〔非公開〕

ＴＲ－Ｃ－０１２１

A Sophisticated Manipulation Aid in a Virtual Environment

イー　エイミー　　北村　喜文
Amy YEE　　Yoshifumi KITAMURA

１９９５　５．３１

ＡＴＲ通信システム研究所

A Sophisticated Manipulation Aid in a Virtual Environment

Amy Yee and Yoshifumi Kitamura
Artificial Intelligence Department
ATR Communication Systems Research Laboratories

May 31, 1995

Contents

List of Figures

List of Tables

Abstract

Object interactions in virtual reality is a subject of much interest as we attempt to apply virtual reality techniques in real world problems. The approach taken for handling interactions between objects may vary greatly depending on the application. In this report we describe a general method to assist users in the manipulation of arbitrary virtual objects by controlling the graphical display of the objects to prevent their inter-penetration. Our manipulation aid method makes use of the intuitive phenomenon of magnetic attraction to constrain the degrees of freedom of an object during collision with other objects. The manipulation aid does not require predefined properties to be attached to objects because it can dynamically constrain objects by using collision output face pairs from a real-time collision detection algorithm. Several experiments were conducted to test the efficiency of the manipulation aid against other methods such as without interface aid, with collision cue, with real objects, and with different input devices. In all these cases the results show that the manipulation aid is able to assist the user to manipulate virtual objects more efficiently and with more ease. The proposed manipulation aid is shown to be intuitive, real-time, accurate, efficient, compatible with user input, able to handle different object interactions and independent of input device.

Chapter 1

Introduction

Virtual reality techniques can be used to provide an intuitive and sophisticated user interface using human spatial perception. Limitations in computational power, however, make it difficult to develop a perfect virtual environment, so a simple task in a real environment often becomes an operation requiring skill in a virtual environment. For example, the easy task of putting a block on a table is difficult in the simplest virtual environment in which there are no constraints on the position of the object. In order to accomplish even an easy task in a virtual environment efficiently, as it likely would be done in the real world, it is considered necessary to calculate and simulate such factors as the avoidance of intersection from the test of interference among virtual objects, the fall of virtual blocks caused by gravity, and friction between blocks and table.

There has been considerable study in simulating the interaction of objects in virtual reality. There are force feedback techniques such as contact sensations for finger [IS93], master station with force functions for telerobotics [KKT92, KTT94, SP94], force sensations for hand [Iwa90], and so on. But these all require special force feedback hardware devices which operate under special configurations. There are also physical simulations including contact analysis [BV] and dynamical simulation [Bar89]. Visual techniques include CAD-based applications [Ven93, Bie90], synthetic fixtures for robotic tasks [SP94], and operator assistance simulating magnetic attraction [CTKK93]. These examples are limited to a single level of constraint complexity or special functions are attached to the objects in advance. For example, the operator assistance method described in [CTKK93] requires attracting faces to be predefined in order to achieve real-time performance. Moreover it considers only single face-to-face interactions and the number of attracting faces is limited to only one for each object. Therefore it is not flexible and cannot apply to a variety of tasks employing multiple objects with complicated shapes. Synthetic fixtures described in [SP94] can handle different object feature interactions at multiple points, but the interaction properties must be attached to the appropriate locations in advance. This method is specific to robotic tasks whose motions are typically predetermined, hence it is not general enough for manipulation of arbitrary objects.

In this report, we describe an intuitive method to assist the user to manipulate objects in a virtual environment without giving special properties to objects in advance and which can handle multiple levels of interactions. The method uses constraints among object features that are dynamically selected while the user manipulates the object. By constraining multiple features during manipulation, the proposed method can improve the accuracy and efficiency of the virtual task. Through several experiments and extensions of the basic method, the proposed manipulation aid is shown to be intuitive, real-time, accurate, efficient, compatible with user input, able to handle different object interactions and independent of input device.

This report is organized into seven chapters. Chapter 2 describes the general method for object manipulation using simple constraints among object faces. Chapters 3 to 5 present three experiments and their results using the manipulation method described in Chapter 2. Chapter 6 contains extensions of the basic face constraints to include other object features and handling of special conditions. Suggested future work and conclusion is given in Chapter 7. Appendix A contains some matrix mathematics.

Chapter 2

Manipulation Aid using Basic Constraints

In this chapter, we describe the proposed Manipulation Aid for basic constraints among object faces. First, we discuss a general scenario for object manipulation and two ways to assist the user. Subsequent sections describe the types of constraints and how they operate in a virtual environment.

2.1 Object Manipulation

Consider a general sequence of object manipulation to be the following 4 steps:

1. grasp the target object
2. move the object to the destination space
3. adjust the precise position and orientation
4. release the object.

For example, when a user places a block on the table in real environment, the degrees of freedom (DOF) of object manipulation which the user adjusts during steps (2) and (3) are 6 and 3, respectively. However, in a simple virtual environment in which no constraints exist on the position of the object, even the manipulation of step (3) requires 6 DOF.

One useful way for providing a natural user interface in a virtual environment is to restrict the DOF concerning the motion of objects or the user's hand. Two main ideas exist to restrict the DOF. The first is to restrict the DOF of the user's hand motion with devices such as force feedback tools which generate a reaction between two faces touching each other. The second is to restrict the DOF of the object motions without restricting the motion of the user's hand. In the first case, the user must be equipped with special hardware (force feed-back devices) capable of generating an accurate reaction to restrict his hand motion or between two faces touching each other. On the other hand, a simple configuration is sufficient for the second one. However, careful verification is necessary, because the user may have a sense of incompatibility caused by a difference between visual feedback and motor control. Our proposed manipulation aid is based on the second approach.

2.2 Basic Constraints among Faces

Suppose a simple task is to place blocks on a table. The block has six planar sides which are connected perpendicularly. We consider the following three situations.

One Face Constraint When a block is placed on the table as shown in Figure 2.1a, the motion of the manipulated block is constrained by the upper surface of the table. In this case, by constraining one pair of faces, the DOF of block motion is restricted from 6 to 3 (2 translations and 1 rotation).

Two Faces Constraint Subsequently, if the second block is aligned adjacent to the first one on the table as shown in Figure 2.1b, the motion of the manipulated block is constrained by two faces (i.e. the upper surface of the table and a touching face of the first cube). In this case, the block is restricted to 1 DOF (1 translation).

Three Faces Constraint When a block on the table is aligned against two other blocks as shown in Figure 2.1c, the aligned block has no DOF. It is constrained by three faces (i.e. the upper surface of the table and two touching faces of the blocks).

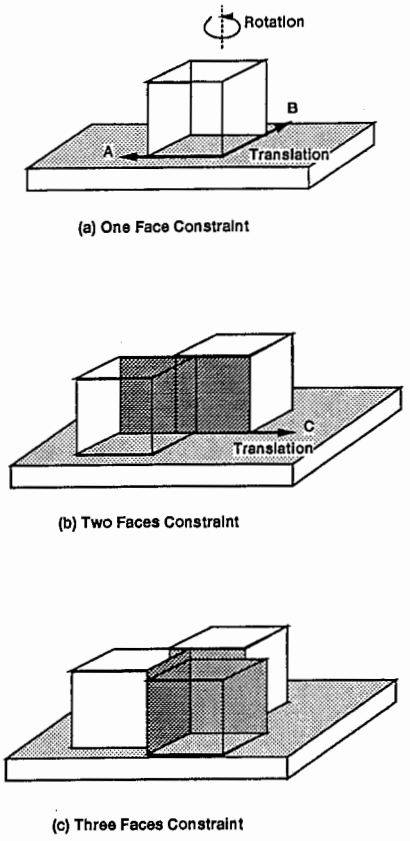


Figure 2.1: Basic constraints among faces

Starting with these simple constraint scenarios, we developed a manipulation aid to assist users to perform such tasks in a virtual environment.

2.3 Method for Manipulation Aid

2.3.1 Assumptions

All objects in the world are modeled as polyhedron (boundary representation). Objects can be concave or convex and are rigid (non-deformable). Objects may be grasped and moved in a non-predetermined way, but it is assumed the speed of motion is sufficiently slow for a real-time collision detection algorithm to report all polygonal collisions. The collision detection algorithm must provide the colliding face pairs between any number of objects. Only the object currently grasped by the user can be assisted.

2.3.2 Outline of Method

Figure 2.2 shows the relationship of the manipulation aid algorithm in the execution flow of a virtual reality system. Our manipulation method is a visual technique which restrict virtual object motion but not user's hand motion, therefore the distinction between actual and displayed object position is made. The actual object position is controlled by the user's hand while the displayed object position is what the user sees. The key to the method is how the manipulation aid modifies the object position to provide intuitive and compatible assistance for the user during object interaction.

In order to determine how an object should be constrained, accurate collision data is required. The efficient collision detection algorithm [SKTK95] is used. This algorithm detects colliding pairs of faces in real-time for three-dimensional graphical environments where objects are undergoing arbitrary motion. The algorithm can be used directly for both convex and concave objects.

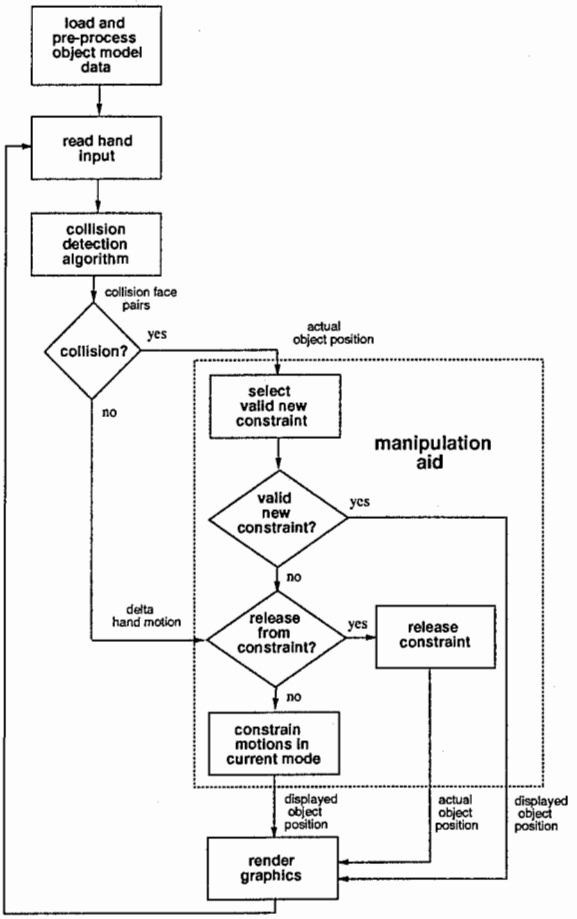


Figure 2.2: Relationship of manipulation aid in execution flow

The manipulation aid algorithm consists of three parts: constraint selection, constrain object motion, and constraint release. The next sections describe how these work.

2.3.3 Constraint Selection

An object with a complicated shape may have a number of colliding face pairs detected by the collision detection algorithm. By examining the geometry between the collision pairs and speed of interaction, we can guess the intention of the user and dynamically select the best faces to constrain. Since we are only considering face to face interaction at this point, we can apply several conditions to each colliding face pair to reduce the number of possible pair candidates. These conditions are:

1. at least one face in the pair is moving
2. angle between the two face normals is more than 120 degrees
3. ratio of overlap area between the two faces to the smaller face is more than a certain threshold

For face pairs which satisfy all of the above conditions, an attraction value is calculated using the following equation.

$$attraction = rC_r + vC_v + dC_d \tag{2.1}$$

where,

r is the rotation angle between the two face normals,
v is the angle between the moving object's velocity vector and the colliding face of the target object,
d is the distance of the moving face projected onto the target face,
C_r, C_v, and C_d are parameter coefficients.

