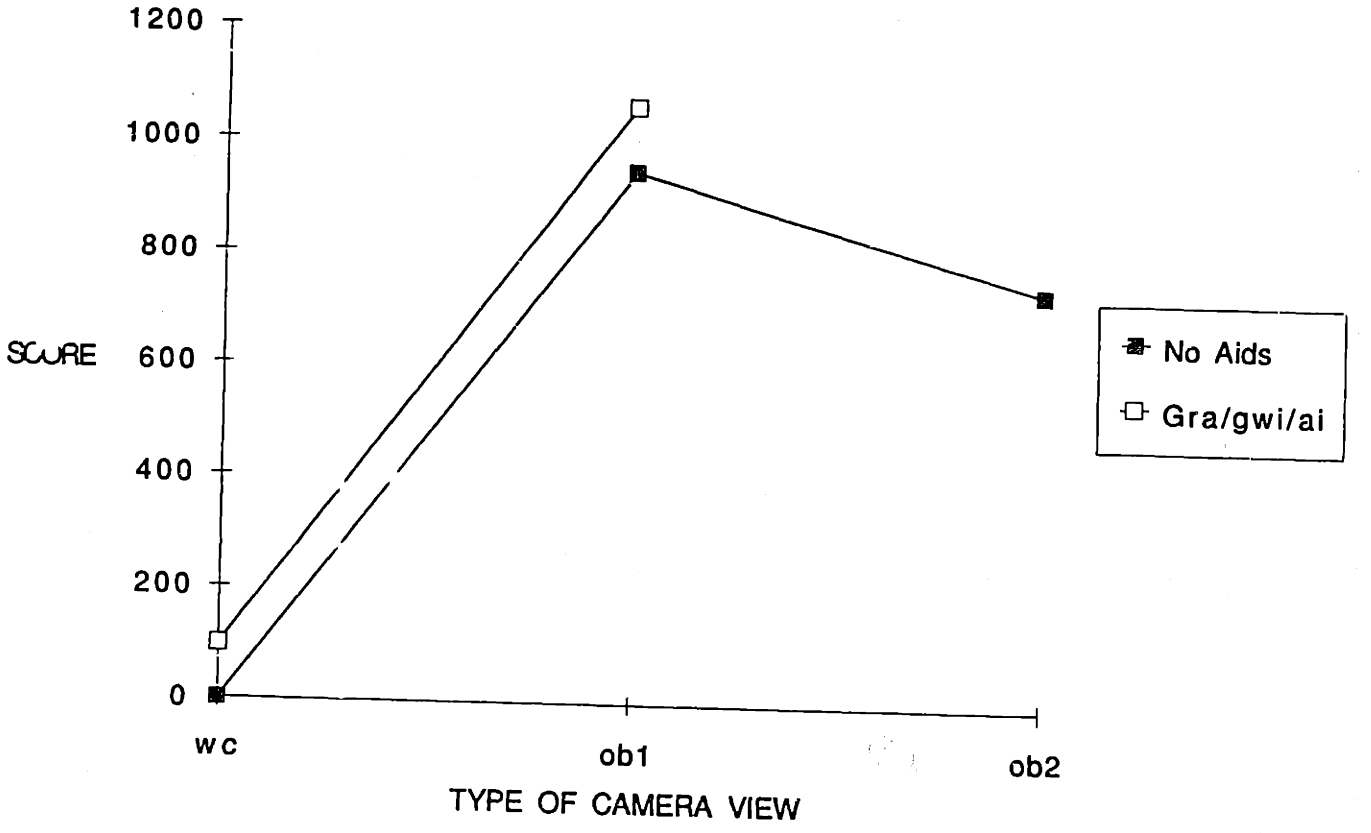

Figure 3-18: Plot of operator scores using three camera views with and without aids

moving the robot without worrying about locating the goal. The fact that the moving goal was transformed into a static one also eased operator workload. A graph of this result is shown in figure 3-18.

2. The graphical aide (relative rotation indictor) all by itself did not help the operator much. This was true because though the operator could get a better perspective on the relative yaw and pitch rotations, it was not easy to convert this knowledge into action. In other words, though the operator could easily see that he needed to rotate in the X direction, he had to take the time to convert this information into a mouse command. The fact that it was indeed easier to see relative rotation differences did not help much.

3. By introducing the interface window with the graphical aide, operator performance was greatly improved. This is because not only could the operator see how far he needed to move for proper rotational alignment, but also, he was provided the means to intuitively correct the situation. In essence, he was directly

moving the graphics themselves into the proper alignment.

4. The results with the increased alphanumeric feedback (lot) varied based on the amount of experience of the operator. An inexperienced operator found the numbers to be of no help at all, because he was too busy just controlling the robot. The numbers, if anything, only served to confuse the novice operator. But, as the operator grew in experience, the numbers became valuable. Fine motions could be performed with full confidence of the robot's exact relative location.

5. Interestingly enough, the auto mode did not greatly help the operators. This was more true the more the operator relied upon it. Some operators, in fact, relied so heavily on it, that they barely looked at the visual graphics. The problem with doing this was that not only were there are six numbers that required constant monitoring, but also that the rotations affected the translations (ie. they were coupled). In other words, the operator could spend the time to line up the translations, and then change the rotation because it is 5 degrees off. In doing so, the rotations could become aligned, but the translations got offset. Then, before the translations could be corrected, the rotations changed again. The result was that the operator was being far more accurate than he had to. Operator performance improved only when they realized that they should rely more on the graphics rather than the numerical feedback. Figure 3-19 shows a graph of the scores attained by operators while using all the above mentioned aids over the two types of camera views.

6. The increased performance by using the object views instead of the wrist camera view was not as large after introducing the graphical window and interface. Though operator performance still went up, it did not go up as dramatically as in the case when the wrist camera view was used. This reveals that by introducing graphical aides into the system, the adverse effects of having a poorer view can be reduced.

