VLEGO: A Simple Two-handed Modeling Environment Based on Toy Blocks

Kiyoshi Kiyokawa, Haruo Takemura,
Yoshiaki Katayama, Hidehiko Iwasa and Naokazu Yokoya

Nara Institute of Science and Technology (NAIST)
8916-5 Takayama, Ikoma, Nara 630-01, JAPAN
Tel: +81 7437 2 5291 Fax: +81 7437 2 5299
{kiyosi-k, takemura, katayama, iwasa, yokoya}@is.aist-nara.ac.jp

ABSTRACT

This paper describes a case study of building a prototype of an immersive three dimensional (3-D) modeler which supports simple two-handed operations. Designing 3-D objects in a virtual environment has a number of advantages for 3-D geometry creation over designing with traditional computer aided design (CAD) tools. In order to enhance the human-computer interaction in a virtual workspace, two-handed spatial input has been incorporated into a few 3-D designing applications. However, existing 3-D designing tools do not utilize two handed interaction for enhancing the interface sufficiently. Our prototype immersive modeler, VLEGO, employs some features of toy blocks to give flexible two-handed interaction for 3-D design. Features of VLEGO can be summarized as follows: Firstly, VLEGO supports various two-handed operations and hence it makes design environment intuitive and efficient. Secondly, possible location and orientation of primitives are discretely limited so that the user can arrange objects accurately with ease. Finally, the system automatically avoids collisions among primitives and adjusts their positions. As a result, precise design of 3-D objects can be achieved easily by using a set of two-handed operations in intuitive way. This paper describes the design and implementation of VLEGO as well as an experiment for examining the effectiveness of two-handed interaction.

Keywords
virtual reality, 3-D user interface, two-handed interaction, collision detection.

INTRODUCTION

Designing 3-D objects in a virtual environment has three potential advantages over designing with traditional 3-D CAD tools: Firstly, 3-D objects can be observed through a stereoscopic display from an arbitrary point of view based on the user's head position in real-time. This feature helps the user to understand the spatial relationships among objects in 3-D space. Secondly, spatial input devices enable direct manipulation of objects, and that makes 3-D design intuitive and usable. Finally, they offer highly interactive modeling environments; i.e., modifications in design process are directly reflected by the appearance of corresponding objects. According to the rapidly accelerating progress of virtual reality (VR) technology, a number of researches have been made to produce a virtual workspace for 3-D design [1, 2, 3, 4].

Two-handed interaction are known to enhance the human-computer interaction in a virtual workspace, and it attracts 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 [5]. There are some 3-D designing tools employing two-handed interaction [3, 4]. However, two-handed interaction in such tools seems to be based on 2-D painting or drawing rather than actual 3-D designing activities; e.g., sculpturing, building up bricks and making a plastic models. 3-D designing activities are expected to make better use of two-handed interaction than 2-D designing. Therefore, for the purpose of providing more intuitive ways for 3-D design, we employ two-handed interaction based on actual 3-D designing methods.

In this paper, described is a case study of building a prototype of an immersive 3-D modeler supporting simple two-handed operations. Here, *immersive style* means such a viewing style that dynamically provides a pair of binocular stereoscopic images according to the user's viewing direction, of which field of view is sufficiently wide for designing. Our immersive modeler, VLEGO (VirtuaL Environment for Generating Objects) employs the metaphor of widespread toy blocks, which includes flexible two-handed manipulations. In addition to the two-handed interaction, this metaphor leads to other significant features : discrete placement constraints and collision detection detailed in later section. These features reduce the difficulty of accurate manipulation of 3-D objects, which arises from computational delay, restricted spatial resolutions of input/output devices, and physiological limitation such as depth perception and trembling of limbs. In consequence, VLEGO offers natural and quick ways to create 3-D objects in the virtual workspace.

This paper is organized as follows. Firstly, previous related works are briefly reviewed. Then, three characteristics of VLEGO are summarized: two-handed manipulations, discrete constraints and collision avoidance. The implementation of VLEGO is then explained, especially its characteristics and typical manipulations. Finally, an experiment for examining the effectiveness of the two-handed manipulation is described.

RELATED WORKS

In this section, related works are briefly reviewed. Table 1 shows the classification of a number of designing tools including VLEGO according to their characteristics.