The collision face pair which have the highest attraction value is selected for constraint.

2.3.4 Constraining Object Motions

If a valid face pair is found as described in the previous section, the displayed object position must be modified to reflect the new assisted position. The grasped object is translated to the surface of the selected face and a rotation matrix is applied to make the chosen face pair parallel. The distance to translate is calculated by projecting the center of the moving face onto the target face. The rotation matrix is found by:

$$M_{rot} = T(vtx)A(\theta, \vec{v})T(-vtx) \tag{2.2}$$

where,

M_{rot} is the rotation to apply to the current object position,
vtx is the center of gravity of the moving face, and
$A(\theta, \vec{v})$ is the matrix for making the moving face parallel to the target face, $\vec{v}$ is the normal vector orthogonal to the two face normals, θ is the angle of rotation found by the dot product of the two face normals (see Appendix A for details of the matrix mathematics.)

After the object position is constrained onto the target face, further motion of the object is restricted by the currently constraining faces until the user deliberately exits from constraint. The method for release from constraint is described in the next section. The rest of this section describes how to constrain object motion for simple single and multiple face constraints.

In **One Face Constraint** mode, the object has one constrained face. The motion is then constrained to 3 DOF: 2 translations on the plane of the target face and 1 rotation around the normal on that face (see Figure 2.1a). The constrained translation is determined by finding the component of the change in translation of the manipulated object projected onto the plane of motion. The equation is:

$$\vec{X} = s\vec{A} + t\vec{B} \tag{2.3}$$

where,

$\vec{X}$ is the absolute position,
$\vec{A}$ and $\vec{B}$ are the orthogonal vectors on the plane, and
s and t are components of the Δ translation projected onto the plane.

The constrained rotation angle is determined by using the Δ angles of the manipulator device and the normal in the direction of rotation:

$$ang = n_z\Delta a + n_y\Delta e + n_x\Delta r \tag{2.4}$$

where,

ang is the constrained rotation angle about the direction of rotation,
a, e and r are azimuth, elevation, and roll, respectively, from the hand orientation, and
n is the unit normal in the direction of rotation.

In **Two Faces Constraint** mode, the object has two constrained faces, and the motion is constrained to 1 DOF—translation along the two faces (see Figure 2.1b). The vector of motion is the vector orthogonal to both assisted face normals. The above equation is simplified to

$$\vec{X} = s\vec{C} \tag{2.5}$$

where,

$\vec{C}$ is the vector of motion, and
s is the component of translation on the plane.

In **Three Faces Constraint** mode, the object has three assisted faces, hence the object is in a corner and there is no DOF while the object remains constrained to all three faces (see Figure 2.1c).

2.3.5 Release From Constraint

Since the manipulation method uses an intuitive "magnetic" attraction to constrain an object, the method for release from a constraint should be intuitive also. We call the release action "unsnap" for obvious reasons. Two conditions allow a constrained object to unsnap from a particular face: overlap ratio and distance from face. A release detected by either one of these conditions is sufficient for unsnap.

Overlap Ratio

Figure 2.3 shows the region of overlap between a pair of constrained face. Checking the overlap area ensures that the object is constrained only when it is still touching another object. To take into account different object sizes and scaling, the overlap ratio is used. The Overlap Ratio is the intersection area of the assisted faces over the area of the smaller face in the pair. Hence the overlap ratio is' a percentage of the smaller face area. The following condition is tested:

when $overlap\_ratio < overlap\_threshold$
unsnap from face

overlap_threshold value should be a value near zero with hysteresis to prevent object snap and unsnap in borderline cases.

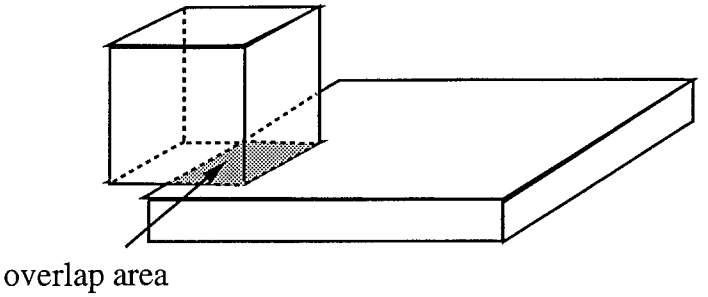


Figure 2.3: Overlap area between constrained faces

Distance from Face

We have discussed how the displayed object position is modified from the actual position using various translation and rotation constraints. And we saw that the delta movement of the actual position (from hand input) is used to determine the constrained movement. Here, we look at how the distance of the actual position from the constrained face can be used to unsnap from a surface.

The Distance from Face is the distance of the current hand position to the constrained face of the target object (see Figure 2.4). Checking this distance allows the user to deliberately unsnap from a face by pulling far enough away.

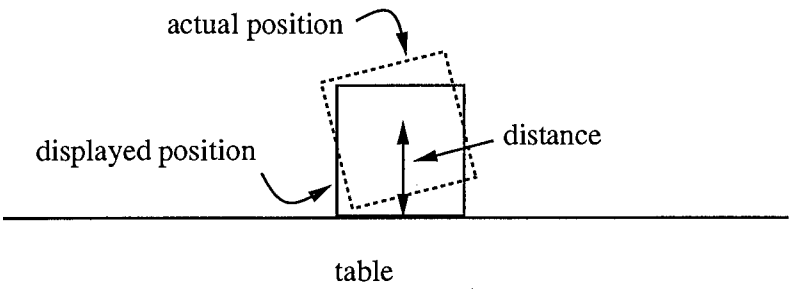


Figure 2.4: Distance from constrained face

when $distance > dist\_threshold$
unsnap from face
where *distance* is the distance from current hand position to constrained face of target object, *dist_threshold* is a dynamic threshold calculated below.

To better simulate the magnetic property of our constraint method, we modeled a simple "magnet" which has equal magnetic fields perpendicular to the surface, as shown in Figure 2.5. Hence, we assume the greater the contact area, the more force is required to pull the object away from the surface, thus a greater distance is required to unsnap from the surface. We call this a dynamic threshold because it changes with overlap area.

$$dist\_threshold = k\sqrt{A} \tag{2.6}$$

where,

A is the overlap area, and
k is a user-adjustable parameter

The parameter k is adjusted accordingly to provide the best feeling of magnetic behavior in a simulation.

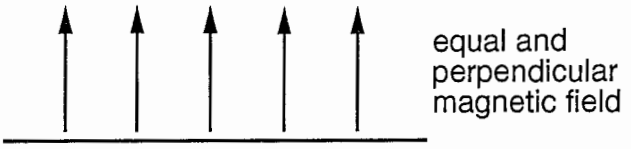


Figure 2.5: Simple model of magnetic attraction on a plane

Chapter 3

Experiment for Basic Constraint

We conducted several experiments to determine the efficiency of the Manipulation Aid method for the basic face-to-face constraints described in chapter 2. The experimental setup, method and results are discussed in this chapter.

3.1 Experimental Setup

Figure 3.1 shows the hardware configuration of our experimental system. All input and output devices and sensors are controlled by an SGI ONYX workstation. A 70-inch CRT projector displays position tracked stereoscopic images. User eye position is derived from a 6 DOF magnetic sensor attached to LCD shutter glasses used for stereo viewing. Accordingly, the system can present non-distorted images with depth sensations and motion parallax. The user can grasp and manipulate objects using the ADL-1_{TM}, a 6 DOF mechanical tracker connected to a serial port of the workstation.

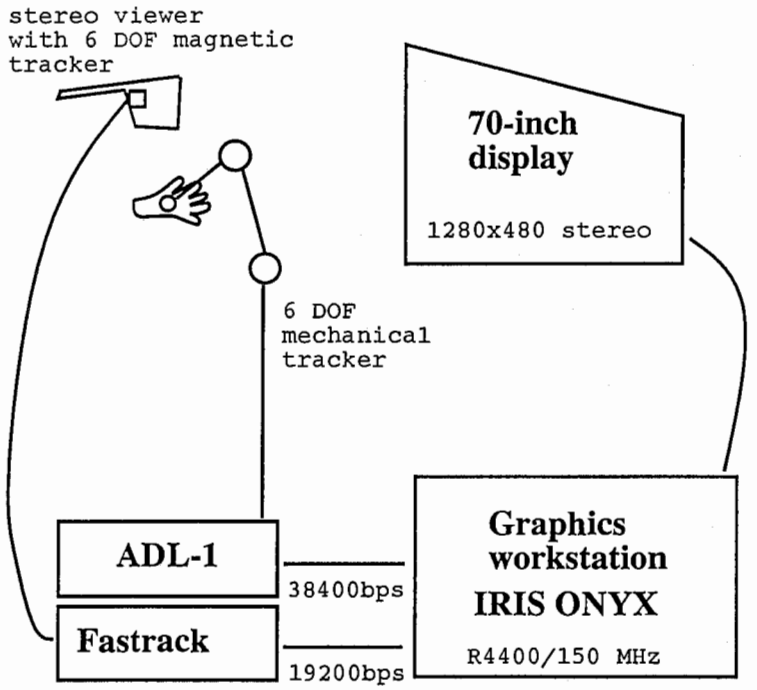
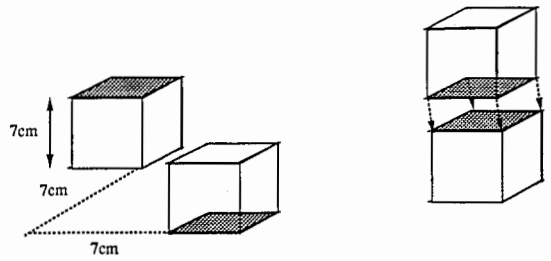


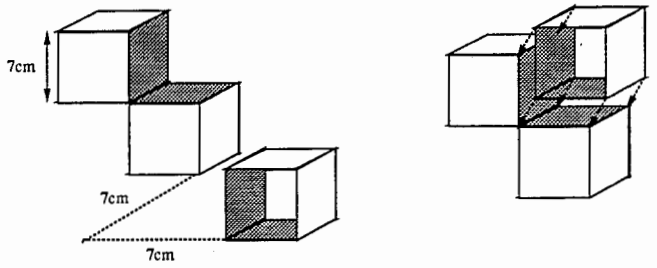
Figure 3.1: Hardware configuration for basic constraint experiment

3.2 Experimental Method

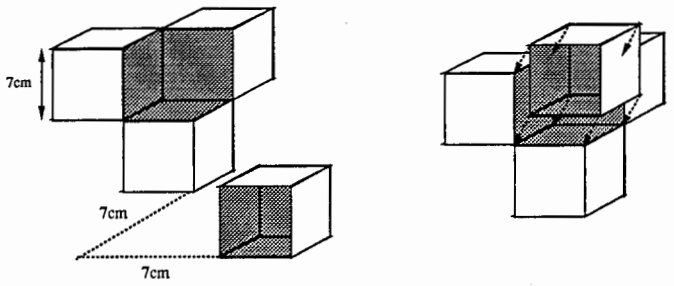
Three tasks, Tasks A, B and C, were designed for testing one, two and three face constraints, respectively. Figure 3.2 shows the task configurations.



(a) Task A - One Face Constraint



(b) Task B - Two Faces Constraint



(c) Task C - Three Faces Constraint

Figure 3.2: Experimental tasks

Task A: One Face Constraint — consists of two 7cm cubes, initially separated by 14cm (between centers) in horizontal and forward directions. The task is to place the front cube on top of the other, aligning all four corners of the faces (see Figure 3.2a).

Task B: Two Faces Constraint — consists of three 7cm cubes, two cubes sharing an edge and forming a right angle surface, and the third cube initially 14cm away from the lower cube as above. The task is to place the third cube into the right angle surface formed by the previous two cubes, aligning all six vertices (see Figure 3.2b).