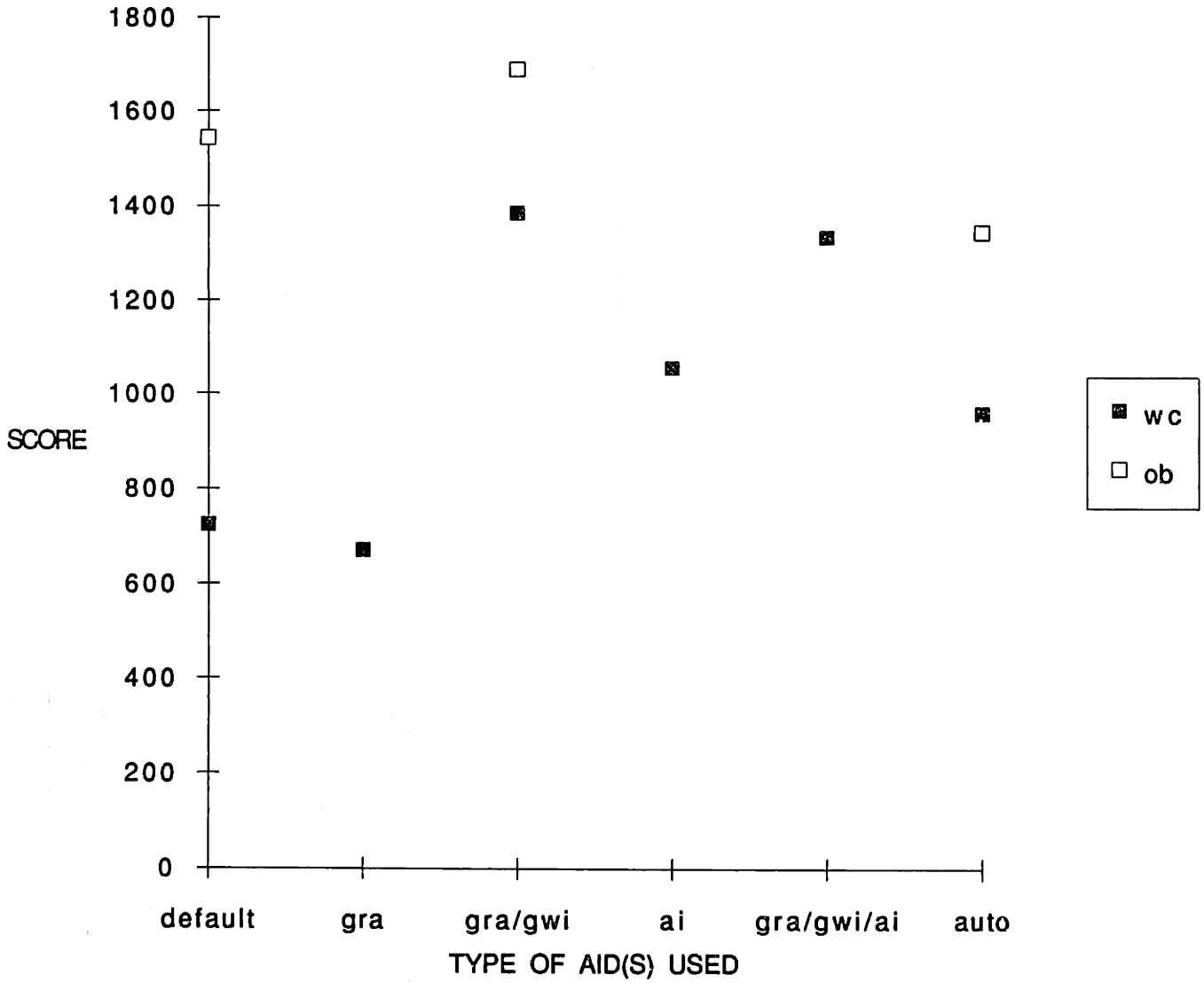

Figure 3-19: Plot of operator scores for the different graphical and control aids over two types of camera views

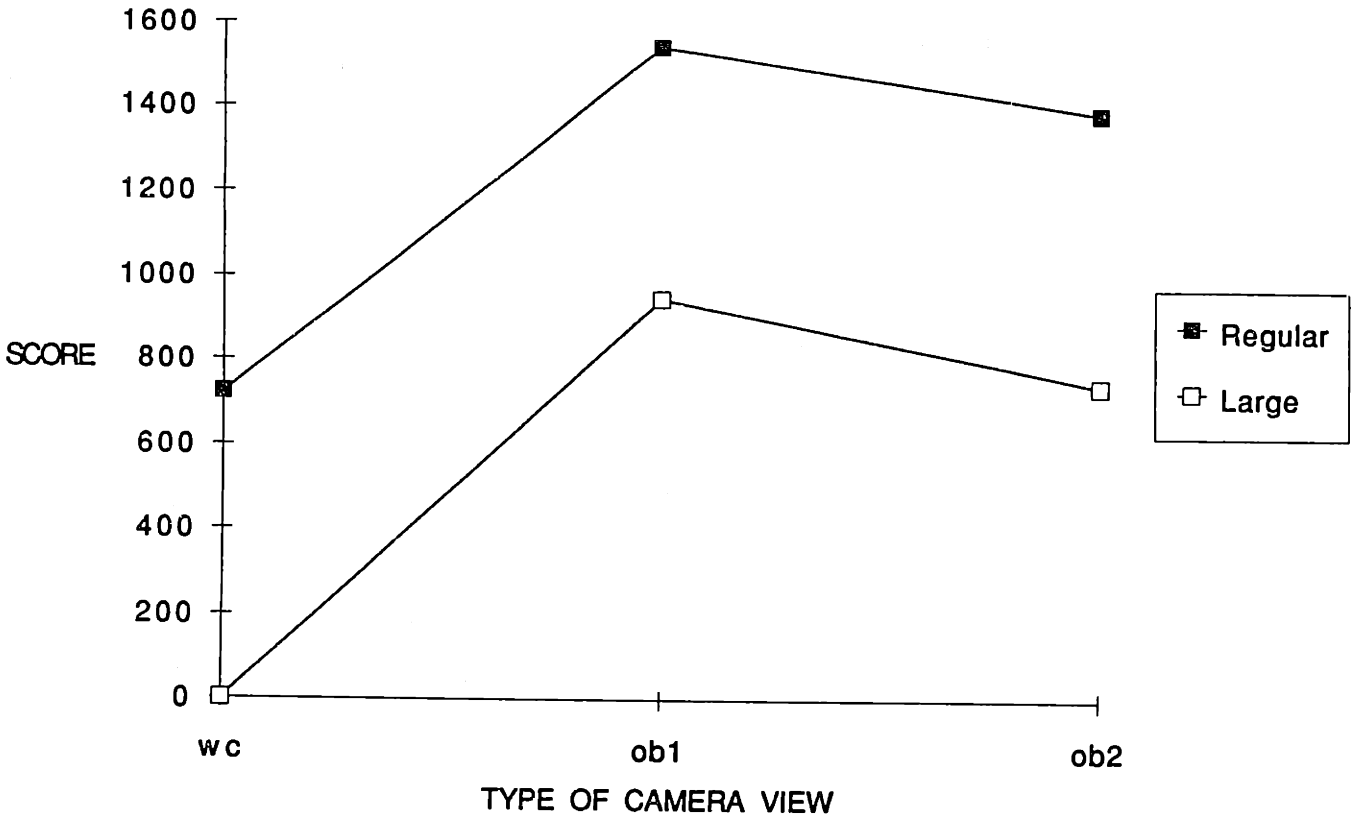

Figure 3-20: Plot of operator scores using the two types of pegs, regular and large, over the three types of camera views

7. When the large peg (bl) was used, the wrist camera view was completely ineffective. This was true even when the graphical aides were introduced. This result emphasizes the importance of a good camera view in teleoperation. Without being able to see the target, the operator, regardless of how much other numerical and graphical information he is given, can not perform the task. In addition, when the object view was used, performance was significantly higher with graphical aides than without. This was because by having a larger block on top of the peg, the task became harder, and the harder the task, the more graphical aides helped. It is also interesting to note that by changing the peg from regular to large, the results vary over the three camera views by a constant offset. A graph showing exactly this is shown in figure 3-20.

8. The joy_var interface was very difficult for the operator to use and provided him with little success in performing the task. There were three main reasons for this. The first was that the operator had no feel for the mapping of force onto

distance. In other words, it was difficult for him to tell how much force he had to apply to have the robot move a certain distance. If it had been something the operator had been used to, like turning a steering wheel on a car, it would have been easier. In addition, the operator's commands often became coupled. For instance, when the operator wanted to move in the x direction, the robot often also rotated in the y direction. Finally, the servo rate was so slow (limited bandwidth), that it took on the order of two seconds after the operator made a move for the robot to complete the move. This was because of the time delay in transmitting the commands to IGRIP. Also, since the joystick continuously emitted values even while not being read, a queue of values built up in IGRIP. Thus, this joystick interface simulated the type of communication delays that would be experienced by teleoperation tasks in space from the earth. Task speed decreased, and especially because a moving object was involved, task success decreased as well.

9. The joy_fix interface was better. The problems of relating force to distance and coupled motions were allieviated for this case. The results were good although the low bandwidth problem prevented it from being the best interface. Figure 3-21 shows a graph of the scores attained by using each of the three control interfaces while employing aids and no aids.

