Manipulation Aids for Two-handed and Collaborative 3-D Design in a Virtual Environment

April 14, 2003

ABSTRACT

In this paper, described is a case study of building a prototype of a cooperative two-handed virtual modeling environment VLEGO which employs a set of manipulation supporting methods. Though there is a number of advantages when designing 3-D objects in a virtual environment, manipulation of virtual objects without a force-feedback is prone to be awkward because of computational delay, restricted spatial resolution of input/output devices, and physiological limitation such as depth perception and trembling of limbs. VLEGO employs a set of manipulation supporting methods to attack the awkwardness of virtual manipulation.

VLEGO II is now under development to support a collaborative design for multiple users. In this paper, we also discuss manipulation supporting methods suitable for virtual collaborative design and show an empirical study evaluating the effectiveness and correlation of the manipulation supporting methods used in VLEGO II in both two-handed and collaborative virtual objects assembly. The results indicate that appropriate combination of the methods can improve the efficiency of both two-handed and collaborative virtual objects assembly.

KEYWORDS: virtual reality, 3-D design, two-handed interaction, computer supported cooperative work.

1 INTRODUCTION

Owing to the extraordinary progress of computer performance, increasing number of virtual reality applications have been built. One of attractive applications using virtual reality technology is to

design 3-D objects in a virtual environment. Designing 3-D objects in a virtual environment has a number of advantages[?, ?, ?]. Firstly, 3-D objects can be observed through a stereoscopic display from an arbitrary point of view in real-time. This feature improves understanding of shape and spatial relationships. Secondly, direct manipulation can be performed using spatial input devices. This feature improves accessibility to 3-D objects in intuitive and quick way. In addition to these realistic features, virtual reality can provide additional unreal features which make the best use of computers. For example, color, shape, and scale can be changed in real-time, collaborative design can be performed through network.

However, manipulating virtual objects is prone to be awkward because of lack of force-feedback, computational delay, restricted spatial resolution of input/output devices, and physiological limitation such as depth perception and trembling of limbs. So, in order to make the interface feasible without bulky force-feedback devices, certain practical methods for virtual object manipulation are required. One reasonable way is to restrict not the movement of user's hands physically but the location of virtual objects visually. That is, to put simple and comprehensible constraints on virtual objects movement based on their geometric or functional features[?, ?, ?].

We have developed a virtual modeling environment, VLEGO[?]. VLEGO employs a number of manipulation supporting methods, "discrete placement constraints" and "collision avoidance" to attack the awkwardness in manipulating virtual objects. With discrete placement constraints, possible location and orientation of primitives are discretely limited. With collision avoidance, the system automatically avoids collisions among primitives and adjusts their positions. These two manipulation supporting methods make arrangement of virtual objects facile and precise.

In addition to these manipulation supporting methods, VLEGO supports two-handed interaction to improve the interface. Two-handed interaction is known to enhance the human-computer interaction in a virtual workspace, hence it has been studied by many researchers[?, ?, ?]. Hinckley et al. claims that two-handed input not only improves the efficiency of human-computer interaction, but can also help to make spatial input comprehensible[?]. In order to support two-handed manipulation in combination with the two manipulation supporting methods, VLEGO dynamically controls degree-of-freedom of two objects each manipulated by left and right hands. Owing to these manipulation aids, Users of VLEGO can easily design precise 3-D objects as with real toy blocks using two hands by assembling several primitives of box- and wedge-shapes.

We are now developing VLEGO II, an extended version of VLEGO, which supports collaborative 3-D design. Collaboration within a shared virtual environment has following advantages over collaboration in the real world[?, ?, ?]. Firstly, operators may be away from each other. Secondly, physical properties may be omitted to calculate if preferred. Finally, computer supported flexible setup can be used, e.g., viewpoints can be changed arbitrarily, the floor for objects manipulation can be controled.

Since VLEGO has originally been built for single user, various functions supported in VLEGO II must be extended or modified for collaborative design. In order to investigate how to extend the manipulation supporting methods suitable for collaborative design, we have conducted an empirical study evaluating the effectiveness and correlation of the manipulation supporting methods in both two-handed and collaborative virtual assembly, which is representative operation of the VLEGO II design environment. The results indicate that appropriate combination of the methods can improve the efficiency of both two-handed and collaborative virtual objects assembly.

This paper is organized as follows. Firstly, the characteristics of VLEGO are summarized. Then, design and implementation of VLEGO is explained. Next, we discuss what should be considered to support collaboration and describes the experiment of virtual assembly. At last, we conclude this paper.

2 SYSTEM CHARACTERISTICS

This section summarizes the characteristics of VLEGO. Our virtual modeling environment VLEGO employs the metaphor of widespread Lego-like block toys. This metaphor leads to a number of features which provide us an easy way to create various 3-D objects. The characteristics of VLEGO are summarized as follows:

1. VLEGO supports a number of two-handed manipulations such as assembly, decomposition, coloring and scaling objects, so that 3-D modeling can be performed in intuitive and efficient ways.

2. Possible location and orientation of primitives are discretely limited. Owing to the feature, users can arrange objects accurately with ease.

3. The system detects collisions among picked and all other objects and adjusts the locations of picked objects so as to avoid interference of any two objects. This feature also assists the user

to manipulate virtual objects accurately and intuitively. For instance, virtual objects can be moved over surfaces of other objects while continuing to touch them.