Task C: Three Faces Constraint — consists of four 7cm cubes, the same as for Task B but with a fourth cube adjacent to the previous two cubes forming a right angle corner. The task is to place the third cube into this corner, aligning all seven vertices (see Figure 3.2c).

To determine the effectiveness of the constraint method, data for comparison were collected for the following three modes.

Mode 1: No aid and no color cue — no user interface aids

Mode 2: No aid with collision color cue — no manipulation aid, but the colliding face patches for both objects change color

Mode 3: With aid and color cue — with manipulation aid, and the constrained faces for both objects change color

Five young subjects participated in the experiments, their ages ranging from mid-twenties to early thirties. There were three males and two females; four had technical and/or VR experience, and one had no technical experience.

The experiments consisted of two stages, accuracy evaluation and efficiency evaluation. Prior to stage 1, the subjects practiced using the system in the various modes to become familiar with the virtual environment behaviors and system hardware in the experiment. During the trials for both stages, subjects took short rest breaks as were required. The two stages are described in details in the next sections.

3.2.1 Method for Accuracy Evaluation (Stage 1)

The purpose of Stage 1 is to compare the accuracy of object placement in each of the three modes described previously. Subjects were asked to complete Tasks A and B as accurately and quickly as possible in each trial. Ten trials were done for each mode and task. Three measurements were taken: *Completion Time*, *Distance Accuracy* and *Angular Accuracy*.

Completion Time is the time between object grasp and release, as measured by the computer real-time clock. Hence the task consists of one object grasp and release only; that is, the user cannot re-adjust the object after it has been released. *Distance Accuracy* is the sum of the distances between the four, six or seven vertices of the cubes in Tasks A, B and C respectively. *Angular Accuracy* is the sum of the three angle errors (azimuth, elevation and roll) from the target position.

3.2.2 Method for Efficiency Evaluation (Stage 2)

The purpose of Stage 2 is to compare the time efficiency for task completion given a certain accuracy level. We used the distance accuracy as the accuracy criteria since the requirement levels were chosen relatively low (accurate) and the angles were small. In order to compare the results of all three tasks, we used the distance error per vertex average instead of the sum. We call this average vertex error the *accuracy requirement level*. The smaller the accuracy requirement, the more difficult the task becomes. When the object is placed such that the distance accuracy falls below the required level, the object changes color indicating task completion. The subjects did Tasks A, B and C each for 3 accuracy levels (4mm, 3mm and 2mm distance error per vertex average) in Modes 1 and 3. The task completion times were measured for ten trials for each combination of level, mode and task.

3.3 Results

Results for the two stages of experiments are discussed in the following sections.

3.3.1 Accuracy Evaluation (Stage 1) Results

The distance accuracy versus completion time for Task A of one subject is shown in Figure 3.3. The distance errors and completion times with manipulation aid (mode 3) are, on the average, less than without (mode 1 and 2). Comparing modes 1 and 2 reveals that mode 2 took slightly longer time without much improvement in accuracy. This result corresponds to the comment made by several subjects that the collision color cue did not aid but rather distracted their object alignment. Graphs of average times shown in the next section also have similar trends.

Figure 3.4 shows the same plot as above but for Task B. Again, the distance errors and completion times are less for mode 3 than the other two modes. Compared to Task A, there is a bigger separation between mode 3 data group and the other groups, which indicates the constraint method provides more gain for Task B than A. This result is expected since Task B is more difficult than A, but in constraint mode Task B has fewer DOF than Task A. The gains for both tasks are discussed in more details in the next section.

The next two graphs, Figures 3.5 and 3.6 show the angular accuracy versus completion time from the same trial data of the previous two graphs. The separation between with aid and no aid data is better for angular accuracy than distance for both Tasks A and B. This is because an angular offset will cause a distance error while there can be a distance offset without angular errors. Therefore Task A, which has more angular errors than Task B, also has higher distance errors. For Task B, the angular errors are virtually zero because the only DOF remaining is a translation. Again, we see that modes 1 and 2 have almost the same amount of error despite the collision color cue.

The scatter plots for the other 4 subjects show similar results as the four previous graphs. The average distance accuracy of each subject can be seen in Figures 3.7 and 3.8 for Tasks A and B, respectively. All subjects had smaller distance errors with manipulation aid than without and the majority of the subjects had

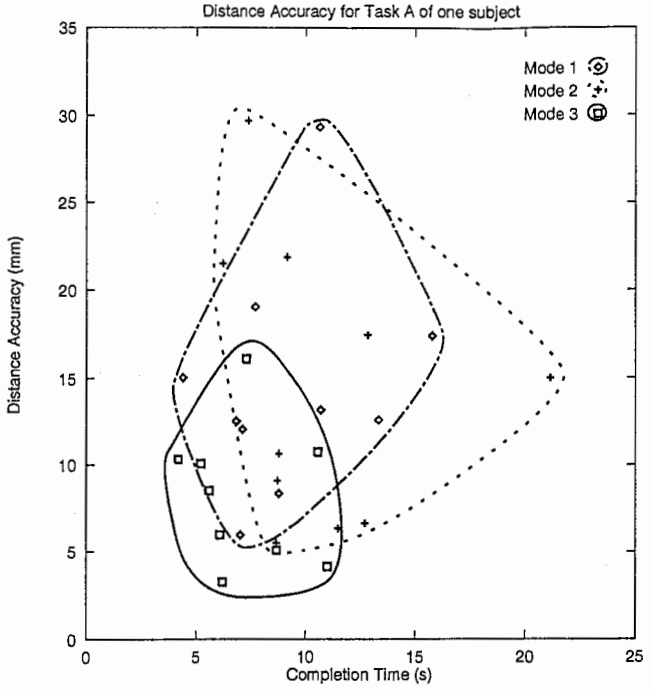


Figure 3.3: Distance accuracy for Task A of one subject

worse results using the collision cue than without. The errors were higher for subject 5 because this subject does not have technical experience compared to the other subjects.

Figures 3.9 and 3.10 show each subject's average angular accuracy for both tasks. Comparing the distance and angular averages for the three modes, the gain in angular accuracy is more substantial than distance with manipulation aid. The angular errors of Task B with aid are virtually zero because there are no rotational DOF in two faces constraint.

Tables 3.1 and 3.2 show the average ratios and gains of the measured data in Tasks A and B using the manipulation aid. The second column is the ratio of performance with aid (mode 3) to without aid (mode 1), and the third column is the gain calculated by: $gain = (1 - ratio) \times 100\%$. The gains are higher for Task B because this task is both more difficult without constraint than Task A, while at the same time it is easier with constraint. The angular gain of Task B should be 100% theoretically, since in constraint mode only 1 DOF remains and it is a translation. With the manipulation aid, these experiments show that on average, the distance accuracy can be as much as 60% better with a time saving of up to 41%.

Table 3.1: Average Ratios and Gains for Task A

	Ratio(with/without)	Gain(%)
Dist Accuracy	0.65	35
Ang Accuracy	0.29	71
Completion Time	0.79	21

Table 3.2: Average Ratios and Gains for Task B

	Ratio(with/without)	Gain(%)
Dist Accuracy	0.41	59
Ang Accuracy	0.01	99
Completion Time	0.59	41

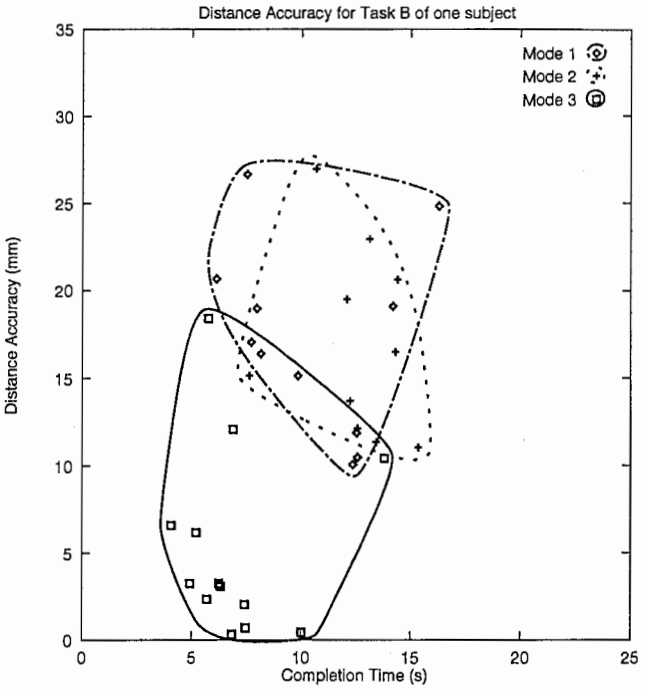


Figure 3.4: Distance accuracy for Task B of one subject

Figure 3.11 combines Tables 3.1 and 3.2 with some theoretical values to obtain the gain versus the number of constrained DOF. (Task A has 3 constrained DOF, B has 5 and C has 6.) Theoretically, Task C should have 100% gain because there are no DOF remaining, and zero assisted DOF corresponds to zero gain. The gains in Table 1 and 2 are plotted for distance and angular accuracies with standard deviations. The large standard deviation may be due to the small number of subjects and the difference in capabilities. Since there are only 2 experimental points with 2 theoretical points for each curve, this graph shows just the rough tendencies for estimating the amount of gain for a certain number of constrained DOF with this method.