Butterworth et al., at the University of North Carolina, built an immersive style tool named 3DM [1]. The system employs the model manipulation techniques from both CAD and 2-D drawing programs like MacDraw. The 3DM uses a head mounted display so as to give the effect of being in the virtual workspace.

Liang, Halliday et al., at the University of Alberta, developed the JDCAD and the JDCAD+ [2]. These non-immersive style systems use a single six-degrees-of-freedom input device to design both geometry and behavior of 3-D mechanical parts.

Although these tools are well designed, they do not provide any two-handed interaction.The less-dominant hand is more than a poor approximation of the dominant hand. Kabbash et al. showed that the less-dominant hand performed as same ability as the dominant hand for relatively large targets and large distance in pointing and dragging tasks [6]. According to the previous works [6, 7, 8, 9, 10] and our observation through the experiment described later, it was confirmed that two-handed interaction improves the efficiency of human-computer interaction in a number of ways as follows.

- Users can perform various tasks with flexible and efficient ways because he/she is able to set multiple values simultaneously with up to 12 degrees of freedom. That is, two-handed interaction enables users to perform two sub-tasks in parallel, and also encourages users to perform one task using two hands cooperatively. Here, two-handed cooperation means such manipulation that the less-dominant hand holds an object and the dominant hand holds another object or some kind of tools, then both hands perform together an action — attaching, shaving, cutting, coloring, and so on. The passive interface props presented by Hinckley et al. are examples of making good use of cooperative two-handed interaction [11].
- Using both hands helps users' understanding of spatial relationship because users know precisely where their hands are relative to each other. In addition to this feature, cooperative two-handed manipulations make it facile to select a point in a virtual workspace in spite of screen vacillation arisen from unstable data of position trackers. This is because both hands vacillate in almost the same direction so that the vacillation is canceled out and their relative positions are comparatively stable.

As a 2-D drawing application, Toolglass developed by Bier et al. at Xerox PARC is a good example using two-handed interaction [12]. The user moves a transparent sheet using one hand on which various tools are located, and the other hand clicks through the tool onto application objects.

There are several tools for creating 3-D objects using two-handed interaction. Sachs et al. at MIT developed the 3-Draw as a non-immersive tool for designing 3-D shapes [3]. The system's user interface is based on a pair of six-degrees-of-freedom input devices, one held in each hand so as to simplify shape design. Shaw et al. at the University of Alberta built the non-immersive style system, THRED, for two-handed polygonal surface design [4].

There are two matters to be pointed out in 3-Draw and THRED in the context of our goal of providing more

Table 1: Classification of designing tools.

	one-handed interface	two-handed interface
2-D drawing	MacDraw	Toolglass
non-immersive style 3-D modeling	JDCAD, JDCAD+	3-Draw, THRED
immersive style 3-D modeling	3DM	VLEGO

intuitive ways for 3-D design. One is that 3-Draw and THRED do not provide immersive style stereo viewing facility, which enables the user to understand the virtual workspace intuitively and to manipulate virtual objects directly. In order to provide intuitive usability, immersive style is one of key factors to be considered, especially when combined with two-handed interaction, because it makes better use of knowledge of spatial relationship in the virtual workspace.

The other matter is that the less-dominant hand is not so effectively utilized for enhancing the usability in comparison with its ability. That is, creating 3-D shapes is performed mainly by the dominant hand and the less-dominant hand is used for moving the base coordinate frame of the model or selecting the status of the modeler. The less-dominant hand has a variety of functions in actual 3-D objects creation such as building a miniature toy model : it performs the same functions as the dominant hand concurrently and assists the functions of the dominant hand cooperatively as well. Therefore, in order to make creating 3-D objects intuitive, it is useful to employ more flexible two-handed interaction based on actual 3-D designing activities.

SYSTEM CHARACTERISTICS

In order to develop intuitive and quick ways for creating 3-D objects, widespread Lego-like block toys attracted us. Toy blocks provide us an easy way of creating various 3-D objects owing to a number of features. Our immersive modeler, VLEGO employs the features of toy blocks. The characteristics of VLEGO are summarized as follows:

1. VLEGO supports a number of flexible 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, the user can arrange objects accurately with ease in spite of the computational delay and screen vacillation arisen from unstable data of position trackers and trembling limbs.
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, a virtual object 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.

