Human-Computer Interface in the CAVE

Oliver Schönbrunner
oliver.schoenbrunner@jk.uni-linz.ac.at

Department for Graphics and Parallel Processing
Johannes Keppler Univertity
Linz / Austria

1 Abstract

This document describes how a user can interfere with an application in a CAVE. It gives a description of the CAVE itself and of possible input devices. It shows methods to interact with a VR application, and problems that can occour thereby. It also tries to give solutions to this problems.

KEYWORDS: CAVE, Human-Computer Interface, HCI

2 Introduction

The interaction with an application should always be as natural (intuitive) as possible. That is valid for all applications be it an 2 dimensional desktop program or an application for an Immersive Virtual Environment, such as the CAVE.

The natural behavior, the natural interaction, with objects from the real environent and with reality itself, is very complex and has by far less constraints than the virtual reality, which is computed by computer system, which is limited itself. Even the fastest computers will not be able to cope with the (unlimited) freedom of ways to interact in reality.

It is the job of the programmer to restrict the possible interaction in a way that it is still intuitive but also in the range of the possibilities of today's computers.

3 CAVE (CAVE Audio Visual Environment)

Fig. 1 shows a model of a CAVE (recursive definition: CAVE Audio Visual Environment)

The Cave is a 3-meter cube whose floor and three walls serve as projection surfaces for stereographically (using shutter glasses) projected computer visualization. The user can submerse himself bodily into the virtual world. In front of and around him the user has a vertical and a horizontal panorama that compares to the real-world panorama. Multiple users can be in the virtual world simultaneously and communicate with one another. This has opened a completely new dimension

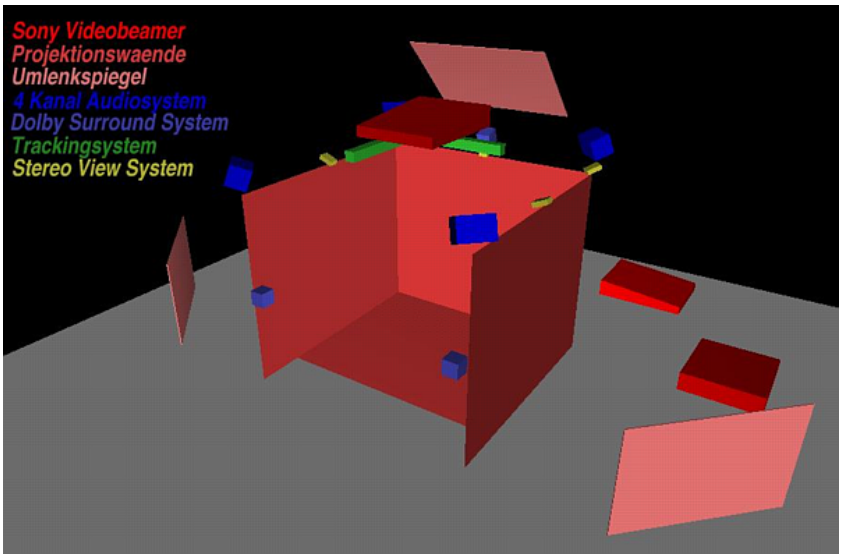

Figure 1: CAVE Audio Visual Environment

of applications that enables new ways of handling industrial and scientific problems with the use of VR and software-technology.

One user (the master-user) is wearing special shutter-glasses with position sensors applied to it. This way the CAVE application is able to locate the exact position of the user in the CAVE and can therefore calculate a projection based on this position. The three dimensional images the users will see then are generated in a way that the edges of the CAVE are hidden.

If there are more than one user inside the CAVE only the one with the master-glasses will get the optimal view. The others will see projections that are distorted on the edges of the CAVE. The farther away they are from the master-user the more distorted the image will be.

Motion Tracker (Position/Orientation Tracker) It is importatnt that the motion tracking system is fast an accurate enough that the computer can still calculate the resulting frames in real time.

At the University of Linz we are using a motiontracking system that uses both infrared and ultrasonics for its tracking.

Computer Performance To get an acceptable view of the VR the correct (regarding the perspective) frames have to be computed in realtime, that means at the very least 20 frames per second (the more the better). For the 3D viewing all frames have to be computed for every eyes, that is 3D systems need two times the calculation-performance that 2D systems require.

Both the motion tracker and the computer system have a hardware defined latency. To accomplish a VR environment that a user finds acceptable this latency has to be as small as possible. That means we need a computer with the best graphic capabilities and motion trackers that are both fast and accurate.

Stereo System Stereo viewing is achieved by using shutter glasses which will darken one eye in synchronisation with the projection of the frame for the other eye.