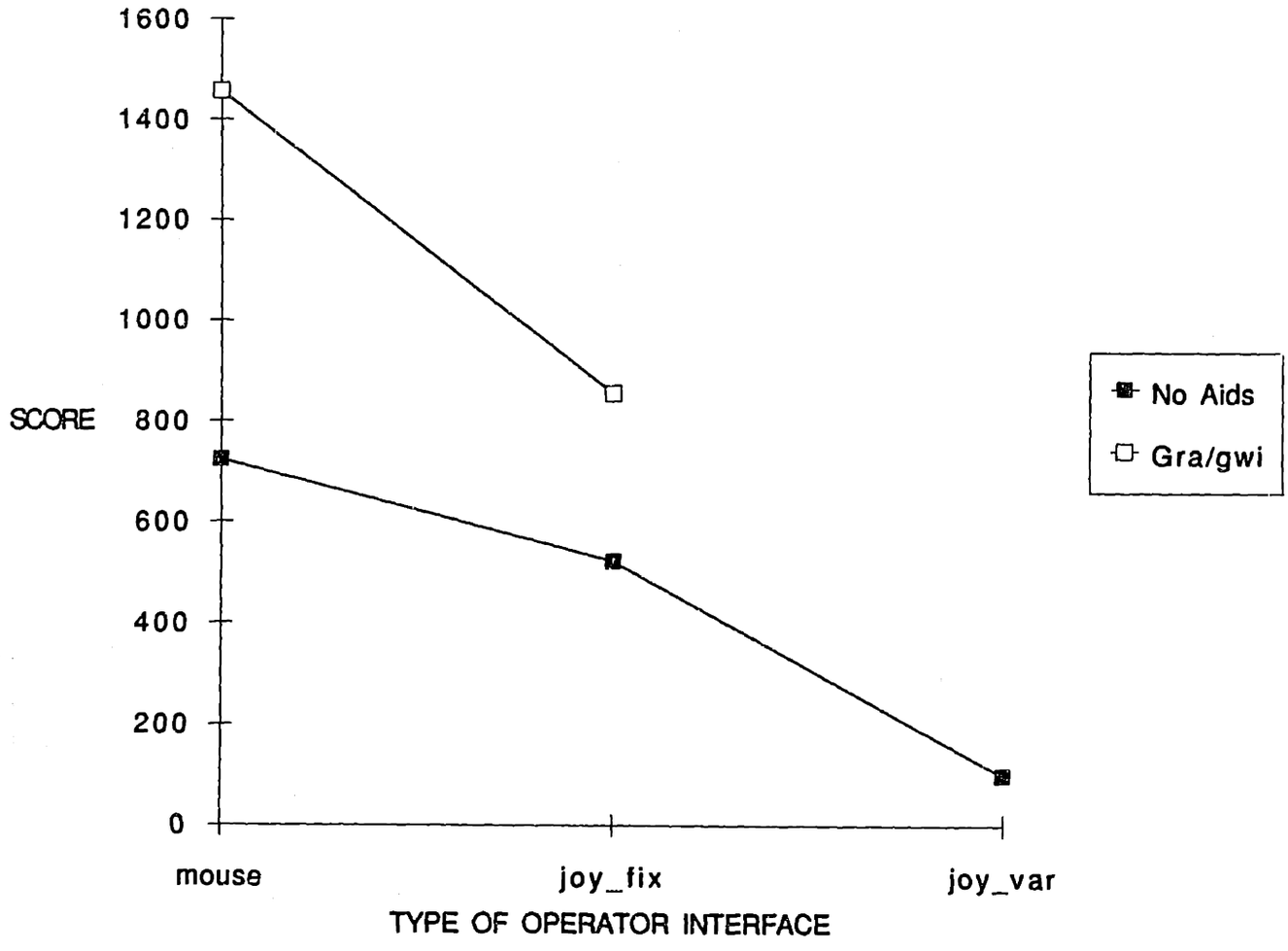

Figure 3-21: Plot of operator scores using the three control interfaces with and without aids

Chapter 4

Automation and Supervisory Control: Experimental Comparison of Three Control Methods

The task of performing an operation on a moving object is a very realistic and challenging one. The ability to perform tasks such as grappling a satellite or placing an ORU (orbital replacement unit) on a space station truss exhibiting oscillatory motion is critical in developing NASA's space program. In fact, in an extensive study performed on the Telerobot Testbed of the Jet Propulsion Laboratory, it was determined that the EVA Retriever task (satellite grappling) rated very high in the categories of technical contribution, importance to user and productivity/safety impact [15]. It had the highest technical contribution rating because the task required a combination of such complex technologies as manipulation, sensing, mobility, perception, reasoning, and communication. In other words, more than any other task NASA could perform in space, the EVA retriever task would most advance the state of the art in its related technologies. It was also determined that there was little chance that this task could be performed completely autonomously, and, in fact, a strong mixture of automation and teleoperation was needed. Given this, it seems appropriate to explore the pos-

sibility of adding automation to teleoperation. In this chapter, a sample scenario, comparable in difficulty to the EVA retriever problem, will be be introduced. The task will be broken down into some fundamental components and three methods for achieving the task will be presented. The simulation experiment for each method will then be described along with the qualitative results and conclusions.

4.1 Task Scenario

The task involves an object floating slowly through space. It is translating and rotating. This is analogous to a satillite drifting in space. There exists a T3-786 robot with a pointer in its gripper. There are two cameras in the environment: one fixed in space and the other attached to the robot's wrist. The operator's task is to maneuver the robot to trace a straight line on the face of the object. This operation simulates a inspection process performed by the robot.

In order to begin thinking about how to best tackle this task, it could be constructive to break the task down into some fundamental components. For this task, the fundamental components could be:

1) Task specification and objective

2) Moving the Robot

3) Monitoring the environment – safety checks

4) Monitoring the task – properly being done?

These components will be referred to later in the chapter.

4.2 Control Methods

There are three ways in which the task presented above could be performed. Each method will be introduced and described below.

4.2.1 Method 1: Teleoperation

In this solution method, the operator manually controls the robot to perform the task. The operator receives direct visual feedback from the two cameras in the environment. He is also provided with numerical feedback on the distance to target and nearest distance to collision. No other aid is available.

4.2.2 Method 2: Telerobotic Control

In this method, in addition to all the operator is given in method one, there is one extra feature. This feature is an autonomous tracking module which continuously tracks the location of the object and follows it. In other words, there no is no relative motion between the robot base and the object.

4.2.3 Method 3: Fully Autonomous Control

This method involves having all of the motion of the robot controlled by a computer. In addition to the resources above, the operator is given a graphics workstation into which the data from the camera views is supplied. The operator specifies the task he wants by viewing a graphic representation of the object's face and manually drawing a line on the computer screen This is done using a mouse. The operator then maintains a supervisory position. He receives graphical feedback of the operation (he can flip to different views etc.) and can abort the task at any time. The automated system, meanwhile, is responsible for locating and moving to the object face, and then tracing the straight line.

There are two means for acheiving this result. The first is to have the robot actually track the object while the operator specifies the task. Thus, in this case, the image that the operator sees is essentially the direct wrist camera view of the object. The second way of implementing this method is to leave the robot motionless until the operator specifies the task. The image on which the operator specifies the task is just an image from the CAD model. Only after the task is specified does the robot move to the object and begin tracing the line.

For the simulation, the second means of implementation was used. The reason for choosing the second means was the overhead of unneccessarily tracking the object before the task could be eliminated. There was no reason to track the object and show the direct camera view because even if the object did somehow change its configuration, the direct camera view would not reflect that change anyway. The reason was that the CAD model that the camera view data was matched with did not get updated. Also, by continuously tracking the object in space, the robot was bound to reach singularity points where it could no longer follow the object over incremental translations and rotations. Finally, because it was transparent to the operator whether the robot was actually moving with the object or not, it was more efficient to keep it stationary.

4.3 Simulation Experiment

The three solution methods described above were modelled in the IGRIP Simulation environment. Operators, both experienced and inexperienced, ran through each of the scenarios and formed qualitative conclusions about each. The specifics of the simulation, and the conclusions of the experiment are given below. It is important to note that it was carefully checked that the simulation only contained information that would be available to a real hardware system. The same assumptions used in the previous experiment (chapter 3) apply here as well.

4.3.1 Procedures

Seven to ten operators experimented with the three control methods. In all three methods, the operator began by seeing a view from the world camera (see figure 4-1) and was then given an option to switch to the wrist camera view. The world camera view was useful for gaining a perspective on the gross relative positions between the robot and the object. The experimental procedure for each task method follows.