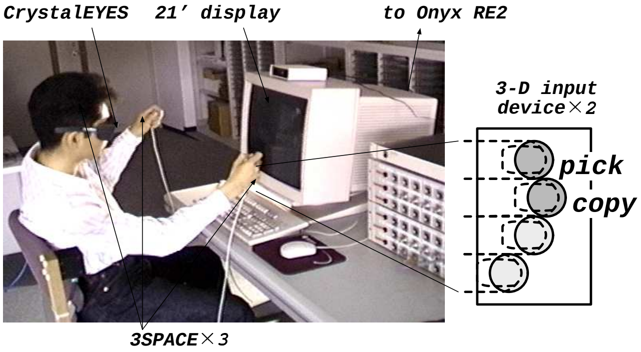

Figure 1: Hardware components.

IMPLEMENTATION

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

Hardware

Figure 1 illustrates hardware components of VLEGO. VLEGO is implemented on a graphics workstation Onyx RE2 (SGI). A user views a stereoscopic virtual workspace through a liquid crystal shuttered glasses, CrystalEYES (StereoGraphics) with head-tracking facility using a 3-D magnetic tracker 3SPACE (Polhemus). A pair of 3-D input devices for two-handed manipulations are build (lower right of Figure 1). Each of them is composed of four feather touch switches and a 3SPACE tracker.

Software

The workspace of the system is composed of a number of elements displayed on the screen as shown in Figure 2: two 3-D cursors, three coordinates axes, object primitives, a primitive box and a palette ball. This subsection is devoted to explain each element.

3-D cursors

There are two arrow-shaped 3-D cursors corresponding to right and left input devices respectively, that move freely following to the devices. 3-D cursors are used to perform all operations in VLEGO.

Coordinate axes

The origin of the coordinate system of VLEGO is the center of the screen. Two horizontal coordinate axes and a vertical coordinate axis, that are perpendicular to one another, are displayed on the screen. Two horizontal coordinate axes can be picked and rotated around the vertical coordinate axis so

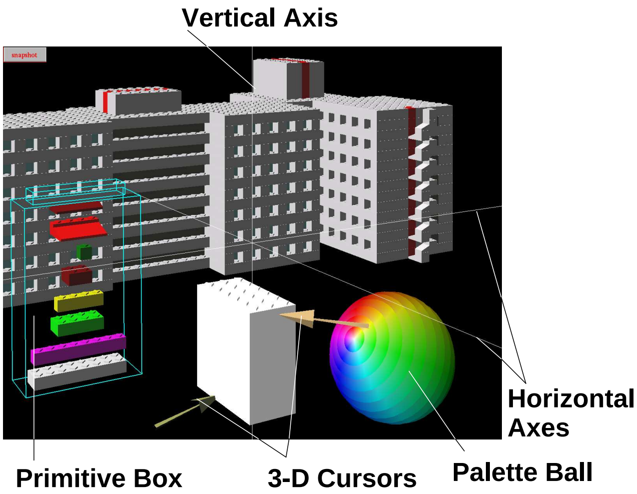

Figure 2: Software components.

that whole workspace can be observed and operated from a variety of view points.

Primitives
There are a few kinds of box- and wedge-shaped primitives. Height, width and depth of primitives are *n* (integer) cm long.Each primitive has studs which are evenly placed on its upper face with intervals of 1cm. Primitives are attachable to each other at a discrete position corresponding to the studs in the similar way as actual toy blocks. In this paper, *block* is a general term which means a single primitive or a composite of assembled primitives.

Primitive box
Primitive box is a movable wire-frame box containing a number of typical primitives. The user can generate primitives unlimitedly from the primitive box simply by picking the primitives in the box. The user can also delete a picked block by releasing it in the primitive box.

Palette ball
There is a pickable colorful palette ball in the workspace, of which hue, saturation and intensity changes gradually. The user can pick the palette ball and use it to color primitives with two hands.