One Problem still remains: Even if every aspect of the generation of the projections are accomplished as well as possible. The stereo images are just projected to a flat screen, and altough each eye gets to see its own image the stereo effect will never be natural. (After all the images are just on a flat screen)

Audio System Altough the sound in a VR environment is an importent aspect, it does not have very much to do with the interaction (besides confirmation of commands and similar things).

4 Restrictions by the Hardware

Of course there are restrictions that cannot be overcome by software means. Even the best software cannot accomplish anything that is not possible with a given hardware.

The first an probably most important issue here is the performance (computing speed) of the computer system and its graphic hardware.

The next hardware restriction is the speed and precision of the motion tracking system. If the tracking is not accurate enough the resulting frames will be distorted and the 3D view will not be satisfying. Especially pointing at objects or grabbing them will be a hard or impossible task.

The size of the virtual space might be limitless (or just limited be the computational memory and speed) but the size of the real space is limited even in the largest hall. Therefore a usable CAVE has to be installed in a way that it uses the given space in the best possible way.

5 Active Interaction

5.1 Voice Recognition

A common belief is that interfaces to applications in the VR should be as natural as possible. It is assumed that the most realistic system will also be the most useful system. Even though commands seem inheritently unnatural, there are physical-world parallels. Since most of these involve giving orders to others using our voice, it is not surprising that one popular method of issuing commands in the VR is through voice recognition.

Problems with Voice Recognition Systems The worst problem of voice recognition is the nearly limitless degree of freedom in such systems.

There are such a variety of sounds produced by a single human voice, not to mention the variety of different voices from different users, that fast and accurate speech-recognition is very difficult to achieve. Algoritms are improving, but the most accurate still require training by each individual user before actual use. This is unacceptabe for systems that will have a large number of occasional and first-time users.

Secondly when the computer (the application) is trained to the voice of one individual user, the user himself has to be trained to know which commands are valid, which words (in what context) can be used to interact with the application.

Of course not every word the user knows can be known by the application and can be interpreted in the right way. The program only knows very few commands in a special context.

Another problem occours when the user is not alone inside the CAVE (as this is a multi user VR system). There is no way for an application to distinguish between words that are given to the program and words that are said to the other users. This will be problematic especially when words said during a conversation trigger commands in the application.

Voice Recognition Menues A solution to some of the given problems are voice recognition menues, which show to the user which commands are valid (and what function they will trigger)

5.2 Data Glove

Fig. 2 shows a typical Data Glove.

Figure 2: Data Glove

The Data-Glove is the device that most people think of when talking about VR input devices. It is the device that allows us to do everything in the virtual world that our real hand can do in the real world. The data glove is the translator from real to virtual. But there are too many degrees of freedom and too many possible device configurations, that the user simply has to come to the point where he fails

to produce the correct position (gesture) to trigger the desired response by the system.

There should be a system which allows the user to know which gesture triggers which command (without the need of a trainer who gives the user a detailed description of the interface). Of course some gestures are very intuitive: grabbing an object, pointing in a direction, pointing at a object. But more abstract commands are hard to know by intuition alone. What gesture should the user apply to move into the direction he is pointing at?

This problem reduces the powerful data glove just to a pointing device where all other commands have to be performed by some abstract finger movements.

5.2.1 Haptics - Force Feedback

A very helpful device for the interation in the VR are haptics, force feedback devices. Such devices make it possible that the user feels virtual object that he touches or grabs.

But until today such devices not very practical, they are too big or too heavy to be used. They will trouble the user more then they help him.

5.3 Pointing Device (Wand, 3D-Mouse)

Fig. 3 shows a 3D-Mouse.

Figure 3: 3D-Mouse

3D-mice (or similar devices like 3D-joysticks) are the most constrained input devices available for the VR. While the device itself has six degrees of freedom of motion (movement along the 3 axes and 3 degrees of freedom for rotation), its buttons have just one: they are either pressed or released. This constraint allows precise input to the system. It may not be as natural as a gesture or a voice-command, but it is accurate, and that is what is needed when interacting in the VR.

3D mice are usually tracked in 3D space. In effect, then, a 3D cursor is created. This produces another constraint problem, in the sense that the input with these devices will be inprecise (moving an object by moving the hand). Since VR is by nature 3 dimensional, the (physical) motion of the device cannot be constrained to 2 dimensions.

Possible Restrictions As we see later the input devices, whatever that will be, cannot be constrained by hardware the device must be constrained by software means.

5.4 Navigation - Interaction with the Scene

When a user sees himself in the VR he wants to navigate through the scene in the most natural way that is possible. Even in the largest CAVEs the freedom to walk around is limited by the borders (screens) of the CAVE. And secondly not all sides of the CAVE may have projection screens, normally there is no screen behind the user, but he may want to look behind him (or wants to see what is behind him)

We must find a way to navigate through the VR scene using the input device (3d mouse) that is available.