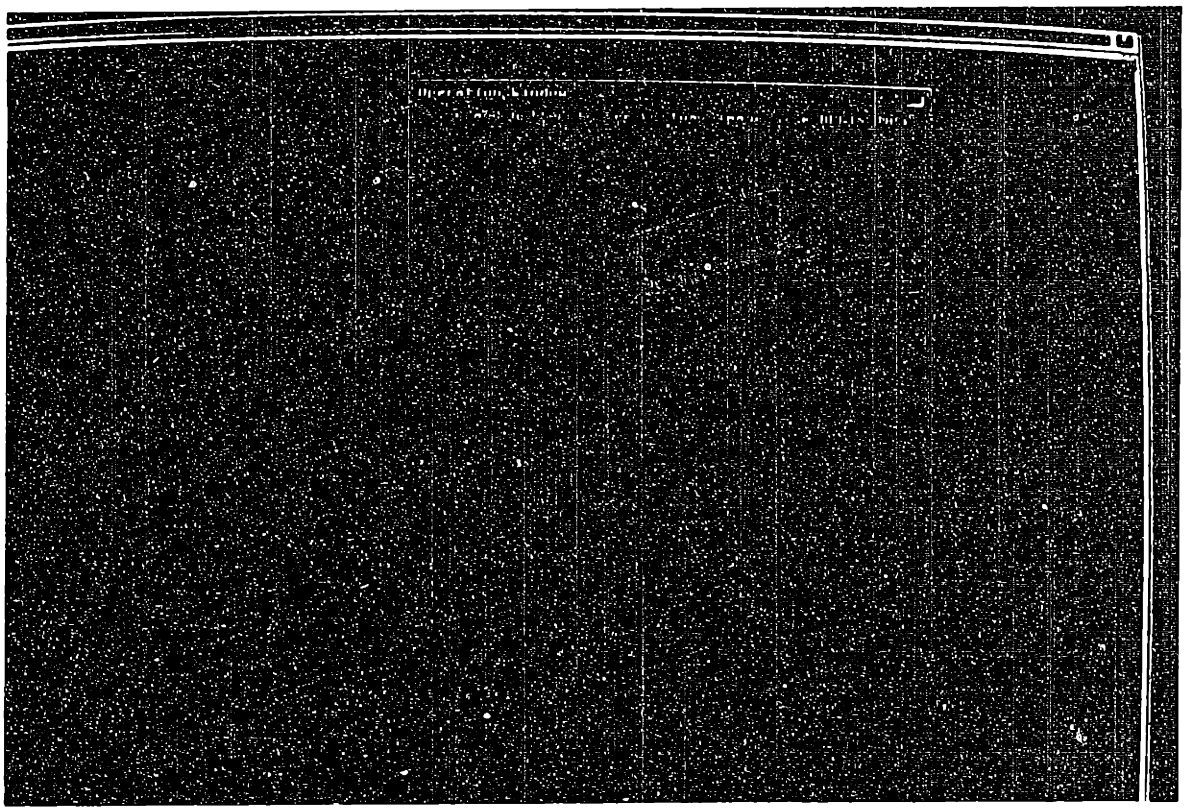

Figure 4-1: World Camera View of the Task

Method 1: Teleoperation

The operator relied on the world camera and wrist camera views and manually controlled the robot via a six degree-of-freedom joystick. When the operator felt that he was close enough to begin the tracing operation, he pressed the mouse, and a line was traced from the tip of the pointer. This line was drawn everytime the robot moved and simulated the path of pointer head. After each segment of the line was drawn, it moved relative to the object so that the line provided a means to check how close the operator really came to successfully completing the task.

Method 2: Telerobotic Control

A computer continuously moved the robot base so that it tracked the moving object. The operator was responsible for performing only the relative motions between the robot and the object. The operator initially tried to position the pointer right above the line to be traced (designated by a black line), and then attempted to manuever the

pointer straight along the line. Whenever the robot was moved, it's relative position to the object was to be maintained by the computer. As in the first method, when the operator felt he was matched up with the welding line, he clicked a button, and a line was traced from the pointer's tip. The operator manually controlled the robot (via a six degree-of-freedom joystick) to perform the task on what appeared to be a stationary object.

Method 3: Fully Autonomous Control

This method included the powerful feature of the virtual image. The operator was given a graphics workstation in which he could receive visual feedback not otherwise attainable (see chapter 2 for further details). In this case, the operator was given the ability to specify the task directly on the image of the object itself, and then be in a position to effectively monitor the task.

When the operator was ready to perform the task, he clicked the mouse button, and he was shown a static image of the face that he wished to operate upon. This static (virtual) image was then used as a drawing board for the operator to specify where on the object the line was to traced. Thus, given the relative position of the location to be traced, the operator could directly draw the line he wanted on the static face. By moving the mouse around on the screen, the operator was fed back the relative mouse position on the object in X and Y coordinates. The bottom left corner of the face was X=0, Y=0; the top right corner was X=20, Y=20. When the mouse was off the face, the operator was notified as such. The first point was selected by clicking the mouse button once. The computer then queried the operator if he wanted a horizontal, vertical, or diagonal line. By choosing another point and clicking the mouse again, the operator was able to watch the line being drawn according to his specification. A picture of this is shown in figure 4-2.

After the task was specified (where the line was to be traced), the operator was able to click a button, and the robot began performing the task. The robot's motion was autonomous; the operator's only responsibility was to monitor the task's progress. The robot was equipped with a tracking mechanism, an on-line path planner, and

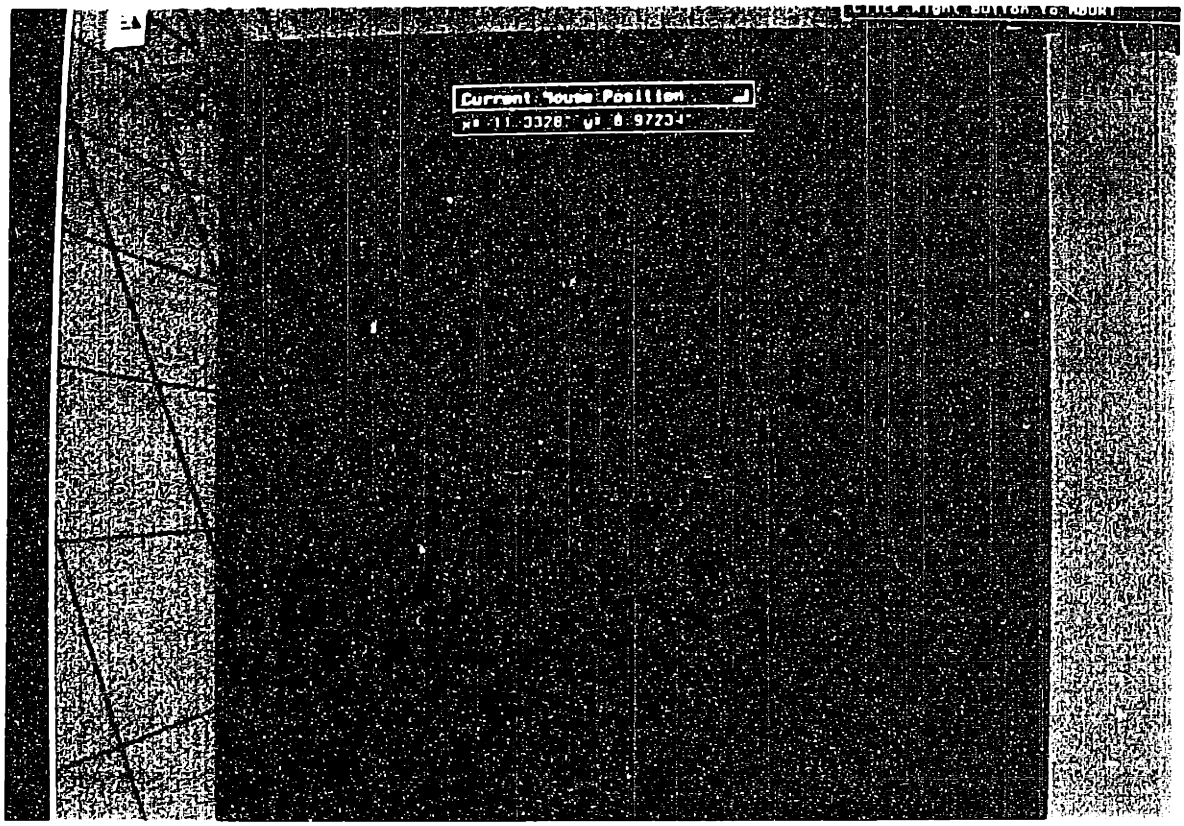

Figure 4-2: Virtual Image View of the Object

collision avoidance routines. The heuristics employed by the autonomous system are described below.

