A VIRTUAL REALITY BUILDING BLOCK COMPOSER FOR ARCHITECTURE

J. BRADFORD, W. S. WONG, A. H. F. TANG
Department of Architecture, The University of Hong Kong,
Pokfulam Rd., Hong Kong
C. S. K. YEUNG
Department of Electrical and Electronic Engineering, The
University of Hong Kong

Abstract. Design is a complex and time consuming process. One way to simplify the design process is to use pre-build blocks for commonly known parts instead of creating them again with CAD. To give the designer an immediate 3D view of the design, designing in virtual reality is a good choice. This paper presents a virtual reality interface tool which allows a user to assemble an architecture structure from a library of pre-built blocks. The library is a distributed client-server database.

1. Introduction

The virtual reality environment described in this paper is provided by a commercial software, dVISE and dVS from Division Ltd. Originally used as a method of presentation for projects of architecture students, it provides a seemingly powerful programming environment which can be used to enhance the VR environment. The first enhancement that comes to mind is to perform design work directly in 3D environment instead of using virtual reality only as a demonstration tool.

As discussed by Liang and Green (1994), allowing a user to directly manipulate objects in 3D space is a good interface for 3D modeling system. For the VR software we are using, the user can interface with the VR environment with either traditional mouse or 3D mouse. However, the default setting of the VR software does not provide much support for design purposes. The composer will have connection to a distributed client-server database which will contain the pre-built blocks of architectural model. The database is the work of another research group.

An overview of the VR composer and the description of the functions added to the VR software will be given in section 2. The object inquiry interface is
described in section 3. A brief conclusion is given in section 4.

2. VR Composer

The composer is developed on a SGI Reality Engine 2 super computer. The backbone of the composer is the commercial software dVS and dVISE from Division Ltd. To view the environment, either a traditional computer monitor or a head-mounted display (HMD) can be used. To manipulate objects inside VR environment, a traditional mouse or other 3D input devices can be used in the current setting of the composer.

This research project originally intended to use a HMD as display device. However, we find that the HMD is not an ideal device for VR design work. As indicated by Liang and Green (1994), the physical burden of prolonged usage of the HMD is intolerable. Also, the resolution of the HMD used in our research is only marginally acceptable. These two problems can be solved with better technology.

When the decision of buying a VR software is made, the usage of the VR software is for visualization and demonstration of historical buildings and architectural designs of students in VR environment. A 3D mouse is adequate for these two kinds of application. For VR design work, a pair of gloves should be better. With a pair of gloves, some manual operations can be done by directly manipulating the objects, for example, scaling of objects. With only one 3D mouse, manual is more desirable for operation mode, but not for interface purposes.

2.1. Creation of Objects

All object creation are done through a manual, or toolbox. Since we only have one 3D mouse (six degree of freedom tracking device) at the moment, it would be difficult to create objects like JDCAD (Green and Liang, 1994). Instead, we use a toolbox to create primitive objects: polygon, sphere, cone, cylinder and cube. Circles are created with number of side of object set to 1, but they are actually a 50-side polygon. The same go for circular cone and cylinder. Semi-circles are created with number of side of object set to 2. The geometry object are created with normal of vertices set. All pre-built blocks and newly created objects are locked in yaw, roll and pitch. This is to prevent accidental rotation of objects while moving them with 3D mouse. Figures 1 and 2 are the 2D and 3D object creation toolbox.
Figure 1. 2D Object Creation Toolbox with a Triangle Created on back

Figure 2. 3D Object Creation Toolbox with a Sphere Created on back
2.2. SETTING STUB IN VR ENVIRONMENT

To make the placing of objects easier, the VR environment is set with invisible anchoring points (stubs). Any object that is moved will be translated a little bit such that its position will be aligned to the nearest stub. For example, if an object is moved to (14, 32, 26) after an operation and stub distances are 5 in all three direction, the object will be translated to (15, 30, 25). The distance between each stub can be set by the user via the toolbox. The default distance is 10mm in x, y and z direction. The stub distances in three direction can be different. The effect of stub can be toggled off inside the toolbox. Figure 3 show the toolbox of stub distance setting.