The two-handed manipulations and collision avoidance make manipulation of virtual objects intuitive, while the discrete constraints and the collision avoidance make precise manipulation facile. In consequence, VLEGO offers natural and quick ways to create 3-D objects in the virtual workspace.

3 IMPLEMENTATION

This section describes the implementation of VLEGO. Hardware components, properties of the elements that consist the virtual workspace, three features touched in the previous section and some typical two-handed operations are explained in the following.

3.1 System's Hardware

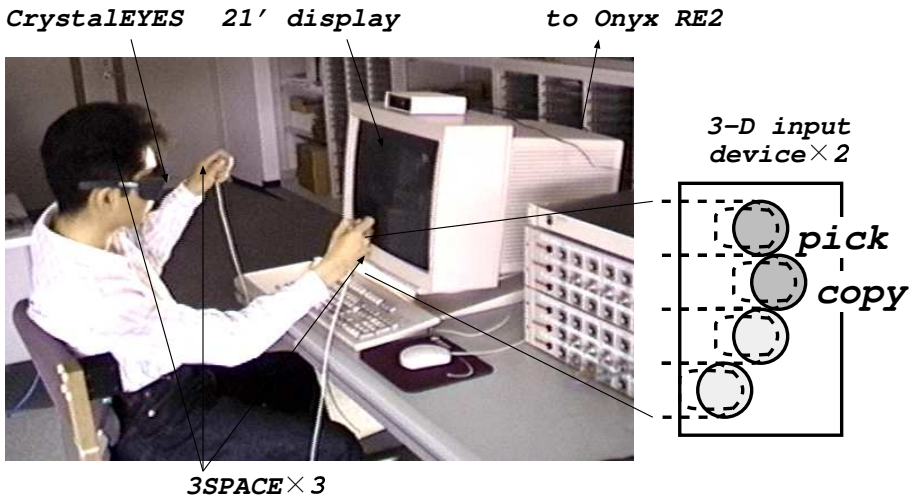

Figure 1: Hardware components.

Figure ?? illustrates hardware components of VLEGO. VLEGO is implemented by C++ and OpenGL on a SGI graphics workstation, Onyx RE2. VLEGO supports both immersive and non-immersive style setup. Non-immersive style setup allows a user to view a stereoscopic virtual workspace displayed on a 21'CRT or a larger projector through a liquid crystal shuttered glasses, CrystalEYES (StereoGraphics). When the setup is an immersive style, virtual workspace can be observed through a head mounted display, i-glasses! (Virtual i.o). In both cases, head-tracking facility is provided using a 3-D magnetic tracker 3SPACE (Polhemus) or mechanic tracker ADL-1 (Shooting Star). A pair of 3-D input devices for two-handed manipulations are made (lower right of Figure??) for two-handed manipulation. Each of them is composed of four feather touch switches and a 3SPACE receiver.

3.2 System's Software

The workspace of the system is composed of a number of elements as shown in Figure **??**: two 3-D cursors, three axes of coordinates, primitives, a primitive box and a palette ball. This subsection is devoted to explain each element.

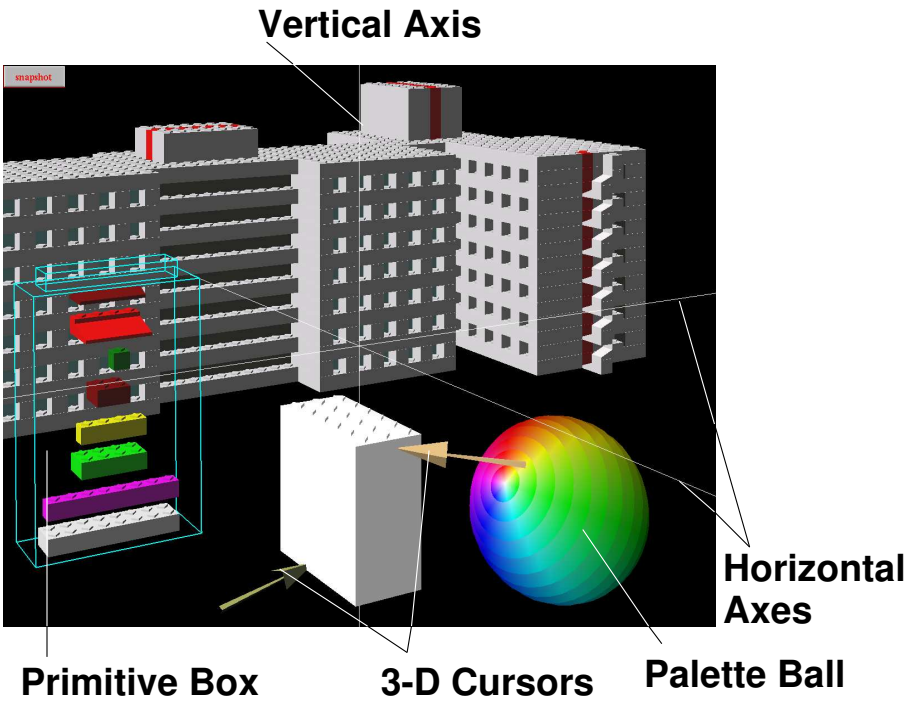

Figure 2: Software components.

3-D cursors: There are two arrow-shaped 3-D cursors that follow to right and left input devices respectively. All operations in VLEGO is performed through the cursors.

Axes: Three pickable axes that are perpendicular to one another are displayed. While one of the horizontal axes is picked, whole world including the axis can be rotated around the vertical axis. While the vertical axis is picked, whole world can be translated in parallel.