The robot began by receiving feedback as to the location of the first point of the line to be traced. A simple path planning algorithm then computed the straight line path from that point's position to the robot tip's current position. The robot then followed that straight line path. Because the position of the target point was constantly being updated, the straight line path was recomputed after every update cycle. The robot moved on each new path. A picture of the robot in motion can be seen in figure 4-3 This motion was performed iteratively until either the robot arrived at its target point (within some tolerance) or any part of the robot came within one inch of another object. If the robot was within an inch, it then automatically backed off by five inches in the direction of approach to the object. The object was then retracked. When the robot came to the target point, it then went down to the surface of the object and began tracing the line. A picture of this can be seen in figure 4-4. During this entire process, the operator was functioning in a supervisory

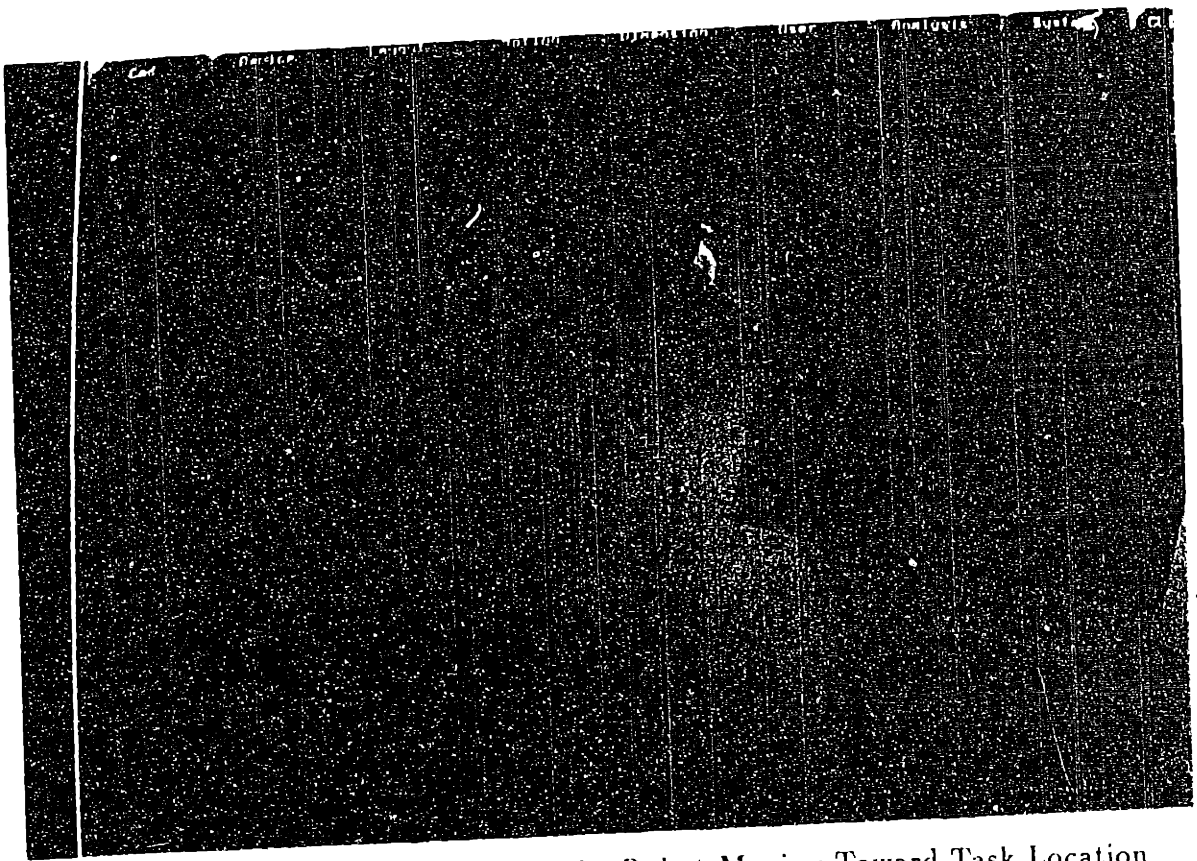

Figure 4-3: World Camera View of the Robot Moving Toward Task Location

role and had the ability to abort the task at any time.

4.3.2 Experimental Results and Conclusions

The first control method proved impossible for even the most experienced operators. The problem was that it was impossible for the operators to concentrate on the task (tracing along a line) and compensate for the unpredictable motion of the object at the same time. The mental and physical load was too great.

The second control method proved substantially easier for the operator. He was only responsible for worrying about manuevering the robot along a fixed path. One problem with this method was that since there was no force/torque feedback, it was difficult for the operator to get a feeling for the distance the pointer should be from the object's surface. Also, the operator felt hampered by not having the power to specify where on the face the line was to be traced.

The third solution method was most widely favored by the operator because the

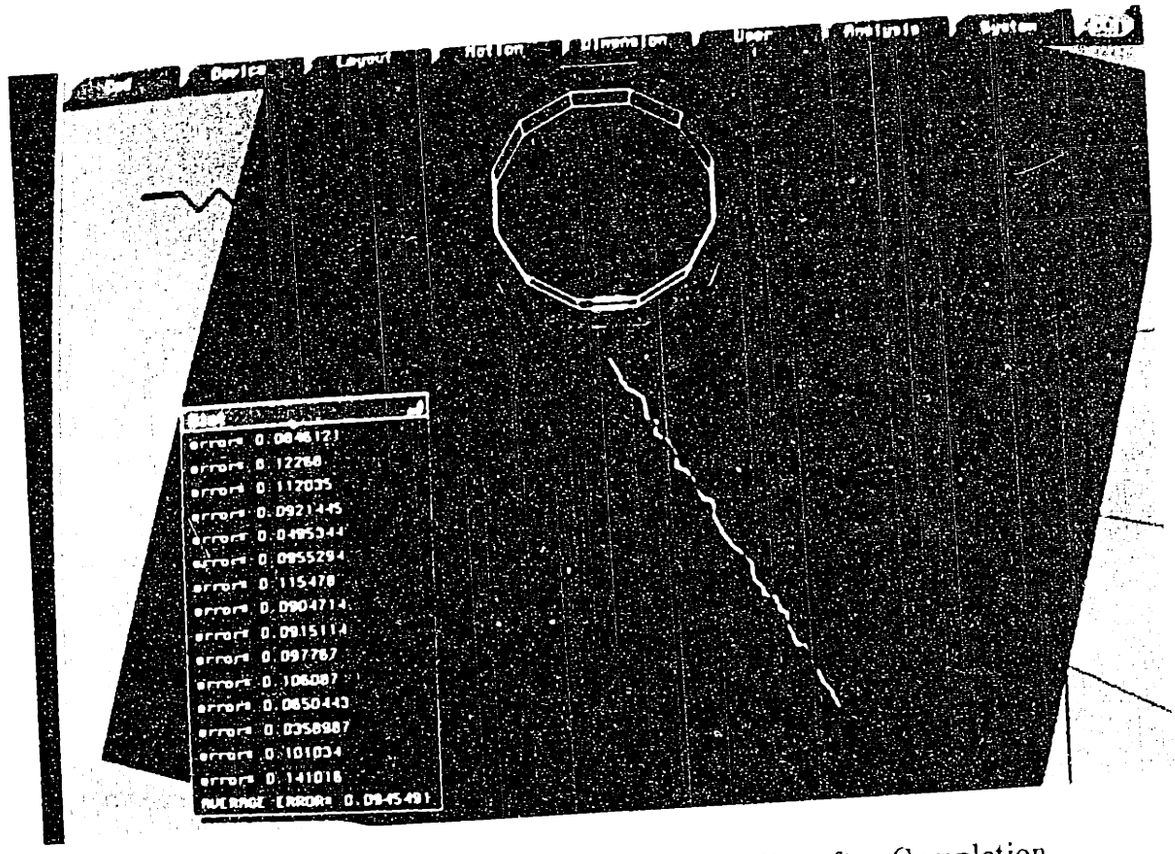

Figure 4-4: Wrist Camera View of Task Site after Completion

operator was only required to supervise the task. The operator was able to specify the task directly on the virtual image without worrying about moving the robot or tracking the object. Once the task was specified, the robot employed a path-planning algorithm which computed the straight line path between it and the most current position of the object. This path-planning and motion was conducted iteratively until the robot arrived safely at the target point (first point on line to be traced). During this process, the operator was able to closely monitor the robot's motion, and from visual feedback from the world and wrist cameras, was in a position to determine the progress of the task. If a problem arose, the abort button was hit. After the robot arrived at its target position, it began to autonomously trace the line based on the wrist camera feedback. At this point, the view was enlarged and the operator was able to closely supervise the line tracing process. The target line was drawn in black and the traced line was in white. To simulate some of the error involved in this autonomous process, an error factor was added into the program. The error consisted of a possible offset of .1 inches in each of the X, Y, and Z directions

for each point on the line.