One-handed Manipulations on Primitives
In VLEGO, as with real toy blocks, the user can easily designed 3-D objects using one or two hands by properly assembling primitives that have a few kinds of box- and wedge-shapes. The user can perform a number of functions on primitives as shown in Table 2. In this subsection, one-handed manipulations are explained. Two-handed manipulations are described in the later subsection after we explain a number of constraints used in two-handed interaction.

Table 2: Functions on primitives.

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

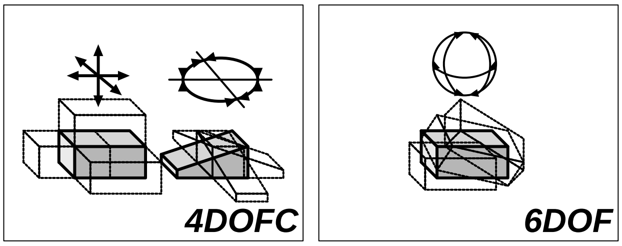

Figure 3: Two types of constraints.

Selection
When a tip of a 3-D cursor is positioned 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 is then unhighlighted. 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 later subsection.

Discrete Constraints and Operating Modes
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 grid 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, the user is able to arrange virtual objects accurately with ease. Left side of Figure 3 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 block can be operated freely as shown in right side of Figure 3.

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 4, S-1 and S-2). 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 and he/she would align each picked block with non-picked blocks placed around each picked block. 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 4, from S-1 to C-1.

Cooperative mode
Cooperative mode is selected when each hand picks a block and the distance between these picked blocks is short enough. Figure 4, C-1 and C-2 show this case. In this mode, then, it is natural to consider that the user would align 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 efficient. When one of the blocks is released or when 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 4, from C-2 to S-2. 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 arising from visual discontinuity.

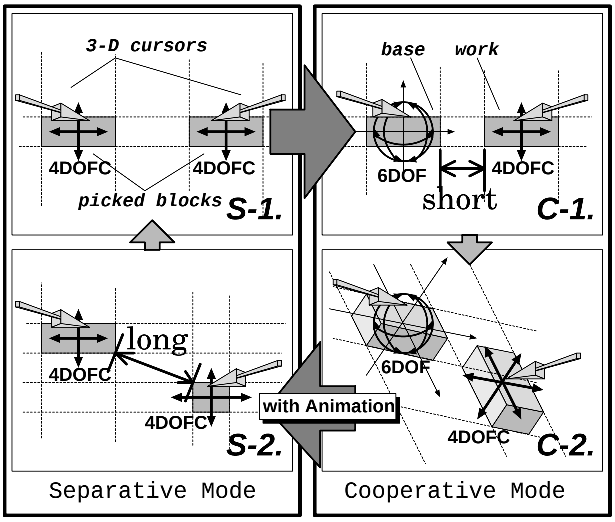

Figure 4: Transition of operating modes.

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 separative mode, all position relationships among picked and all other blocks is checked while only between two picked blocks is checked in cooperative mode. This restriction assures that all the subjects to be checked should be already aligned one another with 4DOFC in each case. As a result, the algorithm for collision detection is so simple to perform. The collision avoidance is performed as follows.

1. Let the picked block move toward the corresponding 3-D cursor in X direction (of the coordinate system of the current 4DOFC) 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.

Two-handed Manipulations on Primitives
In this subsection, a number of two-handed manipulations are explained.

Assembly of blocks
Assembling function utilizes discrete constraints and collision avoidance described above, and is most important to create 3-D objects. All primitives in VLEGO can be simply attached to one another 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. There are one-handed assembly and two-handed assembly according to the two operating modes.