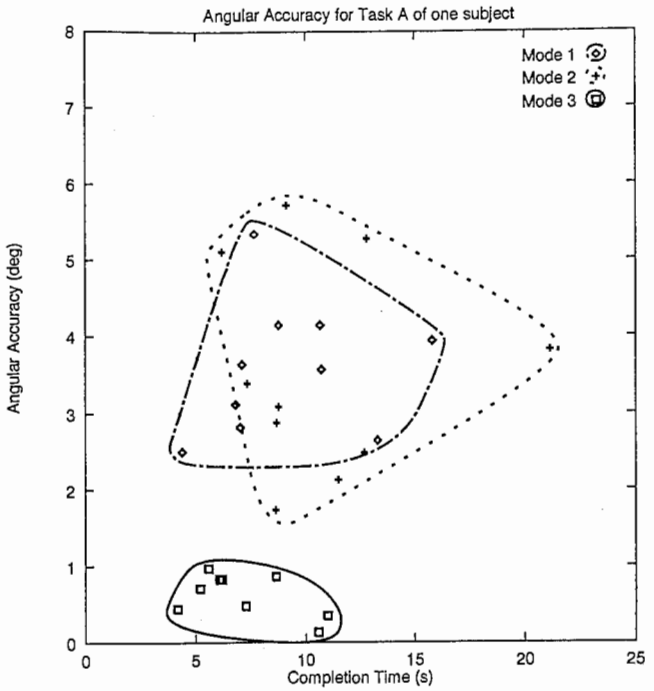


Figure 3.5: Angular accuracy for Task A of one subject

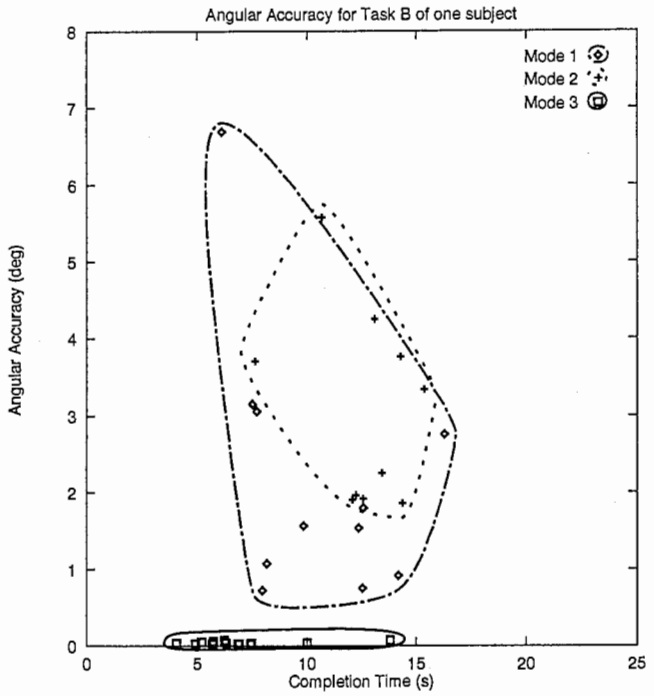


Figure 3.6: Angular accuracy for Task B of one subject

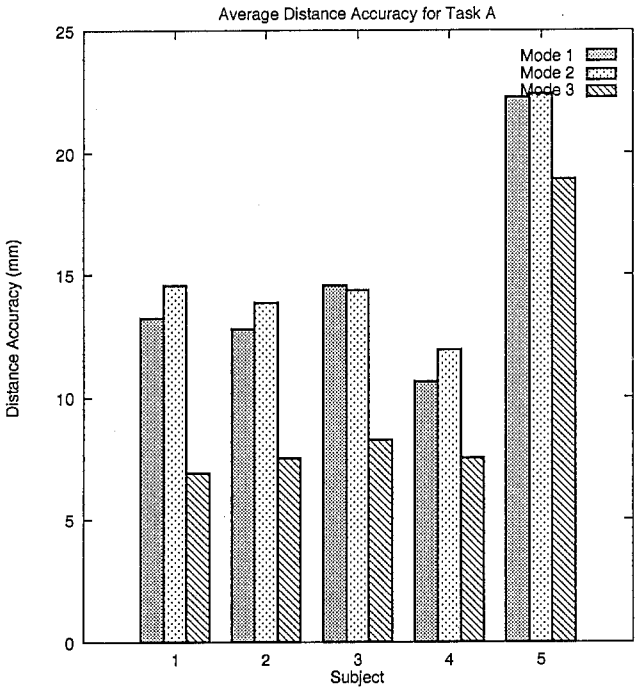


Figure 3.7: Average distance accuracy for Task A

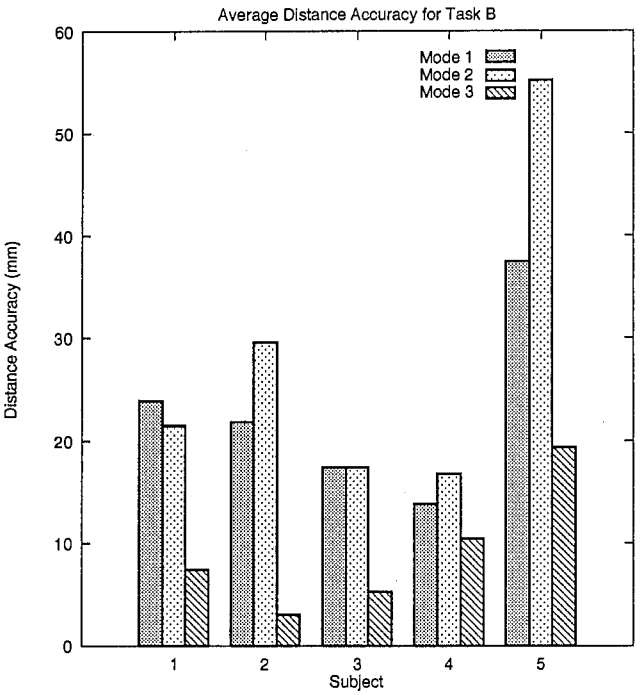


Figure 3.8: Average distance accuracy for Task B

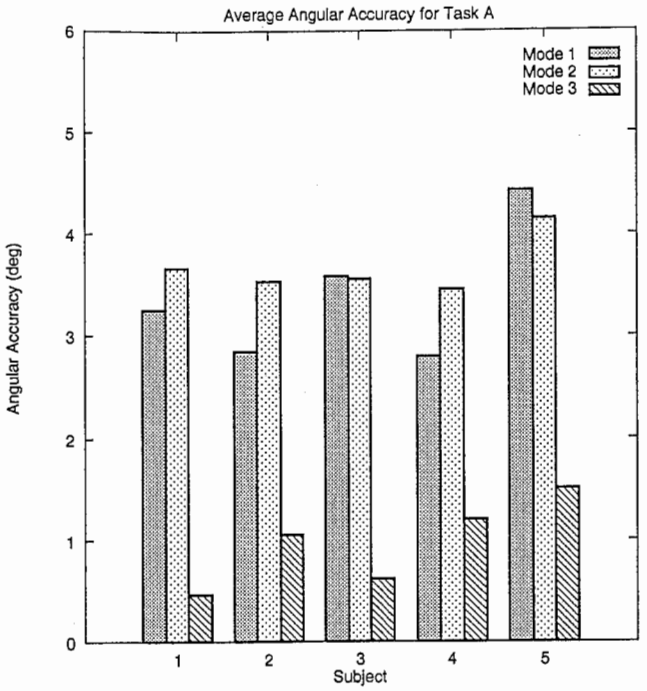


Figure 3.9: Average angular accuracy for Task A

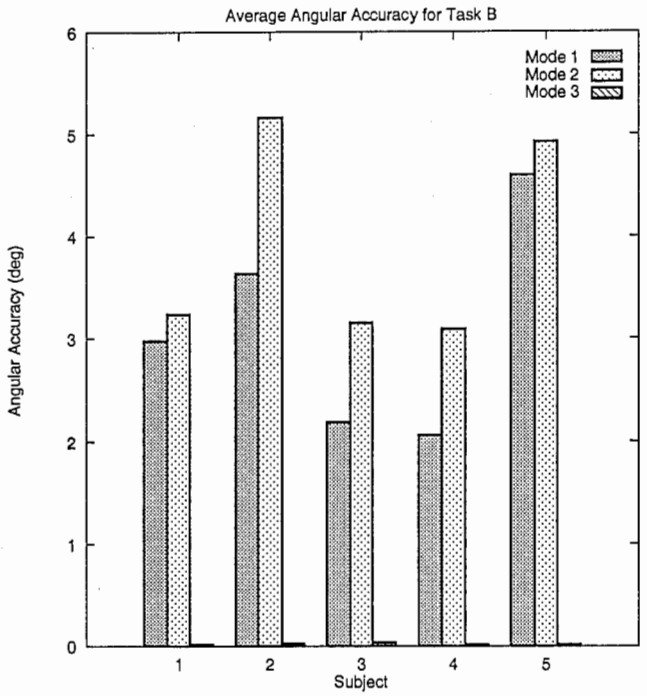


Figure 3.10: Average angular accuracy for Task B

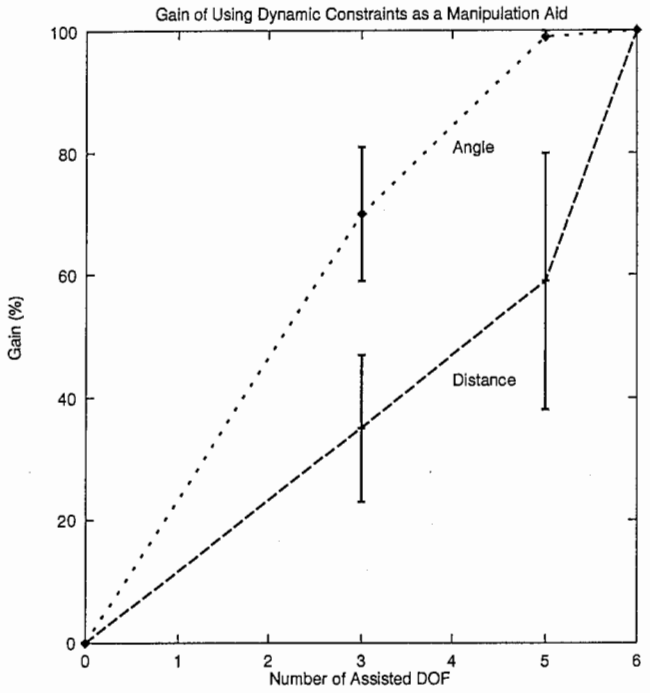


Figure 3.11: Gain of using dynamic constraints as a manipulation aid

3.3.2 Efficiency Evaluation (Stage 2) Results

Since each subject had different skills in using the system, the completion times had to be normalized before the average completion time (over all subjects) for each combination of level, mode and task, can be determined. For each subject, the times for each combination are first averaged. These averages are normalized using the equation: $t_{navg} = \frac{(t - t_{minavg})}{(t_{maxavg} - t_{minavg})}$, so that the normalized averages are in the range [0:1] for each subject. The normalized times are then averaged over the five subjects to obtain the average normalized time. Figure 3.12 shows the normalized times for the three tasks in modes 1 and 3 for three accuracy levels. The combination requiring the longest time can be considered as taking 100% time and all other combinations are compared relative to this. From the graph we see that without manipulation aid, the normalized times are always higher in all 3 accuracy levels. As the accuracy requirement increases, the separation of the mode 1 and mode 3 curves become greater. With manipulation aid, all three tasks take less than 15% of the maximum time; Task B and C take less than 10% due to less DOF.

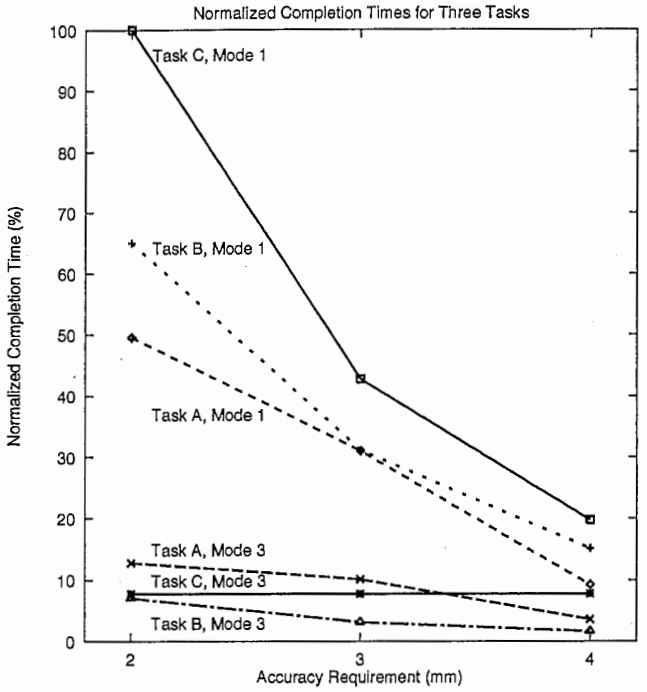


Figure 3.12: Normalized completion times for three tasks

The time gain of using the manipulation aid is calculated using: $gain = 1 - \frac{t_{avgwith}}{t_{avgwithout}}$. Figure 3.13 shows the time gain for the three tasks and accuracy levels. The gain increases as the task becomes more difficult. The two-face constraint task (B) shows a 90% gain for all levels, indicating that without constrain this task was difficult to do in all levels. The gain of the three-face constraint task (C) however, was surprisingly lower for less accuracy; this may be because the third cube helped the user align the cube better. For all three tasks there is 75% or more gain in time for the most difficult level and over 60% for the easiest level.