Though, the autonomous solution method was overwhelmingly selected by the operators, the state of the current technology has to be kept in perspective. The art of real-time tracking is a technology that has yet to be mastered. But, though tracking in a task of this difficulty may not yet be feasible, it is certainly the direction the space robotic program must take. The area of human/machine interaction and the question of maximizing the abilities of each is something that must be carefully examined. In that light, the components of the EVA Retriever type task are listed below with the suggested specialist for each. A short discussion of each component follows.

1. Task specification and objective (human)
2. Actually Moving Robot to task (human/machine)
    a. Follow the object (machine)
    b. Perform the task, i.e. relative motion (human)
3. Monitoring environment – safety checks (human/machine)
    a. Gross checks – the task should still be done, no imminent problems etc. (human)
    b. Fine checks – no collisions are occurring (machine)
4. Monitoring that the task is being properly done (human/machine)
    a. Gross checks – the objectives are being satisfied (human)
    b. Fine checks – quantitative data on reaching objective (machine)

The first component is the responsibility of a human. He must determine what the task will be and a prioritized list of objectives must be made.

The second component requires a mixture of human and machine skills. As revealed by the simulation experiment, it was far too difficult for the human to track an unpredictablly moving object. The machine can do this much more easily because it

is essentially a simple, repetitve task. The machine must iteratively compute where the robot is relative to the object and move in that computed direction. The end task itself, on the other hand, should be performed by a human because he can knowns the task goals and can more easily assess the entire situation and formulate a solution. Depending on the level of the task, the machine would probablly have to be more robust than what is now feasible and/or cost-efficient.

The third component also requires a combination of human/machine interaction. A human is good at looking at a whole picture and making a judgement on a task's progress. He is good at determining relevancy of objects and events and can readily forsee probable future occurrances. An autonomous system, on the other hand, excels at making precise measurements.

The fourth component also requires a mixture of abilities. If the task objective can be precisely quantified (i.e. the line must match the seam within a tolerance of .5 inches), then it is easy for a computer to monitor this. In general though, overall task monitoring is the responsibility of the human operator.

4.4 Supervisory Control

Because of the growing level of task difficulty in space robotics, the theory of supervisory control will become increasingly more important. According to Ferrell and Sheridan, there are three main control loops involved with supervisory control [6]. The first loop is called the remote loop, and it represents local autonomy of the telerobot. The loop goes from the manipulator to a local computer and never actually involves the operator. The second loop, the supervisory loop, represents the interface between the operator and the remote manipulator. This includes setting goals for the manipulator and directly guiding the effectors. The third loop is the local loop. This loop is independent of the manipulator and involves a predictive display system which allows the operator to visualize the manipulator's interaction with the environment as if he were looking at it directly. The time delay problem is an issue here and is overcome in works including Sheridan and Noyes[10]. The key to tackling the time

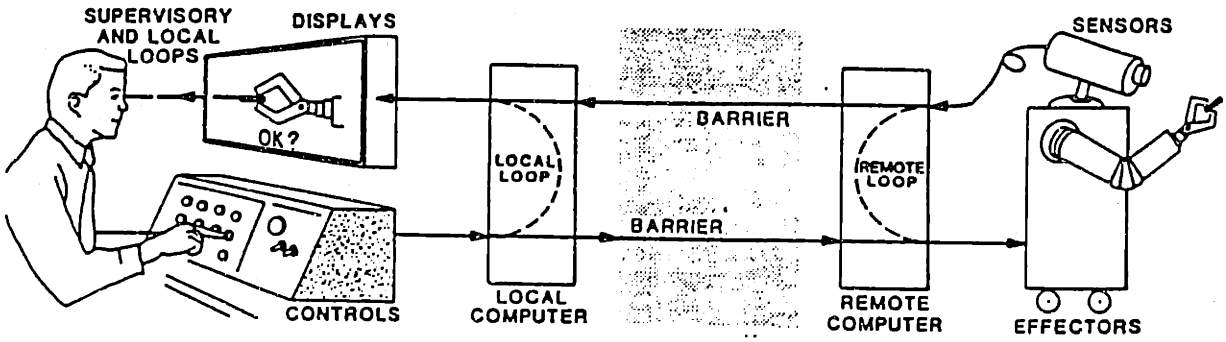

Figure 4-5: The Three Loops of Supervisory Control

delay problem is incorporating a module that can accurately predict the motion of all objects in the world and, by performing a forward simulation, displaying the objects in their new positions without waiting for the actual feedback. The time delay problem is discussed further in the next chapter. Figure 4-5 displays the the three loops /citeferrell .

For the EVA retriver task, the duties within each loop can be assigned as follows:

1. The remote loop could be responsible for automously tracking the moving obstacle, and thus abstracting out the motion of the object for the operator. This requires high-speed vision feedback and processing, collision avoidance, and some basic path planning.

2. In the supervisory loop, the operator could specify the task on the virtual image (where exactly the line must be traced), and perform the relative motion to move the robot on the line.

2. In the local loop, the operator receives feedback from the sensor hardware and

views the results on a computer. He could use this information, along with other simulated data, to plan and execute manual control of the robot.

Chapter 5

Real World Application

How can graphical displays and simulations be used in the real world? One approach is that a simulation can be run one step ahead of the real robot. In other words, the operator can control the robot by first using a real-time simulation, and as long as there are no problems, the commands can then be sent to the real robot. This is valuable because it allows the operator to test his commands on the simulation first, and thus ensure a safe operation. Also, showing the robot's motions in real-time is important because, given long lag times between the operator's move and the robot's, operator performance greatly increases by overlaying a real-time graphics display of the move [10]. This is intuitive because if you are given real-time visual feedback, you know where the robot is relative to the world; otherwise, you must either wait until the robot actually moves and feeds back its position, or make a best guess as to the current position. This approach is in the spirit of the predictive display [10].

How far the simulation works ahead of the real robot (simulation delay) can be varied depending on transmission time and how quickly the environment changes. The key assumptions here are that the world model is accurate and up-to-date and that the environment does not change so fast that by the time the simulation updated, it is out-of-date. It is important to keep in mind that by reducing the simulation delay, the real world will more closely resemble the world model, but the purpose of the simulation gets compromised. Even if a collision is detected, the robot may not be able to stop in time. In simulation a robot can stop instantaneously; in reality, it

must decelerate, and may travel inches or feet before fully halting.

5.1 Predictive Displays

If the local site was the earth and the manipulator was in space, the communication delay could be quite substantial. For example, round-trip delays for vehicles on or near the moon are typically three seconds. In order to better deal with this problem, Conway, Volz, and Walker [3] used a predictive display approach while adding in three additional features: the time clutch, position clutch, and time brake. Because of the fact that operators can generate a path much faster than most telerobots (i.e. NASA's Remote Manipulator System) can follow, these features allow an operator to work in simulation well ahead of the telerobot, to insure smooth, safe operation.

5.1.1 Example

Assume the task is to move a manipulator to a bolt and manueuver the end of the gripper around the bolt. This is a teleoperated task and the local site is on earth and the remote site is thousands of miles away in space. The operator moves the robot in simulation engaging the time clutch and working well ahead of the actual telerobot's position. This enables the operator to "look and think ahead" of the telerobot under control, with the look ahead time completely specified by the operator. Then, when the operator gets near the bolt, he can disengage the position clutch which allows the operator to move the robot in simulation without having the command sent to the robot. He can then perform all of the fine motions and adjustments involved with fitting the gripper around the bolt without the fear of an actual collision or jittery motion. Then, when the forward simulation robot is aligned properly, the position clutch can be reengaged, with the simulation forming a short, smooth path from the last position (where the position clutch was last disengaged) to the current position. Thus, the operator is able to work on the complex, time-consuming motions ahead of time while the telerobot is still responding to the previous commands. Time otherwise wasted by the communication delay in waiting for the robot's motion gets

put to good use by the operator. Though this is a useful concept, it has one basic limitation. It assumes that the environment is basically static and that none of the objects is moving. If this is not the case, the problems are apparent. The important issue here is the accuracy of the initial world model and the rate and accuracy by which it gets updated. This is crucial because even if the environment is completely static, if the world model does not get constantly checked and updated from real-world sensors, the system is doomed to failure. A robust system is one that is flexible enough to adapt to a change in the environment, but also one that can periodically update its views of the world even if nothing has changed. If this problem had been the realistic problem of grappling a moving satillite, predictive displays would not suffice.