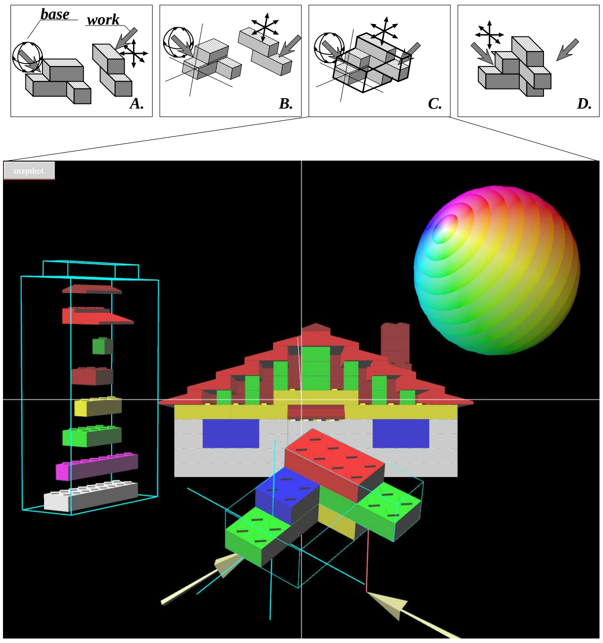

Figure 5: An example of assembly.

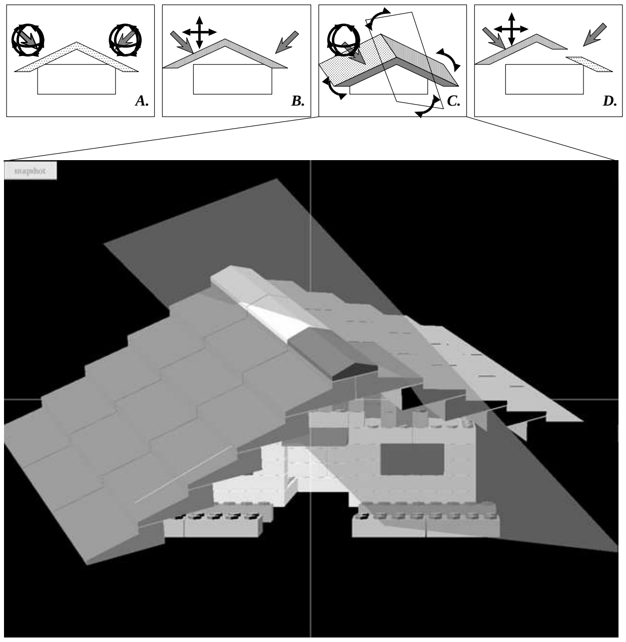

Figure 6: An example of decomposition.

In separative mode, the user assembles blocks by one hand. In this mode, the system detects collisions among picked blocks and all other blocks, then adjusts 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. All these blocks fit together if the picked objects are released.

In cooperative mode, the user assembles blocks by two hands. In this mode, the system avoids the interference between two picked blocks, however it does not check the collision and attachability among picked and non-picked blocks. In this case, two bounding boxes of both picked blocks appear if they can attach mutually as shown in Figure 5-C, and the two blocks fit together when one of them is released as shown in Figure 5-D.

Decomposition of blocks
There are two methods to decompose assembled blocks. First, the user can remove a primitive one by one with one hand from an assembled primitives picked by the other hand. This removal changes the operating mode from separative to cooperative. Second, the user can divide an assembled primitives into two parts using *a cutting plane*. Detailed process is as follows:

1. Pick a block (*base*) with one hand as shown in Figure 6, from A to B.
2. Press the first switch of the other hand (*work*) at any position where no object is selected.
3. Then the shape of the 3-D cursor of the *work* changes translucent square (namely, *cutting plane*) as shown in Figure 6, from B to C. Now, both the picked block and the cutting plane can be manipulated exceptionally with 6DOF.
4. Release the cutting plane at proper location and orientation, then picked block is divided into two blocks as shown in Figure 6, from C to D.

Coloring primitives
For coloring primitives, the user can choose any color using one hand (called *work*) from the palette ball picked by the other 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 7-**b**. While the ball is held, the user can select any color from a primitive or the ball according to the position of the *work*. Figure 7-**c** shows an example of selecting color from a primitive, Figure 7-**d** shows an example of selecting color from the palette ball. Once the user push the picking button, the current color is decided and the small cube colored the decided color appears at the tip of the *work*. Then the user can color primitives as shown in Figure 7-**e** to 7-**h** until the small cube or the palette ball is released.