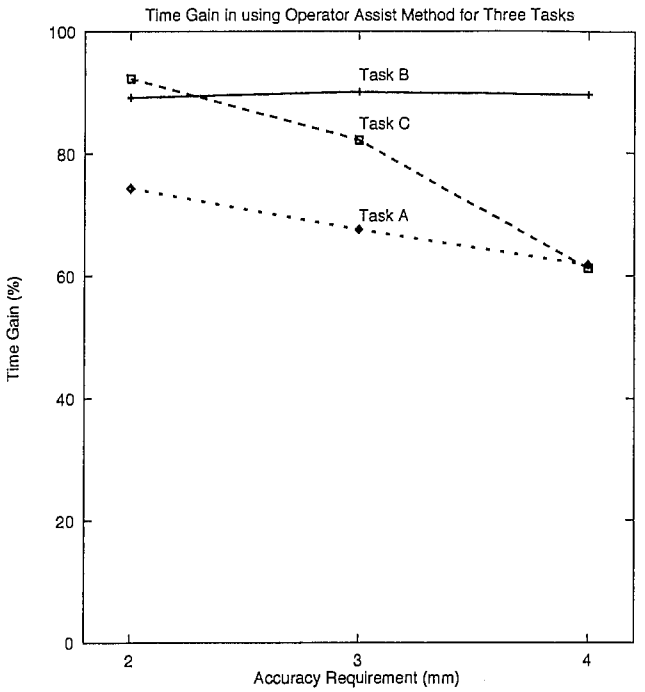


Figure 3.13: Time gain using manipulation aid for three tasks

One observation in this last stage of the experiment provides a qualitative measure of usefulness for the face constraint method. The subject with little or no technical experience had a very difficult time with the trials of highest accuracy for both Tasks B and C without manipulation aid. The subject took a painstakingly long time to meet the accuracy requirement and was frustrated at the difficulty of precise alignment without manipulation aid. From this observation we cannot conclude definitely that the frustration level was high without manipulation aid for non-technical users because we had only one such subject in this experiment; further experiments are required. However the other four subjects with technical experience also expressed some frustrations during the precise placement tasks of B and C. In summary, the users were satisfied and confident with the manipulation aid, but were frustrated and anxious without it, especially for difficult tasks.

3.4 Conclusion

In this chapter, we presented a series of experiments to determine the efficiency of using dynamic face constraints to assist the user in manipulating virtual objects. The results show that this constraint method provides significant gains in distance and angular accuracies as well as task completion time, making this method a useful manipulation aid for a virtual environment.

The constraint method shows higher gains when more faces are constrained. In a complicated virtual environment with many objects, precise placement of objects become increasingly difficult without constraint, but easier with the constraint method. The constraint method also shows higher gains when a high level of precision is required.

It was thought that a collision color cue may assist the user when no other interface aid is available. The results show that this was not the case. Instead the collision cue distracted the user's attention during alignment and caused the results to be poorer than without collision cue. Hence a collision color cue without constraints is insufficient aid for precise manipulation of virtual objects.

Observations for the qualitative effectiveness of the constraint method indicate that users showed immediate confidence in using the intuitive manipulation aid, and at the same time frustration when there was no aid for precise alignment tasks. Hence the constraint method can not only provide better accuracy and times but can also reduce the work strain on the user.

Even in these simple tasks, the constraint method enables the user to manipulate and place objects more precisely and in less time. It is reasonable to predict that this method will provide even better assistance in a more complicated environment.

Chapter 4

Experiments for Virtual and Real Task

We designed two experiments to compare the task of constructing a simple toy in a real versus virtual environment with two manipulation devices. In the first experiment, we used a 6-DOF mechanical arm (ADL-1_{TM}) as the manipulation device in the virtual environment. And in the second experiment, we used a 6-DOF magnetic sensor (Fastrak$_{TM}$) attached to a lightweight block. For the real environment, we used our hand to manipulate a set of real blocks from which the virtual objects were modeled.

4.1 Experimental Setup

The setup for the first experiment is the same as the Basic Constraint Experiment in Chapter 2, and for the second experiment, the ADL is replaced by Fastrak. Figure 4.1 shows the hardware configuration.

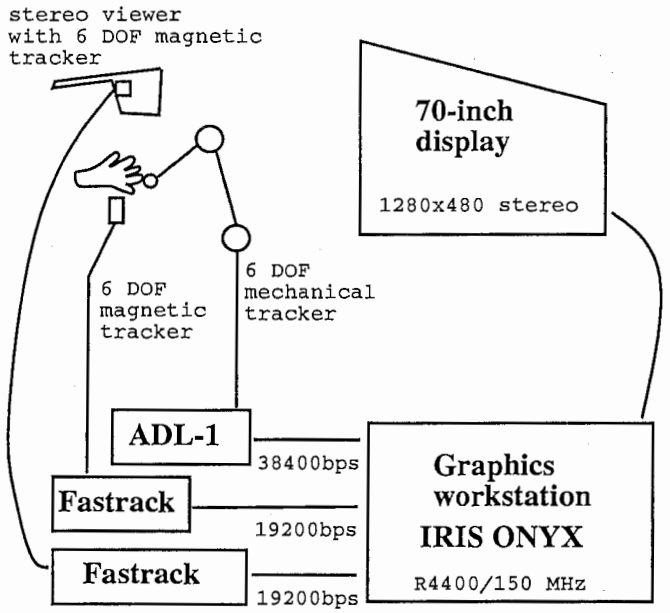


Figure 4.1: Hardware configuration for virtual and real task experiments

4.2 Experimental Method for Both Devices

The task is to construct a snail using five predefined blocks. Figure 4.2 shows the initial positions of the blocks and Figure 4.3 shows the completed snail. The virtual blocks are modeled to resemble the real blocks in geometry, size and color.

The assembly of the toy snail is carried out in three different modes:

Virtual without aid: assemble virtual blocks without manipulation aid and no collision color cue

Virtual with aid: assemble virtual blocks with manipulation aid and color cue

Real: assemble real blocks

In order to compare the virtual and real manipulations as best as possible, several rules were established for the manipulation of real blocks. These rules are:

- only the thumb and one other finger of one hand are allowed to grasp an object
- turning of the object with fingers is not allowed; instead turn the wrist or arm
- only the grasped object may be touched
- during the time between object grasp and release the subject's elbow or palm cannot rest on any structure (i.e. table, other blocks)
- objects already placed should not be moved while placing other blocks

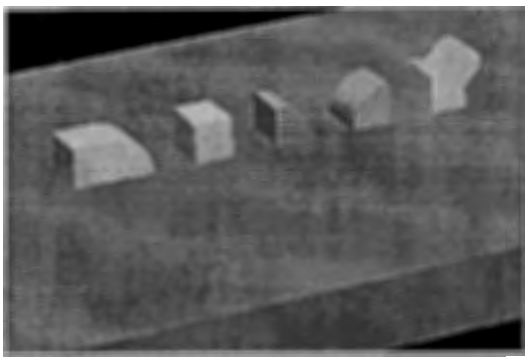


Figure 4.2: Initial positions of blocks for toy snail

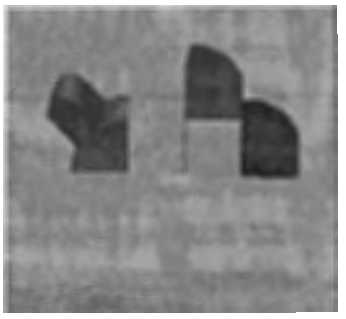


Figure 4.3: Finished construction of toy snail

4.2.1 Method for Accuracy Evaluation (Stage 1)

The purpose of Stage 1 is to compare the distance accuracies and completion times for the three modes. Subjects were asked to build the toy snail as quickly and accuracy as possible. Two measurements were taken: *Completion Time* and *Distance Accuracy*.

Completion Time is the real clock time from grasp of the first block to release of the last block in the assembly sequence. *Distance Accuracy* is the sum of the distance error between all adjacent vertices.

In the real environment, the distance error is difficult to measure precisely because of the imprecise shape of the real blocks. However, rough estimates of several trials yielded errors of less than 2mm per vertex. Hence, we assume a maximum distance error of 2mm per vertex for the real task. In this stage, eight trials were completed in each of the three modes.

4.2.2 Method for Efficiency Evaluation (Stage 2)

The purpose of Stage 2 is to compare the time required to construct the snail within a certain distance accuracy. The accuracy requirement was selected to be 3mm per vertex. Ideally, 2mm per vertex would best correspond to the real task case, but the task becomes considerably difficult for the virtual task without manipulation aid, particularly with limitations of the ADL (which are discussed in the results section). The task is to place each object until the distance error falls below 3mm per vertex, which is indicated by a change of color in the object. In this stage, eight trials were also carried out in each of the three modes and completion times were measured.

4.3 Results with ADL Device

This section presents the results from the first experiment using the ADL device. The results from stages 1 and 2 are given, followed by user feedbacks.

4.3.1 Accuracy Evaluation (Stage 1) Results

A plot of the completion time versus distance accuracy for one subject is shown in Figure 4.4. There are three distinct groups of data, corresponding to the three modes. For the real task, the distance accuracy of all the points has been set to a maximum estimated value, with the actual accuracy somewhere between 0 and the maximum. The maximum error was estimated by assuming a 2mm error per vertex, and multiplying by 14 vertex pairs yields 28mm for maximum distance accuracy for the real task.

From the plot we see the group with manipulation aid is closer to the real task than without; it has distance accuracies in a close range to the real task, but taking slightly more time. The time delay may be due to the nature of a virtual environment and limitations in using a mechanical device instead of hand and fingers. Feedbacks on the ADL device will be discusssed further in Section 4.3.3. As expected, the tasks without manipulation aid had higher errors and completion times.

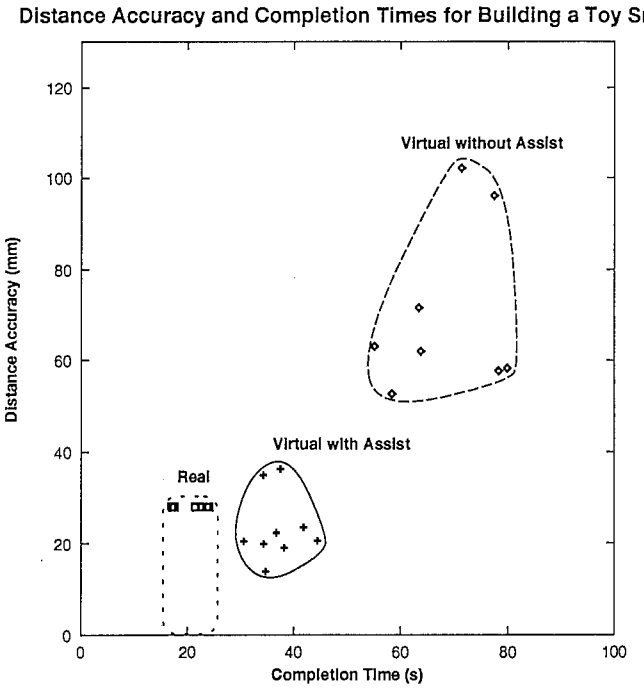


Figure 4.4: Stage one results with ADL device for one subject

4.3.2 Efficiency Evaluation (Stage 2) Results

In this stage, objects were placed within a distance accuracy of 3mm per vertex. The completion times were measured and the average times for each subject are plotted in Figure 4.5. The average times with manipulation aid are generally very close to the real task. The difference may be due to the limitations of the virtual environment, such as graphics resolution and device awkwardness.

To compare the time difference across all subjects, time percentages were calculated for each subject as follows: $percentage = \frac{t_M - t_{realavg}}{t_{realavg}} \times 100\%$, where t_M is the average time for either with aid or without, and $t_{realavg}$ is the average time for real task. Figure 4.6 shows the percentage results. The dashed line separates the two modes. With manipulation aid, all subjects took less than twice the real task time, whereas without the aid took up to five times longer. The most efficient virtual task took only 8% longer than the real task. These percentages indicate that while the virtual task with manipulation aid requires more time than the real task, the additional time is typically less than twice.