![Figure 3. The toolbox of Stub Distance Setting with Stub Distances = 1mm in all three direction](image)

2.3. RELATIVE ORIENTATION ADJUSTMENT OF OBJECT

This operation adjust the orientation of an object over another object, the base. The object selected will be rotated and, if required, translated in such a way that one of its patch surfaces (geometry are made up of series of triangles, or patches) is in contact with the base. For example, a chair is on the floor. The VR software does not care about the orientation of the chair. Thus the chair may be placed with only one of its legs touching the floor. After this operation, one of the patch
surfaces of the chair will be in contact with the floor. This operation is activated through a toolbox. Figure 4 show the toolbox.

![Figure 4. The toolbox of Relative Orientation Adjustment](image)

2.4. CONSTRAINT SETTING OF OBJECT

In the default toolboxes, one of them is used to set the constraints of objects in VR environment. However, HINGE constraint of objects cannot be set with that toolbox. A new toolbox is created such that hinge constraint of objects can be set. This toolbox is important because some objects require hinge constraint, for example, doors and windows. Figure 5 is the new constraint toolbox. Figure 6 is the toolbox for the additional constraints which we think should be useful. These are scale factor constraint. In some case, we want the scale factors (one or more) of an object to be invariant.
2.5. OBJECT MANIPULATION TOOLBOXES

The VR software has already allowed translating and rotating objects in VR.
environment using any 3D input device, and in our case, we use a 3D mouse. After an object is picked up using 3D mouse, the position and orientation of the object will be varied along with the hand, the representation of the 3D mouse in VR environment. However, due to errors (noise) in tracking the exact position and orientation of the 3D mouse, the hand appear shaking and thus make accurate translation or rotation of objects nearly impossible. Also there is no display of the current position and orientation of selected object in the VR environment. The traditional mouse can be used in the graphics user interface to change any attribute of any object, but the effect only appears after the button APPLY or ACCEPT has been pressed. Immediate visualization make some of modifications easier. For example, rotation of object is one of those. Thus, toolboxes for manipulating objects are made. Figure 7 is the toolbox for scaling operation.

![Figure 7. The Toolbox of Scaling Operation](image)

3. Object Inquiry Interface

A user can extract pre-built blocks, or objects from a object database through this toolbox interface. The user can browse into a object node to show what components are included under the object node, or choose different type of that object. After the user has made a decision, the chosen object will be sent to the VR environment and the user can make further adjustment like re-scaling or
translating.

Inside the toolbox named "Get Object" (Figure 8), a user can select objects inside the distributed database. The user can change the selection of an object under the item named "Object" by using the slider beside. The item named "History" stores the object hierarchy. The object hierarchy shows how the objects organize inside the database. For example, when a user chooses temple at first, he or she can then choose roof, body, platform, decoration, finish, and fixture under the object" item. If roof is chosen, the "History" item will become "temp.roof". Inside roof, the user can choose different kinds of roof. After the user has chosen one of the roof, the database will send that corresponding object to the VR environment.

![Figure 8: Object Inquiry Toolbox](image)

4. Conclusion

In this paper, the design interface of an VR composer is presented. We assume some architects may want to use pre-built objects to enhance their work. Originally intended to use immersive VR as design environment, but since commonly available HMDs are not suitable for our purpose, we use conventional monitor instead. Also, since only a 3D mouse is available, most functions of the composer are implemented in toolboxes first.

There are plenty of improvement can be made in the composer. The next few
improvements that have been planned are 1. display the data of object selected in design area, 2. implement the long range pick action, 3. export the design to VRML for another research group project, 4. change other attributes of object selected in design area, and 5. implement a virtual keyboard that allows the user to input the keyword for object inquiry. The improvements 1, 2, 4 and 5 are planned because if the technology is advanced enough such that immersive VR can be used, they will be very useful.

Reference

Tang, H. C., Ho, K. C., Li, W. L., Ho, C. M. : 1983, Chinese Temples on Hong Kong Island, Center of Asian Studies, University of Hong Kong
1996, dVS & dVISE for Unix Workstation, Division Ltd.