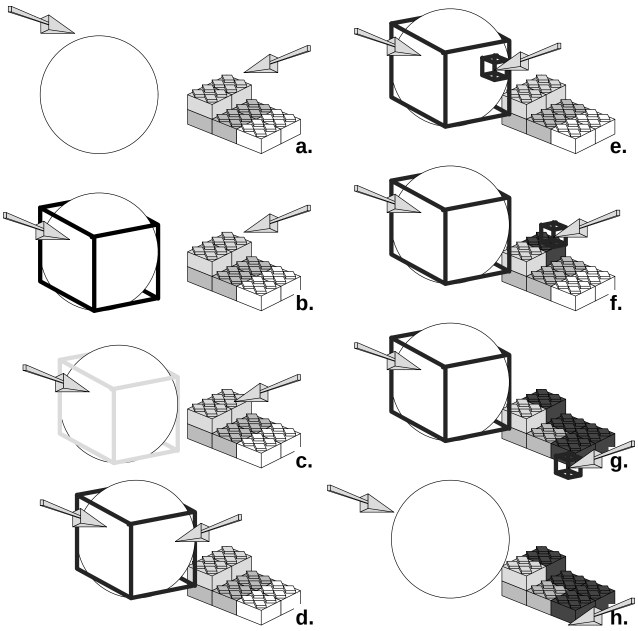

Figure 7: An example of coloring primitives.

Other two-handed operations

A picked primitive can be discretely scaled using two hands. For example, a white primitive is being scaled in Figure 2. In addition to cooperative two-handed operations, all one-handed operations can be performed concurrently by left and right hands.

EXPERIMENT EXAMINING THE EFFECTIVENESS OF TWO-HANDED INTERACTION

An experiment was conducted to evaluate the effectiveness of the two-handed interaction [13]. All activity in a virtual environment is based on a sequence of position-selecting task. For example, assembling task is performed by selecting a picking and a releasing points. A position selecting task is performed by three ways in VLEGO, by one hand, by two hands concurrently and by two hands cooperatively — a cooperative assembly, a decomposition and a color selection from the palette ball are example tasks using two-handed cooperative point selection. As a consequence of this, in order to validate the effectiveness of two-handed interaction with VLEGO, the experimental task should be designed that it can be performed with these three ways under the same criterion.

Taking account of this consideration, the experimental task was designed as follows: A block composed of 72 primitives as shown in Figure 8 was presented to subjects, and 18 of 72 primitives are colored red at random location while the rest are colored white at every trial. All 72 primitives can be seen by moving head position at the initial location. Each primitive turn its color red to white or white to red when clicked. Subjects were asked to click up all red primitives of the block so that the entire block turn out to be white by three ways as shown in Figure 9:**A.** click up by one hand, the block doesn't move. **B.** click up by two hands, the block doesn't move (separative operation). **C.** click up by one hand, and the block can be moved with 6DOF by the other hand (cooperative operation).

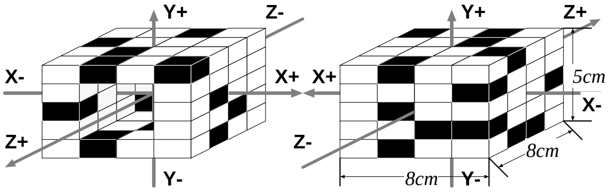

Figure 8: An initial state of the block used for the experiment.

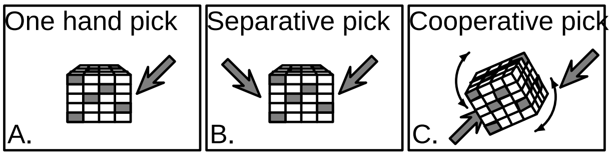

Figure 9: Three ways for picking primitives in the experiment.

Table 3 shows the mean task completion time, variance and error rate for the experiment. Here, error rate is a ratio of the number of clicks coloring white to red to the least number of clicks to be needed to perform the task (namely, 18). The analysis of variance about task completion time have shown the effectiveness of two-handed operations in position selecting with VLEGO ($\alpha = 0.05$). Separative operation surpasses one-handed operation because each hand can manipulate different primitives simultaneously. Cooperative operation surpasses one-handed operation because following three reasons: Firstly, the composed block can be moved with 6DOF so that the red primitives can be easily found. Secondly, as the composed block can be moved, the subjects have to move so slight their heads and the other hands that they can click the targets quickly. Finally, these two factors can be done concurrently. As a matter of fact,

Table 3: Mean task completion time, variance and error rate for the experiment.