Primitives: There are a few kinds of box- and wedge-shaped primitives with or without texture. Height, width and depth of primitives are n (integer) cm long. Primitives are attachable to each other at a discrete position. Color and scale of a primitive can be changed in the workspace. In this paper, *block* stands for a single primitive or a composite of assembled primitives.

Primitive box: Primitive box is a movable wire-frame box which contains a number of typical primitives. Primitives can be generated unlimitedly from the primitive box simply by picking

the contained primitives. On the other hand, users has only to release a picked block in the primitive box to delete the block.

Palette ball: There is a pickable colorful palette ball in the workspace, of which hue, saturation and intensity changes gradually. The palette ball can be picked and used to color primitives using two hands.

3.3 One-handed Manipulations on Primitives

Table 1: Functions on primitives

One-handed	selection, picking, copy, deletion, assembly
Two-handed	assembly, decomposition, coloring, scaling

In VLEGO, as with real toy blocks, 3-D objects can be easily designed using one or two hands by properly assembling primitives that have a few kinds of box- and wedge-shapes. Users can perform a number of functions on primitives as shown in Table **??**. In this subsection, one-handed manipulations are explained.

selection: When a tip of a 3-D cursor is buried in a block, it is selected. To make the selection clear, the stabbed primitive is highlighted and the bounding box of the whole block appears.

picking: The selected block can be picked by pressing the corresponding button. The picked block does not show the visual feedback of the selection any more. The picked block moves following to the 3-D cursor until the block is released by releasing the button.

copy: The picked block can be copied by pressing the corresponding button.

deletion: The picked block can be deleted by releasing the block in the primitive box as mentioned above.

assembly: When the upper or lower face of the picked block contacts with other non-picked blocks, these blocks can be joined by releasing the picked block. Both one-handed and two-handed assembly is detailed in subsection **??**.

3.4 Discrete Constraints and Operating Modes

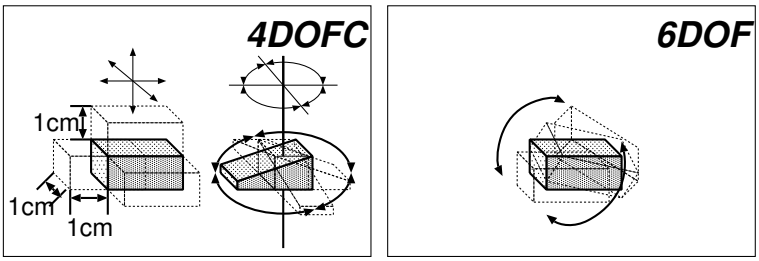

Figure 3: Two types of constraints.

This subsection describes how VLEGO restricts the positions of blocks. First of all, we introduce two alternative types of constraints put on blocks. In principle, each picked block is bound to **4DOFC** (4 Degrees Of Freedom with Constraints). Being bound to 4DOFC, the block can be located on discrete positions at intervals of 1cm and its orientation is restricted at 0, 90, 180 and 270 degrees of horizontal rotation. As a result, all non-picked block is always aligned one another with 4DOFC. Owing to these grid constraints, a user is able to arrange virtual objects accurately with ease. Left side of Figure **??** illustrates 4DOFC. On the other hand, there are a number of cases when a block is bound to **6DOF** (6 Degrees Of Freedom). Being bound to 6DOF, the position and orientation of the blocks can be operated freely as shown in right side of Figure **??**.

In order to support natural two-handed operations, VLEGO has two alternative operating modes, *separative mode* and *cooperative mode*, that are switched dynamically according to the distance between two blocks picked by left and right hands. Each operating mode makes good use of two types of constraints explained above.

Separative mode: Separative mode is selected when only one hand picks a block or when two blocks picked by two hands are distant from each other (the latter case is shown in Figure **??**, S-1). In this mode, the user can not attach two picked blocks each other. Therefore, it is natural to consider that the user would be manipulating each picked block independently around non-picked blocks. In this mode, hence, in order to assist the user to align picked blocks with non-picked blocks, each picked block is manipulated with 4DOFC based on the global coordinate system. When each hand picks a block and these two blocks are close enough to each other, operating mode is smoothly changed from separative mode to cooperative mode mentioned below as shown in Figure **??**, S-1.→C-1.).

Cooperative mode: Cooperative mode is selected when each hand picks a block and the distance

between them is short enough. Figure ??, C-1. and C-2. show this case. In this mode, then, it is natural to consider that a user would be aligning two picked blocks each other. Therefore, to provide as natural and easy way for aligning blocks, the system puts 6DOF and 4DOFC constraints onto firstly picked block (called *base*) and secondly picked one (called *work*), respectively. What is important is that the 4DOFC of the *work* is dynamically determined based on the orientation and location of the *base*. Owing to these constraints, two-handed assembly or decomposition of blocks becomes intuitive and effective. When one of the blocks is released or two picked blocks depart from each other, these blocks are forced to align with all non-picked blocks with 4DOFC, and operating mode changes to the separative one as shown in Figure ??, from C-2.→S-1.. In this mode transition, orientations and positions of blocks are smoothly interpolated by animation for a duration of about 500ms to reduce the cognitive load arisen from visual discontinuity.