4.3.3 Observations and Feedbacks

The purpose of this section is to discuss some of the user feedbacks and observations from the previous experiment to see what factors may affect the performance of the virtual task. The feedbacks also help us to understand more about the experimental results and lead us to further experiments to eliminate potential biases.

1. Object grasping with ADL

   The ADL has a limited roll angle which placed an undesirable constraint on the rotation of objects especially without manipulation aid. Three of the five objects in the snail assembly required rotations. The user had to learn to grasp the objects at certain angles to facilitate the rotations required for placement of objects. This restriction in object grasping may be the main cause for longer completion times and lower accuracy for the virtual task without manipulation aid.

   Another cause for longer time is the grasp misses that happen occasionally when the user attempts to grasp objects too quickly. This occurs due to the difference between virtual object grasping and real object grasping.

2. Object handling with ADL

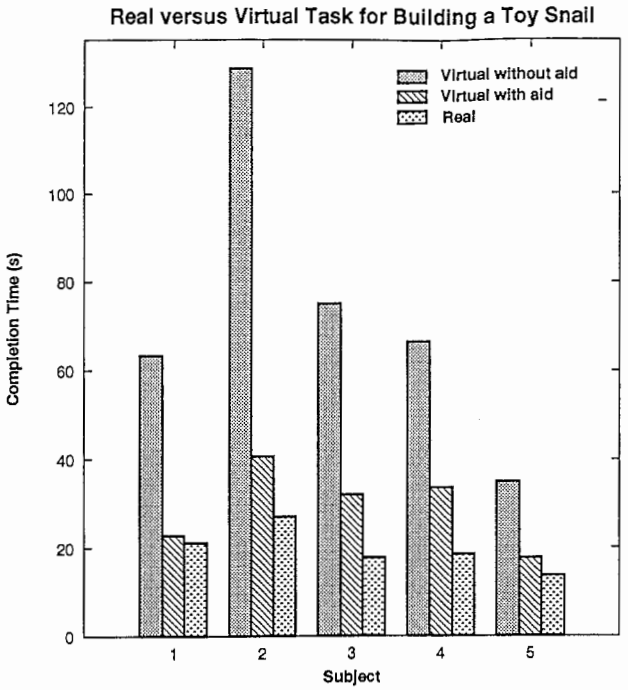


Figure 4.5: Stage two results with ADL device for all subjects

Most of the users found the ADL mechanical arm to be slightly heavy, causing some arm strain and requiring more work compared to the real task. In addition, the grasping of the ADL end effector (EE) is unnatural compared to the grasping of a real block. A real block is grasped with thumb and one other finger, while the ADL EE is held in the palm. The EE can be held with two fingers similar to a real block, but the weight of the ADL would cause too much strain. The unnatural way of manipulating objects using the ADL may account for the longer completion time and lower accuracy.

3. Absence of force feedback

   In the real environment, there is force feedback when a collision with another block occurs. When one object is placed onto another object, the bottom one supports the top one, therefore less work is required in horizontal movement. In the case of the ADL, the user still has to hold the ADL weight without any support or force feedback for movement in any direction.

4. Virtual versus Real Environment

   In the virtual environment, users tend to spend some time observing or mentally evaluating the placement of the previous block before continuing. This does not occur in the real environment. Since the completion time is measured from beginning of the assembly to the end, the slower behavior of the user in the virtual environment results in a longer completion time.

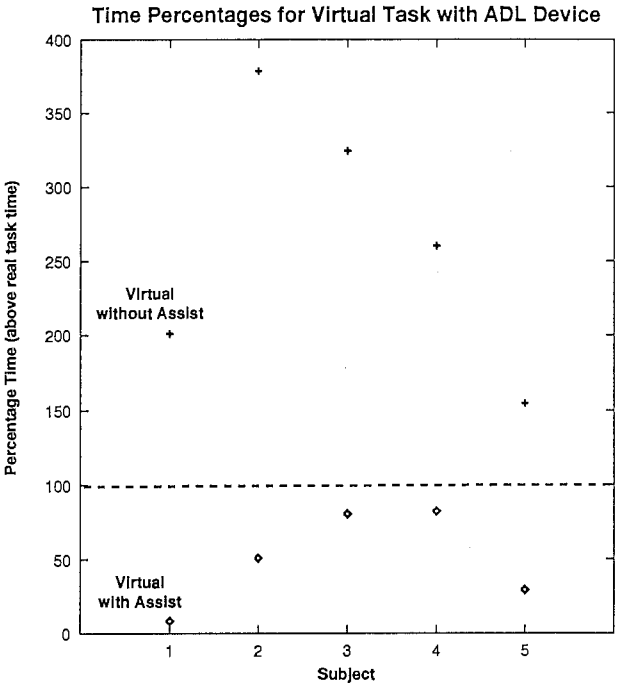


Figure 4.6: Time percentages from stage two with ADL

The user's hand position in the virtual world is sensitive to the movement of the user's head during object alignment. The viewing position of the user is tracked by a magnetic sensor attached to the side of stereographics glasses. When the user adjusts his view, the alignment is often upset, hence resulting in longer task completion time. The quality of 3D effect, contrast and screen resolution of the virtual system may also contribute to more error and time compared to the real task.

The above observations and feedbacks suggest that the ADL device may be a major factor in determining the performance of the virtual tasks. To test this hypothesis and compare the performance of the Manipulation Aid method with another device, we replaced the ADL with Fastrak, a compact and lightweight 6-DOF magnetic sensor which can be held naturally with the hand. The next section contains the results from experiments using this new device.

4.4 Results with Fastrak Device

This section presents the results from the second experiment in the construction of a toy snail using Fastrak instead of the ADL. In this experiment, only stage 2, the task requiring a 3mm distance accuracy, was done. The results for this stage and user feedbacks are given.

4.4.1 Efficiency Evaluation (Stage 2) Results

All of the users found the virtual task to be easier with the magnetic sensor than ADL. The average completion time for each subject are shown in Figure 4.7. For comparison with ADL, the time scale is the same as the graph in Figure 4.5 and the times for the real task are also the same as before. Looking at the two graphs, we can see a significant decrease in task time for the magnetic sensor in the virtual task without manipulation aid. This result agrees with the hypothesis that ADL limitations affect the task time. However, this effect seems to dominate only when manipulation aid is not used, because the times for virtual with aid (middle bar) do not differ by much; that is there was not much gain from using Fastrak over the ADL with manipulation aid. The reasons for this is discussed in the device comparison later in this section.

The time percentages are plotted in Figure 4.8 with the same method as in section 4.3.2. Again, the dashed line separates the two modes, except for one point. With manipulation aid, all subjects took less than about 70% additional time in the virtual task, whereas without aid took 65 to 200% more time. Comparing these results to the ADL graph, we see a gain of a factor of two for the virtual task with manipulation aid, whereas the percentages with the aid are similar. A direct comparison of the ADL and Fastrak is discussed next.

The completion times for both devices are graphed together in Figure 4.9. Three subjects performed better with the ADL while two subjects were better with Fastrak. The time differences are small, with the largest difference being about 4 seconds. There are two possible explanations for the same or slightly worse performance in using Fastrak. The first is that the Manipulation Aid method may be independent of device so that the time required are almost the same for a particular task. The second explanation is the difference of physical configuration of the ADL and Fastrak. The ADL end effector is located directly in front of the user's body, therefore arm motions are relatively small and close to the user. On the other hand, the operating space of the magnetic sensor is further away from the user and closer to the projection screen. This configuration is more natural since the virtual object location corresponds closer to the hand location (but not exactly because of occlusion by hand). Hence, to complete the task with the magnetic sensor, the user has to reach further away from the body, which can account for an increased completion time or counterbalance any time savings over the ADL, in the case with manipulation aid.

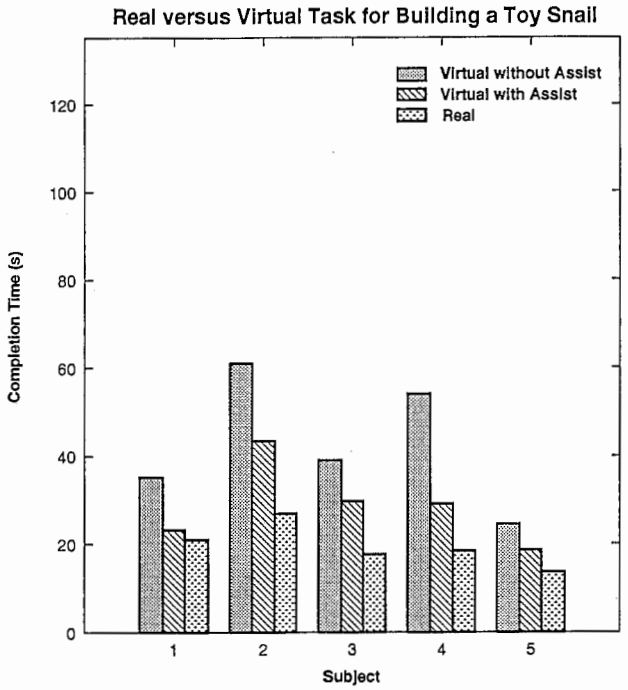


Figure 4.7: Stage two results with Fastrak device for all subjects

4.4.2 Observations and Feedbacks

1. Easier to use than ADL

   All the users felt that the magnetic sensor was easier to use than the ADL because it was lighter and rotations were no longer restricted by the shape of the device, but only in the limitations of the user's wrist and arm. This accounts for time saving when the placement was difficult because manipulation aid was not used.

2. Virtual versus Real Environment

   The method of grasping is the same as for the ADL, hence the problem of grasp misses still exists. Even though the user's hand correspond closer with the graphics, it does not meet exactly, therefore the user requires more time to grasp and place objects than in a real task. When asked about the difference between the real and virtual task with manipulation aid, one subject commented on the absence of sound in the interaction of the blocks.

4.5 Conclusion

In this chapter, we presented two experiments for the construction of a toy snail with simple blocks in both real and virtual environments. The results from both experiments show that the virtual task with manipulation aid is close to the real task in distance accuracy and completion time.

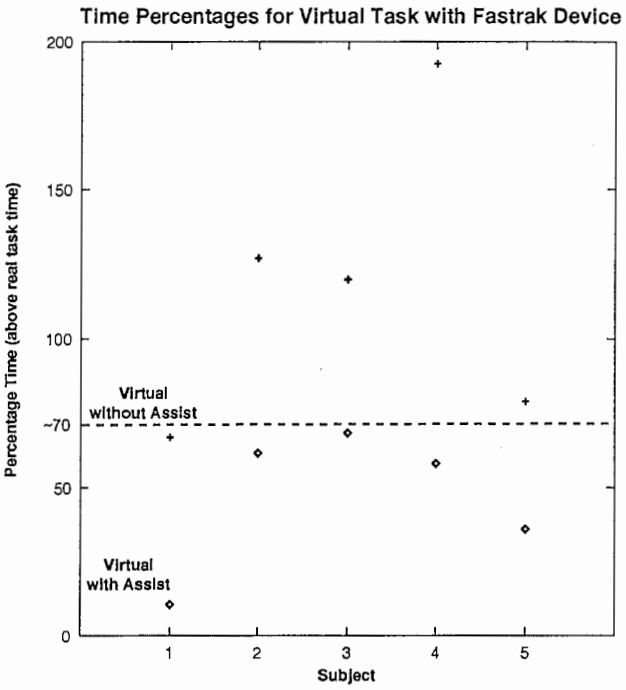


Figure 4.8: Time percentages from stage two with Fastrak

Two manipulation devices were used in the experiments, a mechanical tracker (ADL) and a magnetic tracker (Fastrak). The weight of the ADL as well as its rotation limitations affected the performance in virtual tasks without manipulation aid. Many of the problems with the ADL were removed by using Fastrak, which is lighter and more compact. The magnetic sensor allowed for more natural manipulations similar to handling real blocks. When the performance of both devices are compared for tasks with aid, the completion times are close. This suggests that the Manipulation Aid method is independent of manipulation device, yielding better accuracy and times over a task without such an interface aid.