The only real solution to this problem is to introduce some local autonomy and thereby circumvent the communication delays altogether. One method to solve this would be to have the telerobot autonomously track the moving object, and then have the operator perform the specific task (as discussed in the previous chapter).

5.2 Sources of Error

In the opening remarks for the NASA Conference on Space Telerobotics [9], Dr. Montemerlo, the manager of the Automation and Robotics Program at NASA, said, "Every time we developed either a tool or a jig based on a CAD/CAM database, to fix one of those satellites or hold it down, it did not work because the CAD/CAM database was wrong, every time, 100 percent of the time." It is clear that a CAD/CAM system can not be solely relied on in telerobotics and that periodic reality checks must be made. Experiences from the JPL Telerobotic Testbed [1] are also enlightening is this area.

One of the most fundamental problems when converting this system into the real world is that of accurately and quickly determining the location of an object in 3D space [12]. The basic components of a vision system are:

- CAD database of all the objects in the world.

- Sensing system which returns a digitized picture of the object and its relative measurements.
- Feature extraction system which takes the raw data and distinguishes features on the object and their relative positions. Features include points, vectors, line segments, axes, surface patches, edges, boundaries etc.
- Matching system which tries to pair the sensed features with the CAD database's features,
- Computing system which computes the transformations for locating the entire object relative to the robot's tip.

5.3 Tracking

Tracking a translating and rotating object in space is a very difficult problem. It is a state of the art technology which is still evolving [8] [7]. There are two fundamental approaches to tracking an object: either iteratively recieving sensor feedback on the object's position and moving to that position, or predicting the motion of the object and moving ahead to meet it. The problem with the first approach is that the robot always remains one step behind the object. Because it takes a finite amount of time to receive sensor information, process it to find the position and orientation of the object, and move to that location, the robot will never be exactly where it should. In the ideal, theoretical case where processing and movement times approach zero, this method is feasible and efficient. Unfortunately, this is not the case in reality. The only way to implement a real system with this approach is to stipulate that the object moves very slowly. The other approach is what most researchers today are investigating. From a sequence of target positions, the system must be able to predict future positions of the target. One big assumption here is that there are no external forces on the object. Thus, given the principal axes of inertia, both the translations and rotations about these axes will be uniform. The most common method for tackling this problem involves the Kalman Filter.

Chapter 6

Conclusion

The important results and contributions of this paper are listed below along with recommendations for future work.

6.1 Contributions

Three major contributions from this work are:

1. The concept of a virtual image was introduced and its use in the teleooperation domain was displayed. The feasibility of employing graphic displays, instead of direct camera feedback, was demonstrated for teleoperation tasks.

2. Three main graphical display categories (camera views, graphical display content, and alphanumeric display content) were identified and the relative merits of each were determined through experimentation. Also, multiple operator control interfaces (mouse, six dof joystick) and schemes (auto mode, graphical window interface) were introduced, and their relative effectiveness and interdependence with the graphical feedback were determined experimentally.

3. A framework was provided to begin answering the question of how much autonomy should be introduced into a system. The concept of supervisory control was explored and deemed critical in enhancing space telerobotics. This result was drawn through experimentation using three control methods (teleoperation,

telerobotic control, and fully autonomous control) for performing a complex task.

6.2 Future Work

The most obvious next extension to this work is to connect the simulation to a real robotic system. In this research, the simulation artificially synthesized data which is supposedly came from a vision system. By connecting the simulation to a real robot, the program could receive input from an actual vision system, which it could then process. Then, based on the derived graphics, the operator could move the graphical robot via the control interface, after which the commands would get sent to the real robot. By incorporating the visual display aids discussed in this research to an actual hardware system, much more knowledge would be gained about the real-time feasibility of employing the graphics. Quantatative and qualitative conclusions could be drawn about the nature of the vision system required, the update rates, variability of the environment etc.

Another extension for this work is to incorporate other sensors into the system. Currently, the only real-world feedback that was received was through a video camera. By adding, for example, a force/torque sensor, the simulation could be extended to provide this additional feedback. This added feedback would prove especially useful in the mating type operations performed in this research.

Appendix A

Experimental Data

		Op#1			Op#2	Data		Op#3			Op#4	
TRIAL	TIME	COLL	SCORE	TIME	COLL	SCORE	TIME	COLL	SCORE	TIME	COLL	SCORE
1 --	171	3	118	200	2	109	145	0	203	230	1	99
1 f/h	0	4	0	0	3	0	0	4	0	0	3	0
1 fast	0	1	0	0	1	0	0	3	0	0	1	0
1 hard	183	0	166	261	3	28	0	4	0	0	5	0
TOTAL			284			137			203			99
2 --	130	0	218	168	0	180	141	0	207	123	0	225
2 f/h	0	2	0	0	4	0	237	1	92	0	2	0
2 fast	265	0	84	186	1	143	154	0	194	256	1	73
2 hard	0	1	0	0	2	0	0	2	0	222	0	127
TOTAL			302			323			493			425
3 --	170	0	178	259	2	50	189	1	140	165	0	183
3 f/h	190	0	159	0	4	0	0	4	0	0	5	0
3 fast	203	0	146	210	1	119	232	1	97	201	3	88
3 hard	213	0	136	0	5	0	220	2	89	0	3	0
TOTAL			619			169			326			271
4 --	0	1	0	261	2	48	187	0	162	137	1	191
4 f/h	180	0	169	0	2	0	0	3	0	0	2	0
4 fast	0	2	0	0	4	0	209	2	100	0	2	0
4 hard	0	1	0	0	3	0	0	2	0	0	3	0
TOTAL			169			48			262			191
5 --	172	0	176	167	0	181	167	1	162	234	0	115
5 f/h	150	0	198	0	3	0	198	1	131	0	2	0
5 fast	0	2	0	189	0	160	255	2	54	211	2	98
5 hard	263	1	66	0	1	0	0	1	0	214	1	115
TOTAL			440			341			347			328
6 --	221	1	108	174	0	174	194	1	135	122	0	226
6 f/h	225	0	124	0	3	0	246	0	103	199	1	130
6 fast	186	1	143	149	0	199	211	0	138	0	2	0
6 hard	0	1	0	0	3	0	189	1	140	240	2	69
TOTAL			375			373			516			425
7 --	160	1	168	137	1	191	129	0	219	230	1	99
7 f/h	0	3	0	0	4	0	0	3	0	0	4	0
7 fast	0	3	0	0	4	0	230	2	79	0	4	0
7 hard	228	2	81	0	3	0	181	1	148	238	2	71
TOTAL			249			191			446			170
8 --	215	0	134	151	0	197	160	0	188	170	2	139
8 f/h	0	6	0	229	3	60	0	2	0	0	4	0
8 fast	167	1	162	0	1	0	169	2	140	234	1	95
8 hard	252	2	57	245	1	84	211	3	78	0	3	0
TOTAL			353			341			406			234
9 --	220	1	109	228	1	101	0	3	0	230	1	99
9 f/h	192	1	137	0	4	0	175	1	154	0	3	0
9 fast	0	3	0	225	2	84	194	2	115	168	1	161
9 hard	0	4	0	0	3	0	0	3	0	0	2	0
TOTAL			246			185			269			260