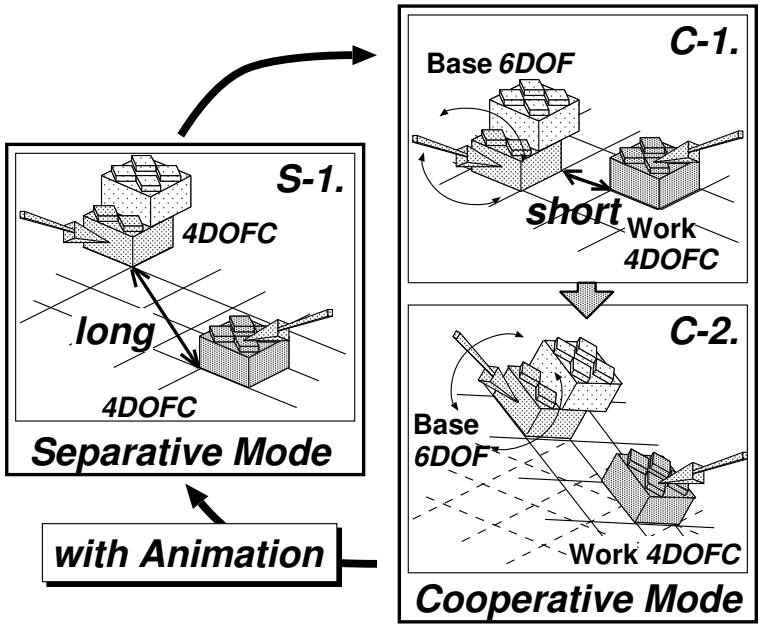

Figure 4: Transition of operating modes.

3.5 Collision Avoidance

In order to simplify the algorithm of the collision detection, the subjects of the collision detection vary according to the two operating modes. In the separative mode, among picked and all other blocks are checked while only between two picked blocks is checked in the cooperative mode, so that all subjects to be checked should be already aligned one another with 4DOFC in each case. This restriction simplify the algorithm for collision detection between two blocks.

The collision avoidance is performed as follows.

1. Let the picked block move toward the corresponding 3-D cursor in X direction while no collision occurs.

2. Let the picked block in Y and Z direction in the same way as above.

3. Draw a line between the picked block and the cursor to indicate the discrepancy of their locations.

3.6 Two-handed Manipulations on Primitives

In this subsection, a number of two-handed manipulations are explained.

Assembly of blocks All primitives in VLEGO can be simply attached to each other in the same way as toy blocks. If a picked block contacts to other blocks vertically after the collision avoidance, these blocks are joined when the picked block is released.

In the separative mode, the system detects collisions between picked blocks and all other blocks, then adjust the positions of picked blocks before the scene is rendered in order to avoid the interference among blocks. After the collision avoidance process, if the picked blocks can be attached to any other block, all bounding boxes of the picked and all attachable blocks appear to indicate the situation. All these blocks stick one another if the picked objects are released. In the cooperative mode, the system avoids the interference between two picked blocks, however it does not check the collision and attachability between picked and all other blocks. In this case, two bounding boxes of both blocks appear if they can attach mutually (Figure **??**, C), and the two blocks stick each other when one of them is released (Figure **??**, D).

Decomposition of blocks There are two methods to decompose assembled blocks. First, users can remove a primitive with a hand from the block picked by the other hand. The removal changes the operating mode to the cooperative one. Second, assembled blocks can be divided into two parts using a **cutting plane**. Detailed process is as follows: 1. pick a block (base) with a hand (Figure**??**, A→B),
2. press the first switch of the other hand (work) at some position where no object is selected,
3. then the shape of the 3-D cursor of the work changes translucent square (namely, cutting plane) (Figure **??**, B→C), now, both of the picked block and the cutting plane can be manipulated exceptionally with 6DOF,

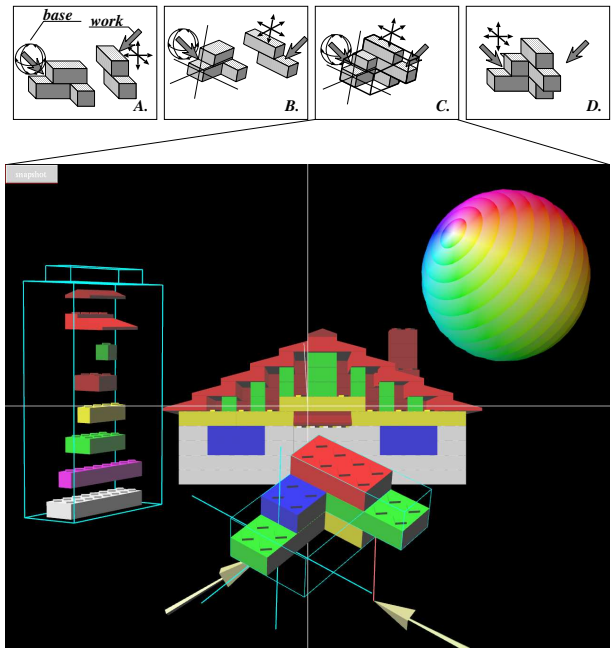

Figure 5: An example of assembly.

Figure 6: An example of decomposition.

4. release the cutting plane at proper angle and picked block is divided into two blocks (Figure **??**, C→D).

Coloring primitives For coloring primitives, users can choose any color using one hand from the palette ball picked by another hand. Once the palette ball is picked, it can be manipulated with 6DOF and a wire-frame cube indicating current color appears as shown in Figure **??**, b. While the ball is picked, users can select any color by pointing a primitive or the ball using another hand (called work). Figure **??**, c shows an example of selecting color from a primitive, Figure **??**, d shows an example from the ball. Once users push the picking button, the current color is decided to color primitives and the small cube colored the decided color appears the tip of the work. Then users can color primitives until the switch or the ball is released as shown in Figure **??**, e → h.