The Manipulation Aid method has proven to be a valuable tool for the virtual task of constructing a simple toy. With this user interface aid, a virtual task is more natural and simpler, but is still a step away from realism since only visual feedback is exploited. Perhaps when the three senses of sight, touch and sound are all influenced together with the integration of Manipulation Aid, force feedback and sound feedback, then we may approach towards a true virtual reality.

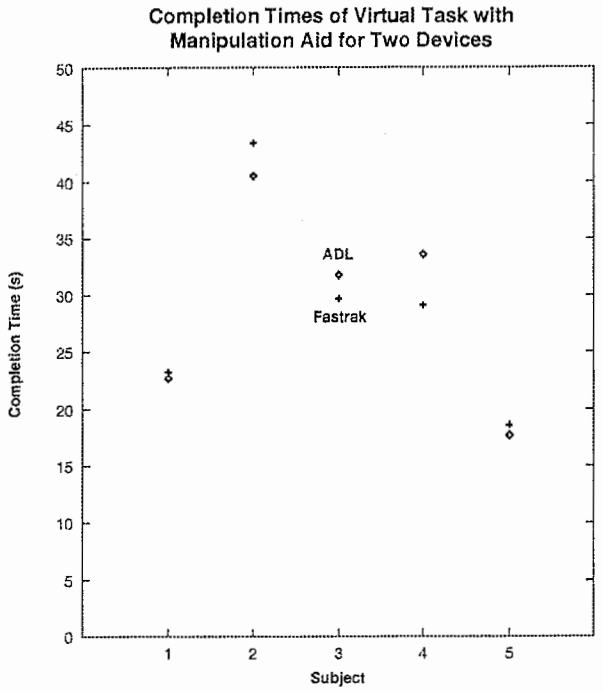


Figure 4.9: Comparing time percentages of two devices

Chapter 5

Experiment for Manipulation Cue

We designed an experiment to test the effect of visual, haptic and audio cues used with our Manipulation Aid method. The cues are used to indicate a change in constraint status. This chapter describes the experiment and the results.

5.1 Experimental Setup

The experimental setup is essentially the same as the previous two experiments except a haptic input device is used and a MIDI sound generator is added for audio feedback. The feature of the haptic device is its Ultrasonic Motor (USM) which can provide high torque and controllable braking torque. We used the latter mode to generate the clutch sensation for the haptic cue. The version of USM device we employed was capable of only 3 DOF (translation).

Figure 5.1 shows the hardware configuration. The USM device is controlled by a PC which is connected to the workstation through Ethernet.

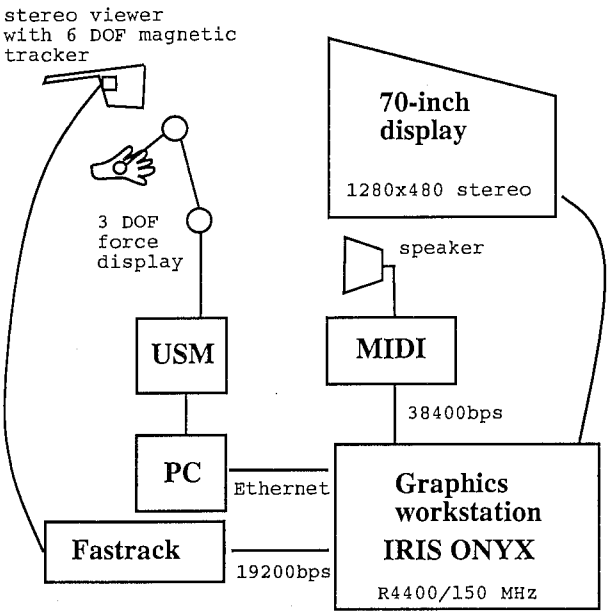


Figure 5.1: Hardware configuration for manipulation cue experiment

5.2 Experimental Method

The three cues we tested were color, haptic and sound feedback. Color feedback is simply a change in color when the face is constrained. Haptic feedback is simulated by stopping the ultrasonic motor of the USM device for a brief period (on the order of 100ms) to generate a clutch sensation. Sound feedback is simulated as a wood knocking sound when faces are snapped together. We tested a combination of these cues in a simple task shown in Figure 5.2.

The objective of the task is to investigate which combination of feedback cues is most efficient with manipulation aid. The task is to pick up object A and touch the corresponding labelled faces on object B in the numbered sequence. Object A must be constrained at each face before proceeding to the next, but alignment is not necessary. After touching all 6 faces, the user finally releases object A on top of object B, and the task time is measured from object grasp to release.

The task is done with five different combinations of cues: none, color only, color and haptic, color and sound, and finally all three. For each mode there were 8 trials. Five subjects took part in the experiment.

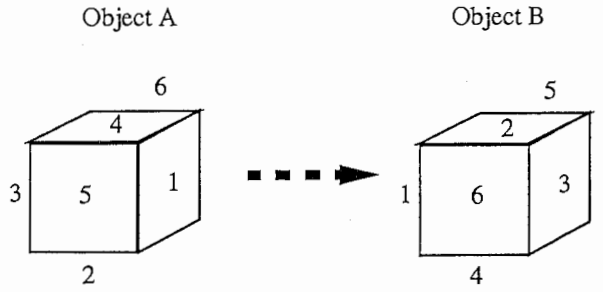


Figure 5.2: Task to investigate manipulation cues

5.3 Results

Figure 5.3 shows the average task time for each of the different cue combinations. As expected, the task took the longest without any cues to indicate when faces are constrained. However the best combination appears to be the combination of color and sound feedback, rather than all three. The reason for this may be due to the clutch behavior used for the haptic cue. The clutch behavior creates a slightly sticky feeling as the ultrasonic motor is braked, hence this may be the cause for increased task times. Based on the user's comments, it appears that sound feedback provides the best cue because visual feedback can sometimes be blocked or insufficient with poor contrast and oblique viewing angles.

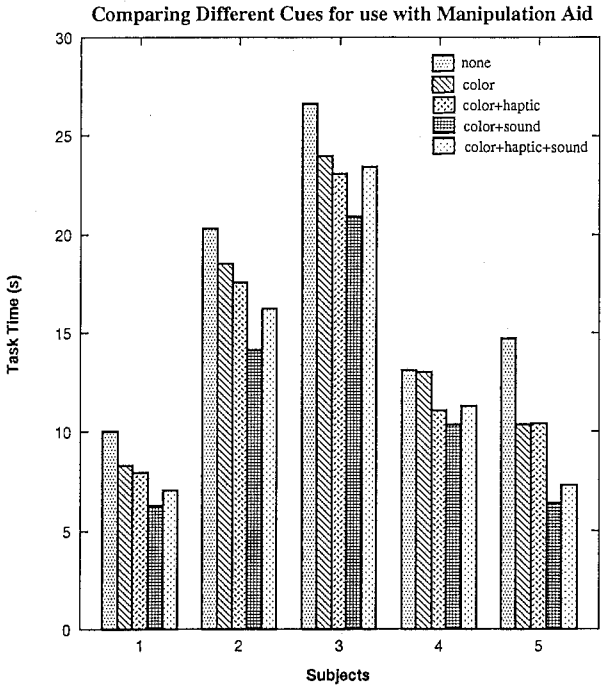


Figure 5.3: Comparing visual, haptic and audio cues for manipulation aid

5.4 Conclusion

In this chapter, we presented an experiment and its results on the investigation of different cues for use with our Manipulation Aid. The results show that a combination of visual and audio cue provide the best efficiency. The haptic feedback provides a feeling of object awareness but the clutch behavior may not be appropriate for use as a constraint cue. The audio cue provides quick situation awareness while visual cue is useful for localization of situation change.

Chapter 6

Extension of Basic Constraints

In this chapter, we present extensions to the basic face constraints described in Chapter 2. We include interactions between other object features such as vertex to face and edge to face. These different interactions can be detected and selected automatically based on assumptions about the user's intentions. There are two other functions that may be useful. The first is called Obstruction Constraint which handles interactions where objects cannot be snapped together because of geometry. The second function handles interactions when there are parallel faces by allowing constraints to be transferred between faces.

6.1 Vertex to Face Constraint

The objective of this constraint is to constrain one vertex of the moving object onto a face of a target object such that there are no interpenetration of the objects. See Figure 6.1.

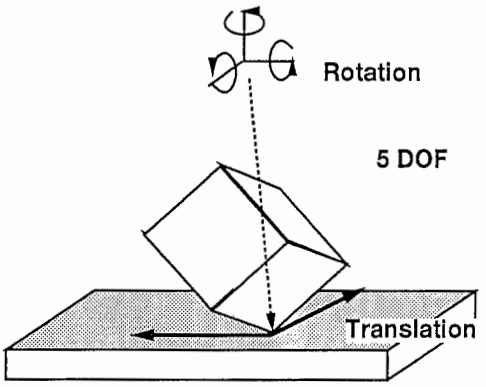


Figure 6.1: Vertex to face constraint

6.1.1 Selecting the Vertex

The best face pair is determined as in Chapter 2.3.3. From the chosen face pair, find the vertex of the moving object closest to the colliding face on the target object. The closest vertex is the one to constrain on the surface.

6.1.2 Constraining Object Motion

The object is translated onto the surface by projecting the closest vertex to the constraining face. Once the object is on the surface, constrain begins.

A vertex on a face can have 5 DOF: 2 translations on the face and 3 rotations about the constrained vertex as shown in Figure 6.1. The translation is constrained with vectors A and B in the same way as one face constraint described in Chapter 2.3.4. The rotation is a little more complicated. The rotation must be about the constrained vertex point. We calculate the rotation matrix as follows:

$$M_{rot} = T(vtx)R_z(\Delta a)R_y(\Delta e)R_x(\Delta r)T(-vtx) \tag{6.1}$$

where,

M_{rot} is the rotation matrix to apply to the current object position,
vtx is the chosen vertex point, and
Δa, Δe and Δr are the changes in azimuth, elevation and roll of user's hand orientation.

6.1.3 Release from Constraint

There are two conditions for unsnap: distance from the surface and vertex inside face. Unlike the face constraint method, the distance check does not use a dynamic threshold since overlap area is not applicable in a vertex to face constraint. Hence, we use a static distance threshold which can be adjusted by the user. The second condition checks whether the vertex is still inside the constraining face. This is done using a simple vertex-in-polygon test.

6.2 Edge to Face Constraint

The objective of this constraint is to constrain one edge of the moving object on a face such that there are no interpenetration as shown in Figure 6.2.

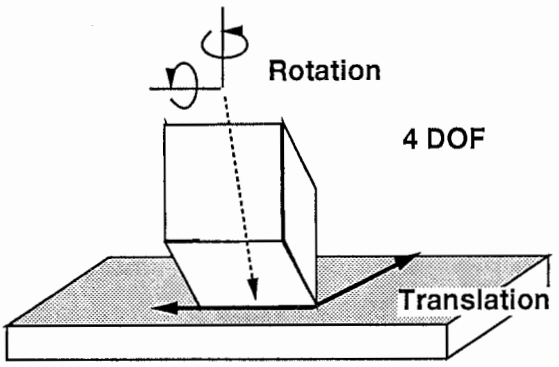


Figure 6.2: Edge to face constraint

6.2.1 Selecting the Edge

The best face pair is determined as in Chapter 2.3.3. From the chosen face pairs, find two vertices of the moving object closest to the colliding face on the target object. These two vertices form the edge to be constrained on the surface.