Manipulating style	Mean task completion time (Secs.)	Variance	Error rate
A. One-handed	31.7	11.40	0.096
B. Separative two-handed	27.1	9.29	0.102
C. Cooperative two-handed	26.8	14.41	0.065

most subjects reported that this manipulating style is the easiest to perform the task among three manipulating styles. The error rate in cooperative two-handed style is improved in contrast to the error rate in one-handed style in certain extent. This implies one of the advantages of the cooperative two-handed operation, facilitation of point selecting. On the other hand, The error rate in separative two-handed style is not improved in contrast to the error rate in one-handed style.

CONCLUSIONS AND FURTHER WORK

VLEGO has demonstrated that flexible two-handed interaction based on actual 3-D designing activities enhances the usability of the virtual modeling environment. For example, it took only twenty minutes to create the school buildings of NAIST composed of 368 primitives shown in Figure 2. This high level of performance results partly from the separative and cooperative use of both hands as shown in the experiment for examining the effectiveness of the two-handed interaction. 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. Despite of the limited modeling ability, VLEGO is useful in applications in which 3-D objects have to be created to understand their appearance in a short time. Possible applications of VLEGO include architecture, education, training and amusement.

As further studies of virtual designing workspace employing simultaneous use of multiple hands, we focus on the following two themes for development and evaluation. One is on designing methods such as *virtual clay* that have high ability for creating complicated or freeform 3-D objects using two-handed interaction. The other concerns supporting methods for manipulating virtual objects with multiple users.

ACKOWLEDGEMENTS

This work is partly supported by Grant-in-Aid for Scientific Research under Grant Nos. 08234218 and 08555103 by Ministry of Education, Science and Culture. We also wish to thank the VRST '96 paper reviewers for their thoughtful comments.

REFERENCES

1. Butterworth, J., Davidson, A., Hench, S. and Olano, T. M.: 3DM: A Three Dimensional Modeler Using a Head-Mounted Display, *ACM SIGGRAPH: Proc. ACM Sympo. on Interactive 3D Graphics (GI '92)*, 135-139.
2. Halliday, S. and Green, M.: A Geometric Modeling and Animation System for Virtual Reality, *Proc. ACM Conf. on Virtual Reality Software and Technology (VRST '94)*, 71-84.
3. Sachs, E., Roberts, A. and Stoops, D.: 3-Draw: A Tool for Designing 3D Shapes, *IEEE Computer Graphics & Applications*, 11, No.6, 18-26, Nov. 1991.
4. Shaw, C. and Green, M.: Two-Handed Polygonal Surface Design, *Proc. ACM Sympo. on User Interface Software and Technology (UIST '94)*, 205-212.
5. 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)*, 213-222.
6. Kabbash, P., MacKenzie, I. S. and Buxton, W.: Human Performance Using Computer Input Devices in the Preferred and Non-Preferred Hands, *Proc. ACM Conf. on Human Factors in Computing Systems (INTERCHI '93)*, 474-481.
7. 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)*, 417-423.
8. Buxton, W. and Myers, B. A.: A Study in Two-Handed Input, *Proc. ACM Conf. on Human Factors in Computing Systems (CHI '86)*, 321-326.
9. Bolt, R. A. and Herranz, E.: Two-Handed Gesture In Multi-Modal Natural Dialog, *Proc. ACM Sympo. on User Interface Software and Technology (UIST '92)*, 7-14.
10. Mapes, D. P. and Moshell, J. M.: A Two-Handed Interface for Object Manipulation in Virtual Environments, *PRESENCE: Teleoperators and Virtual Environments*, Vol.4, No.4, 403-416, 1995.
11. Hinckley, K., Pausch, R., Goble, J. C. and Kassell, N. F.: Passive Real-World Interface Props for Neurosurgical Visualization, *Proc. ACM Conf. on Human Factors in Computing Systems (CHI '94)*, 452-458.
12. 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)*, 358-364.
13. Kiyokawa, K., Takemura, H., Katayama, Y., Iwasa, H. and Yokoya, N.: VLEGO - VirtuaL Environment for Generating Objects, *Proc. of SICE HI Sympo.*, 375-380, 1995 (in Japanese).