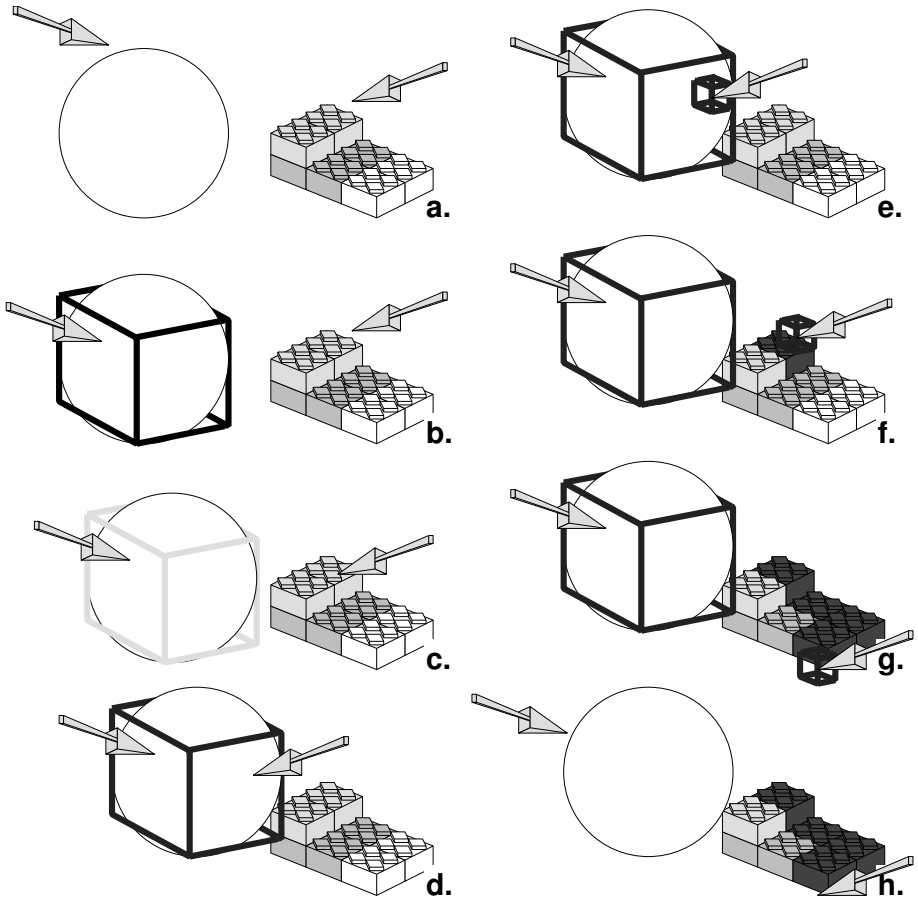

Figure 7: An example of coloring primitives.

Other two-handed operations A picked primitive can be discretely scaled using two hands. For example, a white primitive is being scaled in Figure **??**. In addition to cooperative two-handed operations, all one-handed operations can be performed concurrently by left and right hands.

4 COLLABORATIVE DESIGN WITHIN VLEGO II

In this section, we discuss manipulation supporting methods suitable for collaborative design within VLEGO II. The most significant difference between two-handed and collaborative objects manipulation is mutual understanding between the hands. A user manipulates both objects in two-handed manipulation, hence he/she knows of course what each hand wants to do. So, two-handed manipulation such as assembly and decomposing objects can be performed easily. However, it is much more difficult to recognize what the partner wants to do in collaboration so that it may be harder to perform collaborative design.

Needless to say, this difficulty also arises in the real world. In the real world, however, we utilize various information to communicate with partners, such as expressions, gestures, bearing, glances,

voice and sound. Transferring these properties should enhance awareness and make virtual collaboration smooth.

In addition to enrich the communication channels, what we should consider here is how to support manipulation. Visual placement constrains lead significant discrepancy of locations of a hand and a corresponding manipulated object, causing abrupt movement of objects. Such unexpected sudden movement easily impede virtual tasks. This demerit influences little when behavior of the base coordinate system of the constraints is well understood by a user, e.g. one-handed and two-handed manipulation for single user. In these cases, he/she can manage to understand the abrupt movement and manipulate objects as he/she likes. However, in collaborative work, it is hard to recognize the abrupt movement so that the demerit of the visual constraints may overwhelms their advantages.

To provide too poor visual placement constraints leads to the difficulty of accurate arrangement of virtual objects. To provide too strong visual placement constraints leads then to confusion arisen from the abrupt movement of objects. Though appropriate manipulation aids depends on the application, there should be a compromise. In order to find the compromise for collaboration within VLEGO II, we focus virtual collaborative assembly. We show an experiment for evaluating the effectiveness and correlation of the manipulation supporting methods in two-handed and collaborative assembly in the next section.

5 EXPERIMENT

This section describes an experiment for evaluating the effectiveness of the manipulation supporting methods in two-handed and collaborative assembly. VLEGO has been expanded to VLEGO II to share the workspace for two users through network. Each user can manipulate single 3-D cursor and shares the same workspace. Users can change their points of view with no influence to each other. The hardware setup for the VLEGO II for two users is illustrated in Figure **??**.