6.2.2 Constraining Object Motion

To snap the edge onto the surface, we need to translate and rotate the object. The translation is simply projecting the closest vertex to the surface as we did for vertex to face. Rotation is slightly more complicated because we must rotate the object to put the second vertex exactly on the surface. The mathematics involves rotating an arbitrary vector and the details are provided in Appendix A.

An edge on a face can have 4 DOF: 2 translations on the face and 2 rotations, one about the edge itself, the other about the normal of the constraining face. See Figure 6.2. The translation is constrained with vectors A and B in the same way as one face constraint described in Chapter 2.3.4.

Since there are two possible rotations, we calculate each rotation matrix separately and then combine. We use the same method as described for rotation in one face constraint. The first rotation is about the edge itself; the center of rotation is one of the edge endpoints. The second rotation is about the normal of the constraining face; the center of rotation is the midpoint of the edge. The component angle is the dot product of the rotation vector and the input delta angles, as for one face constraint (Equation 2.4). The combined rotation matrix is then:

$$M_R = M_{edge} M_{orth} \tag{6.2}$$

where,

M_{edge} is the matrix for rotation about the edge, and
M_{orth} is the matrix for rotation about the normal of the face

6.2.3 Release from Constraint

There are two conditions for unsnap: distance from surface and edge inside face. The distance from surface check is the same as for vertex to face constraint using a static distance threshold. The edge inside face condition requires the following check.

```
if either endpoint is inside face,
  constraint is ok
else (both is outside face, but edge can still overlap face)
  check for intersection of edge with all the edges of constraining face
  if intersection is found
    constraint is ok

otherwise, edge is totally outside face, so unsnap from constraint
```

6.3 Auto Selection of Feature Constraint

The three types of feature interactions, face-face, vertex-face and edge-face are dynamically selected based on the feature angles of the moving object relative to the target object. Our assumption is that the user will make the features as parallel as possible if the user's intention is to join the features together. We use simple threshold angles to select between the three interactions. The pseudocode is as follows:

```
(after finding the best face pair)
find face-face angle
if face-face angle <= face-face_threshold
  interaction is face-face
else
  find best edge
  find edge-face angle
  if edge-face angle <= edge-edge_threshold
    interaction is edge-face
  else
    find best vertex
    interaction is vertex-face
```

We can also proceed from vertex-face to edge-face, and to face-face constraints by using these threshold checks.

The current implementation can only handle vertex-face and edge-face at the first constrain. Secondary constraints are assumed to be face-face. These limitations may be removed in future expansion of this manipulation method when all feature combinations can be handled.

6.4 Obstruction Constraint

The objective of this constraint is to handle collisions with objects which cannot be handled using the methods described previously. An example of such a situation is shown in Figure 6.3.

Theoretically, this constraint is not necessary when all combinations of feature interactions are implemented. For special circumstances however, this constraint may still be useful.

6.4.1 Detection of Obstruction

The detection for whether a pair is valid for snap or an obstruction is determined in two ways depending on how many faces are currently constrained.

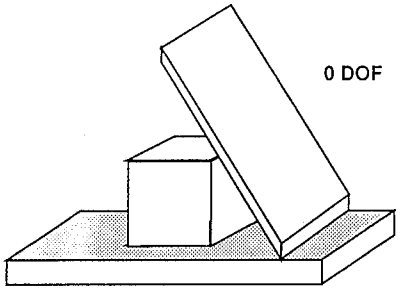


Figure 6.3: Example of an obstruction constraint

If one face is constrained, we calculate the rotation vector required to snap to the new face. If the direction of the rotation vector is the same as the normal of the first constrained face, then the pair is valid for constrain, otherwise it is an obstruction.

If two faces are constrained, we have 0 DOF rotation. Hence the third face pair must be parallel for valid face-face constrain. Therefore, we can check the normals of the face pair to determine if it is valid to snap or an obstruction.

6.4.2 Constraining Object Motion

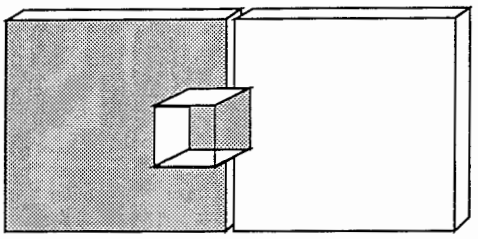
After obstruction is detected, we save the point of collision and the normal of the constraining face. The object is constrained at the collision position as long as the user's hand position is on the opposite side of the constraining face. This function should be replaced with a more sophisticated scheme when all feature interactions can be handled.

6.5 Transfer of Constraint

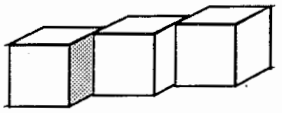
We discussed the ways in which objects may unsnap from constraint in Chapter 2.3.5. The method assume there are no objects immediately near by to continue constraining the object.

Consider the situation shown in Figure 6.4a. A cube is constrained to one of two parallel-aligned walls. The distance from surface and overlap area are checked as the cube moves to the right. Once the cube moves off the first wall, it will unsnap and the cube will jump to the free hand position. However, the user's intention is probably to continue moving the cube across the right wall. To handle situations like these involving parallel faces, we developed a method called Constraint Transfer. Other examples are shown in Figure 6.4b and c.

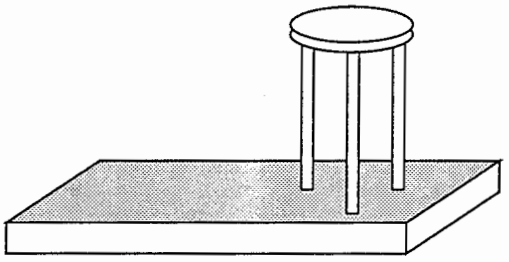
The goal of Constraint Transfer is to simulate the natural motion across two parallel faces by transferring constraint of the current faces without disruption. This is accomplished by using the overlap areas on each of the parallel faces to decide the transition. The face with the greater overlap area has more attraction power and constraint is transferred to it.



(a) parallel aligned faces



(b) parallel opposite faces



(c) multiple parallel pairs

Figure 6.4: Examples of constraint transfer

Chapter 7

Conclusion and Future Work

In this report we described a method to assist users in the manipulation of virtual objects by controlling the graphical display of objects to prevent their inter-penetration. Our manipulation aid method makes use of the intuitive phenomenon of magnetic attraction to constrain the degrees of freedom of an object during collision with other objects. The manipulation aid does not require predefined properties to be attached to objects because it can dynamically constrain objects by using collision output face pairs from a real-time collision detection algorithm.

Several experiments were conducted to test the efficiency of the manipulation aid against other methods such as without interface aid, with collision cue, with real objects, and with different input devices. In all these cases the results show that the manipulation aid is able to assist the user to manipulate virtual objects more efficiently and with more ease.

The interaction of arbitrary objects in virtual space is a complex problem which depends on the application. The manipulation aid presented in this report attempts to handle only some of the more common ways objects may interact, such as between faces, edges and vertices. Not all of the feature interactions are implemented yet, hence future work may include other constraints such as edge to edge, face to vertex, face to edge, etc. Another extension is to handle multilevels of these different types of constraints.

Interactions between curved objects present an interesting problem since natural manipulation with curves is often done in a smooth manner. In polyhedral representation, curved surfaces are modeled by small pieces of face segments. The simultaneous interaction of these many faces to provide smooth and natural assistance is a topic for future work.

Another area of interest is combining the manipulation aid with a force feedback device that can properly simulate reactions between objects. We have already found that the manipulation aid may be independent of device. However employing a suitable haptic device may further increase the naturalness and realism of object manipulation.

Acknowledgements

I would like to thank Dr. K. Habara, Executive Vice President of ATR International and Chairman of the Board of ATR Communication Systems Research Laboratories, and Dr. N. Terashima, President of ATR Communication Systems Research Laboratories, for inviting me to ATR for 8 months.

I would like to express my sincere gratitude to Mr. F. Kishino, Head of the Artificial Intelligence Department, who welcomed me into his department.

I would like to thank my advisor Mr. Y. Kitamura, Researcher, for his generous guidance and support of my work, and for his helpful advice in writing technical papers and reports.

Also, I would like to thank Mr. T. Takumi, CSK programmer, for all the help in programming and setting up experiments. Also, much thanks to everyone who participated in the experiments.

Finally, I would like to thank the members of the Planning Division and the Planning Section of ATR Communication Systems Research Laboratories for their support and organization of my stay in Japan.

Bibliography

[Bar89] Baraff, D. Analytical methods for dynamical simulation of non-penetrating rigid bodies. In *Computer Graphics, Proceedings SIGGRAPH*, pp. 223–232. ACM, 1989.

[Bie90] Bier, Eric A. Snap-dragging in three dimensions. In *Computer Graphics, 1990 Symposium on Interactive 3D Graphics*, pp. 193–204. ACM, 1990.

[BV] Bouma, William J. and Vanecek, George Jr. Modeling contacts in a physically based simulation. In *To appear in Computer Aided Design.*

[CTKK93] Chanezon, A., Takemura, H., Kitamura, Y., and Kishino, F. A study of an operator assistant for virtual space. In *Virtual Reality Annual International Symposium*, pp. 492–498. IEEE, 1993.

[IS93] Ishii, M. and Sato, M. A 3D interface device with force feedback: a virtual work space for pick-and-place tasks. In *Virtual Reality Annual International Symposium*, pp. 331–335. IEEE, 1993.

[Iwa90] Iwata, Hiroo. Artificial reality with force-feedback: development of desktop virtual space with compact master manipulator. *Computer Graphics*, Vol. 24, No. 4, pp. 165–170, 1990.

[KKT92] Kotoku, Tetsuo, Komoriya, Kiyoshi, and Tanie, Kazuo. A robot simulator with force generating function — configuration space approach. In *Second International Symposium on Measurement and Control in Robotics*, pp. 805–810, 1992.

[KTT94] Kotoku, Tetsuo, Takamune, Kouich, and Tanie, Kazuo. A virtual environment display with constraint feeling based on position/force control switching. In *IEEE International Workshop on Robot and Human Communication*, pp. 255–260. IEEE, 1994.

[RA76] Rogers, D.F. and Adams, J.A. *Mathematical Elements for computer graphics.* 1976.

[SKTK95] Smith, A., Kitamura, Y., Takemura, H., and Kishino, F. A simple and efficient method for accurate collision detection among deformable polyhedral objects in arbitrary motion. In *Virtual Reality Annual International Symposium, North Carolina, USA.* IEEE, March 1995.

[SP94] Sayers, Craig P. and Paul, Richard P. An operator interface for teleprogramming employing synthetic fixtures. In *Presence: Special Issue on Networked Virtual Environments and Teleoperation*, 1994.

[Ven93] Venolia, Dan. Facile 3D direct manipulation. In *INTERCHI*, pp. 31–36. ACM, 1993.

Appendix A

Matrix Mathematics

The following matrix is used to rotate an object about a vector $\vec{v} = (v_0, v_1, v_2)$ and rotation angle θ. [RA76]

$$A = \begin{pmatrix} v_0^2 + (1 - v_0^2)cos\theta & v_0 v_1(1 - cos\theta) + v_2 sin\theta & v_0 v_2(1 - cos\theta) - v_1 sin\theta & 0 \\ v_0 v_1(1 - cos\theta) - v_2 sin\theta & v_1^2 + (1 - v_1^2)cos\theta & v_1 v_2(1 - cos\theta) + v_0 sin\theta & 0 \\ v_0 v_2(1 - cos\theta) + v_1 sin\theta & v_1 v_2(1 - cos\theta) - v_0 sin\theta & v_2^2 + (1 - v_2^2)cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$