							Data							
10 --	130	0	218		121	0	227		147	0	201		196	1	133
10 f/h	0	1	0		230	3	59		159	1	169		0	3	0
10 fast	0	3	0		0	1	0		0	3	0		265	2	44
10 hard	228	0	121		266	2	43		0	3	0		179	2	130
TOTAL			339				329				370				307
11 --	250	3	39		0	1	0		0	5	0		249	2	60
11 f/h	0	4	0		0	3	0		0	3	0		0	7	0
11 fast	0	3	0		0	4	0		0	2	0		0	3	0
11 hard	0	3	0		0	2	0		0	4	0		0	2	0
TOTAL			39				0				0				60
12 --	213	0	136		210	1	119		0	4	0		245	2	64
12 f/h	0	3	0		0	2	0		0	3	0		0	4	0
12 fast	0	2	0		0	0	0		206	1	123		268	3	21
12 hard	0	3	0		0	1	0		0	2	0		249	2	60
TOTAL			136				119				123				145
13 --	245	1	84		203	1	126		230	1	99		0	4	0
13 f/h	0	3	0		0	2	0		0	4	0		0	5	0
13 fast	0	4	0		235	1	94		261	0	88		198	0	151
13 hard	221	0	128		0	3	0		243	1	86		0	1	0
TOTAL			212				220				273				151
14 --	0	3	0		0	4	0		0	1	0		0	4	0
14 f/h	0	1	0		0	5	0		0	1	0		0	3	0
14 fast	0	5	0		0	2	0		0	2	0		0	6	0
14 hard	0	3	0		0	6	0		0	1	0		0	6	0
TOTAL			0				0				0				0
15 --	176	0	173		190	0	159		243	1	86		251	2	58
15 f/h	267	1	62		0	3	0		0	3	0		0	4	0
15 fast	0	3	0		0	2	0		222	2	87		190	1	139
15 hard	245	1	84		0	2	0		210	2	99		0	2	0
TOTAL			319				159				272				197
16 --	243	1	86		203	1	126		0	2	0		235	0	114
16 f/h	0	2	0		0	2	0		0	5	0		0	2	0
16 fast	0	4	0		220	2	89		0	2	0		247	0	102
16 hard	221	0	128		0	3	0		234	1	95		0	3	0
TOTAL			214				215				95				216
17 --	212	2	97		0	4	0		0	2	0		0	3	0
17 f/h	0		0		0	3	0		0	2	0		0	3	0
17 fast	0		0		0	5	0		0	2	0		0	2	0
17 hard	0		0		0	2	0		0	5	0		0	4	0
TOTAL			97				0				0				0
18 --	231	0	118		185	0	164		210	0	139		181	0	168
18 f/h	0	3	0		0	3	0		0	3	0		263	2	46
18 fast	0	3	0		181	0	168		239	1	90		0	3	0
18 hard	256	1	73		0	2	0		0	4	0		211	2	98
TOTAL			191				332				229				312

					Data
	TRIAL TOTALS			GRAND TOTALS	
1	Norm-SCORE	529		TOT - NORM	8226
	F/H - SCORE	0		TOT - F/H	1793
	FAST - SCORE	0		TOT - FAST	3868
	HARD -SCORE	194		TOT - HARD	2600
	TOT - SCORE	723		GRAND - TOT	16487
2	Norm-SCORE	830			
	F/H - SCORE	92			
	FAST - SCORE	494			
	HARD -SCORE	127			
	TOT - SCORE	1543			
3	Norm-SCORE	551			
	F/H - SCORE	159			
	FAST - SCORE	450			
	HARD -SCORE	225			
	TOT - SCORE	1385			
4	Norm-SCORE	401			
	F/H - SCORE	169			
	FAST - SCORE	100			
	HARD -SCORE	0			
	TOT - SCORE	670			
5	Norm-SCORE	634			
	F/H - SCORE	329			
	FAST - SCORE	312			
	HARD -SCORE	181			
	TOT - SCORE	1456			
6	Norm-SCORE	643			
	F/H - SCORE	357			
	FAST - SCORE	480			
	HARD -SCORE	209			
	TOT - SCORE	1689			
7	Norm-SCORE	677			
	F/H - SCORE	0			
	FAST - SCORE	79			
	HARD -SCORE	300			
	TOT - SCORE	1056			
8	Norm-SCORE	658			
	F/H - SCORE	60			
	FAST - SCORE	397			
	HARD -SCORE	219			
	TOT - SCORE	1334			
9	Norm-SCORE	309			
	F/H - SCORE	291			
	FAST - SCORE	360			
	HARD -SCORE	0			
	TOT - SCORE	960			

10	Norm-SCORE	779	
	F/H - SCORE	228	
	FAST - SCORE	44	
	HARD -SCORE	294	
	TOT - SCORE	1345	
11	Norm-SCORE	99	
	F/H - SCORE	0	
	FAST - SCORE	0	
	HARD -SCORE	0	
	TOT - SCORE	99	
12	Norm-SCORE	319	
	F/H - SCORE	0	
	FAST - SCORE	144	
	HARD -SCORE	60	
	TOT - SCORE	523	
13	Norm-SCORE	309	
	F/H - SCORE	0	
	FAST - SCORE	333	
	HARD -SCORE	214	
	TOT - SCORE	856	
14	Norm-SCORE	0	
	F/H - SCORE	0	
	FAST - SCORE	0	
	HARD -SCORE	0	
	TOT - SCORE	0	
15	Norm-SCORE	476	
	F/H - SCORE	62	
	FAST - SCORE	226	
	HARD -SCORE	183	
	TOT - SCORE	947	
16	Norm-SCORE	326	
	F/H - SCORE	0	
	FAST - SCORE	191	
	HARD -SCORE	223	
	TOT - SCORE	740	
17	Norm-SCORE	97	
	F/H - SCORE	0	
	FAST - SCORE	0	
	HARD -SCORE	0	
	TOT - SCORE	97	
18	Norm-SCORE	589	
	F/H - SCORE	46	
	FAST - SCORE	258	
	HARD -SCORE	171	
	TOT - SCORE	1064	

Bibliography

[1] B. Balaram, J. Beahan, and H. W. Stone. Experiences with the jpl telerobot testbed - issues and insights. *Proceedings of the NASA Conference on Space Telerobotics, JPL 89-7*, 2:163–172, 1989.

[2] A. Bejczy, W. Kim, and S. Venema. The phantom robot: Predictive displays for teleoperation with time delay. *IEEE Int. Conf. on Robotics and Automation*, pages 546–551, 1990.

[3] L. Conway, R. Volz, and M. Walker. Tele-autonomous systems: New methods for projecting and coordinating intelligent action at a distance. *Proceedings of the NASA Conference on Space Telerobotics, JPL 89-7*, 3:147–157, 1989.

[4] D.J. Cwynar. Orbital maneuvering vehicle controllability simulations. *SOS Lab Report OMV-85-01, Martin Marietta*, 1985.

[5] D.B. Diner and S.C. Venema. Graphic overlays in high-precision teleoperation: Current and future work at jpl. *Proceedings of the NASA Conference on Space Telerobotics, JPL 89-7*, 3:511–520, 1989.

[6] W.R. Ferrell and T.B. Sheridan. Supervisory control of remote manipulation. *IEEE Spectrum*, 4:81–88, 1967.

[7] L.V. Gool, A. Oosterlinck, and D.B. Zhang. Stochastic predictive control of robot tracking systems with dynamic visual feedback. *Proceedings of the 1990 IEEE International Conference on Robotics and Automation*, 1:610–615, 1990.

[8] Y. Kay and S. Lee. An accurate estimation of 3-d position and orientation of a moving object for robot stereo vision: Kalman filter approach. *Proceedings of the 1990 IEEE International Conference on Robotics and Automation*, 1:414–419, 1990.

[9] M. Montemerlo. Evolving space teleoperation to space telerobotics: Research and systems considerations. *Proceedings of the NASA Conference on Space Telerobotics, JPL 89-7*, 1:7–14, 1989.

[10] M. Noyes and T.B. Sheridan. A novel predictor for telemanipulation through a time delay. *Proceedings of the Annual Conference on Manual Control, NASA Ames*, 1984.

[11] J.E. Pennington. A rate-controlled teleoperator task with simulated transport delays. *NASA Technical Memo 85653, Marshall Space Flight Center, AL: NASA Publications*, 1983.

[12] L. Shao and R. Volz. Methods and strategies of object localization. *Proceedings of the NASA Conference on Space Telerobotics, JPL 89-7*, 1:229–239, 1989.

[13] G. Varsi. Telerobotics for the efficient utilization of space. *Presented at the XXXIXth Congress of the International Astronautical Federation, IAF-88-023*, 1988.

[14] J.P. Yorchak. Teleoperator human factors study: Task 6. *Nasa Contract NAS8-35184, Martin Marietta*, 1986.

[15] W. Zimmerman. Establishing viable task domains for telerobotic demonstrations. *Proceedings of the NASA Conference on Space Telerobotics, JPL 89-7*, 5:111–120, 1989.