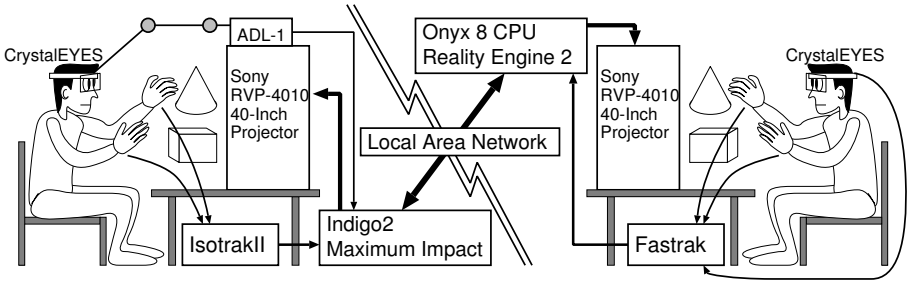

Figure 8: Network configuration.

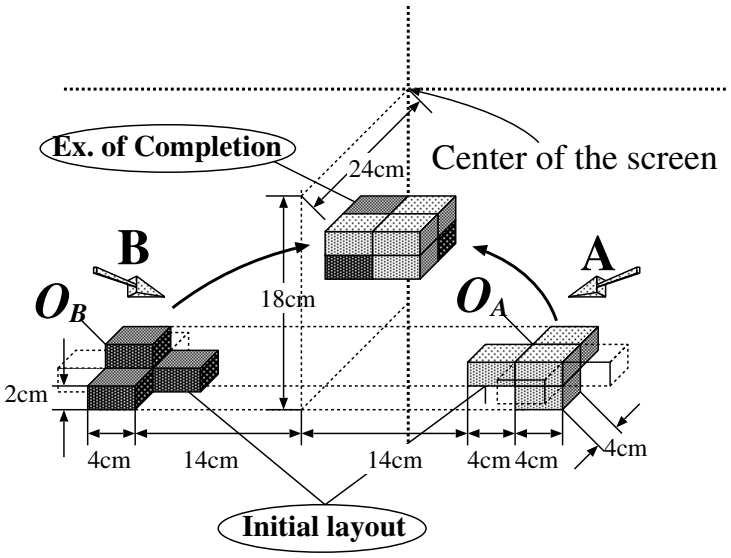

Figure 9: Configuration of blocks in the experiment.

5.1 Experimental Setup

Figure ?? shows the configuration of the experiment. There are two virtual blocks O_A and O_B, and two 3-D cursors A and B in the workspace. These blocks can be fitted together into a box. At an initial state of each trial, O_A and O_B are at fixed points as shown in Figure ??. Each block having four initial postures of horizontal rotation, there are 16 possible patterns of an initial layout, which varies at every trial. One or two subjects are asked to assemble these blocks into a box. This is a compound task composed of the following subtasks.

1. Move each 3-D cursor by moving the corresponding 3-D input device toward its responsible block.

2. Pick each block by pushing down the corresponding button on the 3-D input device.

3. Bring two picked blocks close to each other.

4. Assemble two picked blocks into a box. There can be cases when two blocks are joined but not form a box. Once assembled, however, the block cannot be decomposed.

5. Release the assembled block by releasing one of the picking buttons.

Subject(s) perform this task with the four combination of manipulation methods shown in Table ??.

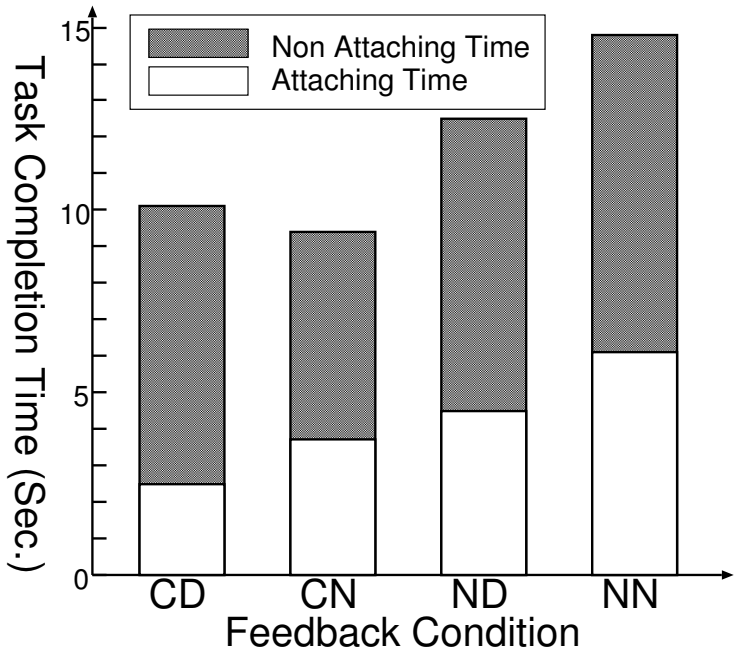

Figure 10: Task Completion Time (Two-Handed).

Table 2: Four Manipulation Conditions in the Experiment.

	collision avoidance	discrete placement constraints
CD	√	√
CN	√	
ND		√
NN		

The following three types of data are recorded.

Task Completion Time Time interval from the first pick of a block to the release of the assembled block.

Attaching Time Total time while two blocks contacted each other.

Error Rate The ratio of the number of the false trials, in which the assembled block does not form a box, to the total number of trials.

5.2 Experimental Results

Eight students of our laboratory performed two-handed assembling task, and four pairs of students performed collaborative assembling task. They are all novice users of VLEGO II. Each subject

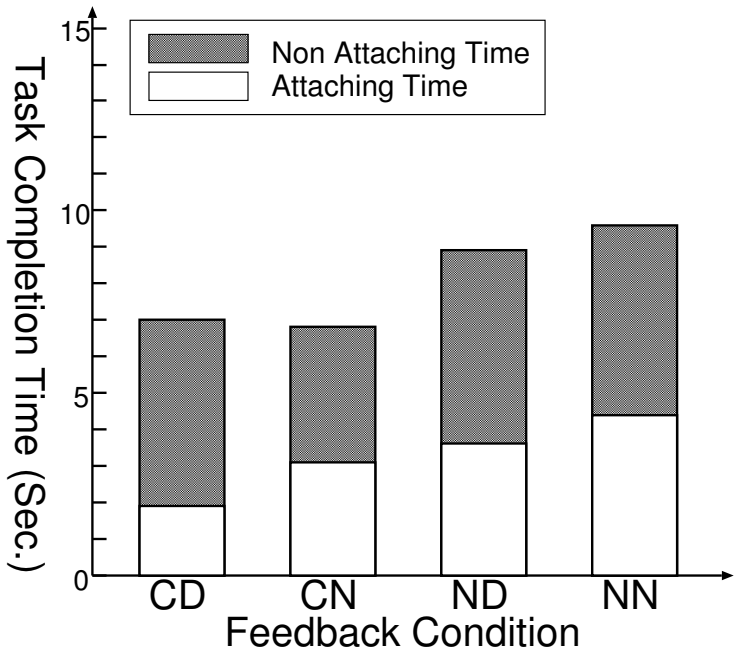

Figure 11: Task Completion Time (Collaborative).

performed 128 trials, i.e., 32 for each manipulation condition. So as to examine and eliminate the effect of practice, in two-handed assembly, four of eight subjects (Group A) performed two-handed assembly with NN, ND, CN and CD methods in order, while the rest (Group B) performed with CD, CN, ND and NN methods in order. Similarly, in collaborative assembly, two of four pairs (Group A') performed with NN, ND, CN and CD methods in order, while the rest (Group B') performed with CD, CN, ND and NN methods in order.

Figures ?? and ?? show the averages of task completion time, attaching time and non attaching time for four conditions. Here, non attaching time is subtraction between task completion time and attaching time. We can see the following features from these results.

Table 3: Ave. of Performance Time (Two-Handed).

	Non Attaching Time				Attaching Time			
	CD	CN	ND	NN	CD	CN	ND	NN
A&B	7.6	5.7	8.0	8.7	2.5	3.7	4.5	6.1
A	4.4	4.6	6.8	8.2	1.9	3.6	4.3	6.2
B	10.7	6.8	9.2	9.2	3.1	3.8	4.7	5.9

Table 4: Error Rates (Two-Handed).

	CD	CN	ND	NN
A&B	0.05	0.08	0.12	0.16
A	0.04	0.07	0.11	0.13
B	0.06	0.09	0.13	0.19

- The order of four methods in average task completion time is the same in both two-handed and collaborative assembly: from the shortest; CN, CD, ND and NN. This shows that the collision avoidance improves the efficiency of virtual assembly.

- The order in four attaching time is also the same in both two-handed and collaborative assembly: from the shortest; CD, CN, ND and NN. This shows that both the discrete placement constraints and collision avoidance improve the efficiency of attaching manipulation. Collision avoidance improves the efficiency because it is easier to keep the relative position of two blocks. Discrete placement constraints improves the efficiency because it is easier to check the relative position and orientation of two blocks.

- The non attaching time with CN is prominently short in both two-handed and collaborative assembly. Furthermore, collision avoidance shortened non attaching time by 19% in average. This is probably because that trusting to collision avoidance, subject(s) could brought two blocks close quickly with rough positioning.

- On the other hand, discrete placement constraints prolonged non attaching time by 12% in average. This is probably because the orientation of *the work* is forcibly changed by the rotation of *the base* so that it is awkward for the subjects to align relative orientation of two blocks.

Tables **??** and **??** show the average attaching time and non attaching time for four manipulation conditions for each group in two-handed and collaborative assembly respectively. Tables **??** and **??** show the average error rates for the conditions for each group. We can see the following additional features from these Tables.

- Both collision avoidance and discrete placement constraints shorten attaching time regardless of the experiment sequence.

Table 5: Ave. of Performance Time (Collaborative).

	Non Attaching Time				Attaching Time			
	CD	CN	ND	NN	CD	CN	ND	NN
A'&B'	5.1	3.7	5.3	5.2	1.9	3.1	3.6	4.4
A'	6.2	3.2	5.8	6.2	2.2	3.3	4.7	5.5
B'	4.0	4.2	4.8	4.3	1.5	2.9	2.6	3.2

Table 6: Error Rates (Collaborative).

	CD	CN	ND	NN
A'&B'	0.05	0.07	0.13	0.12
A'	0.02	0	0.08	0.03
B'	0.09	0.14	0.17	0.20

- Both collision avoidance and discrete placement constraints decrease error rates. This is more distinct with collision avoidance than with discrete placement constraints.

- Comparing CD to CN and ND to NN in further detail, we found that discrete placement constraints prolonged non attaching time in 4 out of 12 cases in groups A and A', and in 10 out of 12 cases in groups B and B'. This is probably because the subjects in groups B and B' performed the task with discrete placement constraints before without the constraints, so that it was harder for them to understand the correlation between the rotation of their hands and the blocks with the constraints than in groups A and A'. In other words, it can be said that the better a user understands how discrete placement constraints work, the better he/she handles the constraints.

There are few differences between two-handed and collaborative assembly. As a whole, the performance time in collaborative mode is shorter than in two-handed mode. This may be because three of four pairs of subjects had experienced two-handed mode in advance, and because they could concentrate their attention on each responsible block so that they could manipulate each block quickly.

6 CONCLUSIONS AND FURTHER WORK

This paper describes a virtual modeling environment VLEGO. Firstly, the implementation and functions of VLEGO for single user was introduced. VLEGO has demonstrated that flexible two-handed interaction based on actual 3-D designing activities enhances the usability of the virtual modeling environment. Being combined with proper constraints on the arrangement of virtual objects and collision avoidance among them, VLEGO offers natural and quick ways to create 3-D objects in the virtual workspace for single user.

Secondly, manipulation supporting methods suitable for virtual collaboration was discussed and an empirical study on virtual assembly was described. The results show the effectiveness of two manipulation supporting methods, "discrete placement constraints" and "collision avoidance," in two-handed and collaborative virtual objects assembly. The results also show that employing both methods is appropriate for attachment of objects, while employing only collision avoidance is appropriate for aligning their orientation in advance of their attachment. Combining these two approaches will facilitate virtual assembly further, which will be investigated in the near future.

As further studies of virtual designing workspace employing simultaneous use of multiple hands, we focus following two themes for development and evaluation. The one is designing methods that have

high ability for creating complicated or freeform 3-D objects using two-handed interaction. The other concerns the enhancement of collaboration, such as providing awareness and supporting simultaneous two-handed manipulation for multiple users more than two.

References

[1] Butterworth, J., Davidson, A., Hench, S. and Olano, T. M.: "3DM: A Three Dimensional Modeler Using a Head-Mounted Display", *Proc. ACM Sympo. on Interactive 3D Graphics*, pp.135–139, 1992.

[2] Smets, G. J. F., Stappers, P. J., Overbeeke, K. and Mast, C. V. D.: "Designing in Virtual Reality: Implementing Perceptual-Action Coupling with Affordances", *Proc. Conf. on Virtual Reality Software and Technology (VRST '94)*, pp.97–110, Aug. 1994.

[3] Kurmann, D. and Engell, M.: "Modelling Virtual Space in Architecture", *Proc. ACM Sympo. on Virtual Reality Software and Technology (VRST '96)*, pp.77–82, July 1996.

[4] Houde, S.: "Interactive Design of an Interface for Easy 3-D Direct Manipulation", *Proc. ACM Conf. on Human Factors in Computing Systems (CHI '92)*, pp.135–142, 1992.

[5] Venolia, D.: "Facile 3D Direct Manipulation", *Proc. ACM Conf. on Human Factors in Computing Systems (INTERCHI '93)*, pp.31–36, 1993.

[6] Kitamura, Y., Yee, A. and Kishino, F.: "Virtual Object Manipulation Using Dynamically Selected Constraints with Real-Time Collision Detection", *Proc. ACM Sympo. on Virtual Reality Software and Technology (VRST '96)*, pp.173–181, July 1996.

[7] Kiyokawa, K., Takemura, H., Katayama, Y., Iwasa, H. and Yokoya, N.: "VLEGO: A Simple Two-handed Modeling Environment Based on Toy Blocks", *Proc. ACM Sympo. on Virtual Reality Software and Technology (VRST '96)*, pp.27–34, July 1996.

[8] Buxton, W. and Myers, B. A.: "A Study in Two-handed Input", *Proc. ACM Conf. on Human Factors in Computing Systems (CHI '86)*, pp.321–326, 1986.

[9] Bier, E. A., Stone, M. C., Fishkin, K., Buxton, W. and Baudel, T.: "A Taxonomy of See-Through Tools", *Proc. ACM Conf. on Human Factors in Computing Systems (CHI '94)*, pp.358–364, 1994.

[10] Kabbash, P., Buxton, W. and Sellen, A.: "Two-Handed Input in a Compound Task", *Proc. ACM Conf. on Human Factors in Computing Systems (CHI '94)*, pp.417–423, 1994.

[11] Hinckley, K., Pausch, R., Goble, J. C. and Kassell, N. F.: "A Survey of Design Issues in Spatial Input", *Proc. ACM Sympo. on User Interface Software and Technology (UIST '94)*, pp.213–222, Nov. 1994.

[12] Takemura, H. and Kishino, F.: "Cooperative Work Environment Using Virtual Workspace", *Proc. Conf. on Computer-Supported Cooperative Work (CSCW '92)*, pp.226–232, 1992.

[13] Ishii, M., Nakata, M. and Sato, M.: "Networked SPIDAR: A Networked Virtual Environment with Visual, Auditory, and Haptic Interactions", *PRESENSE Teleoperators and Virtual Environments*, 3, No.4, pp.351–359, 1994.

[14] Benford, S., Bowers, J., Fahlén, L. E., Greenhalgh, C., Mariani, J. and Rodden, T.: "Networked Virtual Reality and Cooperative Work", *PRESENSE Teleoperators and Virtual Environments*, 4, No.4, pp.364–386, 1995.