The first method that comes to mind, is to go in the direction that is pointed to by the mouse (point plus button press). This solution would be quite appropriate for VR environments like a head mounted display, but in the CAVE one will get some problems, if the user wants to go in a direction where there is no screen (backwards). There has to be a way to turn around without physically turning around, that means we need a way to rotate the whole scene.

That can be acomplished by using the trackball/joystick on the pointing device or by rotating the scene whenever the user is turning around far enough (e.g. whenever the user turns around more than 80 degrees, the scene will rotate into the opposite direction)

5.4.1 Restriction for Navigation

Our navigation through the real world is restricted by several means, one is for example gravitation, as much as someone may wish, we cannot simply fly away just by pointing our hands into that direction and say some magical spell. That is not so in the VR, we can move in every direction through the sky and even through walls or the ground. Sometimes that will be desireable (e.g. to look into some hidden parts of a machine) but usually we want to navigate in the VR as natural as possible that is walking just on the ground.

Another restriction we have in reality but we may not have in the virtual environment is the restriction of speed. In VR we can travel with any speed we wish, we can even jump from one point in space to another in no time at all.

Collision Detection Collision detection in the CAVE is a more serious and more complex problem than many people may see at once.

This problem results from the reason that the user can walk around inside the CAVE physically. And although that motion can be detected, there are no means to prohibit the user from walking through an object that is just virtually somewhere inside the CAVE.

One way to hinder the user from walking through objects could be to black out the whole scene whenever there is an unwanted collision.

Another way could be to make a warning noise that tells the user he has made a forbidden movement.

5.4.2 Physical User Motion

As the user can walk around inside the CAVE as he wishes and there are no means to prohibit this, we may restrict movement of the user by giving him another way to physically move around. This could be a bicycle he can use to navigate through the VR without moving around in the CAVE.

Another more complicated possibility would be a spherical CAVE. A CAVE that is not a cube but a sphere which rotates as the user walks through the virtual world.

5.5 Commands - Interaction with Objects

When a navigation is accomplished that is as natural/intuitive as possible the user may want to interact not only with the whole scene, but also with the objects in the VR. He may want to move objects, transform objects, group objects, create and delete objects, ...

Somehow the user has to know which objects he can interact with. Usually not everything that can be seen (to make to VR environment more natural there will be some decorational objects) can be interacted with (in contrary to the real reality, where everything that is in reach can be touched and therefore interacted with - if that is always desirable is another question).

The most intuitive possibility is to grab an object by intersectiong it with the input device. With a data glove the user can simply grab for an objcet just the way he would have done it in reality. With a 3D-mouse he will move the mouse inside (near) the object and then press a button.

Especially in the CAVE grabbing for objects is problematic as objects cannot be seen when one's hand (or the input device is near by). Remember that the objects only seem to be three dimensional inside the CAVE but they are just projections on the screen that is usually farther away. Or even worse the objects are behind a screen and there is no way to grab for these objects.

The most common solution for a virtual grabbing device is ray casting. A line extends the pointing device and objects that are intersected by this line are selected and can be interaced with. (Ray casting is very precise as the intersection between an object and the ray is always one single point.)

There must be a way for the user to know if he can interact with an object he is pointing at, he is grabbing (intersection with his input device). One method would be to highlight this object (change its color, put a semi-transparent bounding sphere/box around it), another could be to make a signal sound that tells the user it is ok to interact with this object.

The next thing that the user might want to know is, what he can do with this object. Can it be moved around, is it just a button that can be pressed a lever that can be pulled, ... Probably the object can be transformed in many different ways, it might be possible to change its color, to scale it, delete it...

A more accurate method than ray casting is casting a cone that will get wider the farther away it gets from the user, this way it will be easier for the user to interact with objects that are far away.

5.5.1 Object Positioning

Usually it is rather hard to position an object as accurate as it is desired (as it would be possible in reality). The main reasons for this are inaccurate position trackers and an inaccurate stereo viewing.

Large Scale Positioning Positioning an object at a large scale is a task that can be accomplished just by selecting and grabbing the desired object, then move it to the requested position (bring it into the requested orientation) and then just release it. What happens to the object after it is released is usually a question that has to be answerd for each individual application itself (can the object float in the air or will it drop to the floor, ...?)

Fine Placement A more precise positioning can be accomplished by attaching (virtual) handles to an object that allow the user to rotate/move the object just along one axis (or in one plane). The idea is to limit the freedom of movement.

To achive an even finer placement the user at first has to specify a vector (straight line) along which the movement can be performed and then with some kind of controller the distance (in centimeters, or even millimeters) that the object should move. (In a similar way rotations can be implemented, by specifying the plane (normal vector - axis) and the angle of the rotation)

5.5.2 Object Manipulation

To accomplish tasks that are special to the VR (or are done in a very different way than in reality), new means of interaction have to be found. Such tasks include scaling, coloring or grouping objects, creating or deleting objects are also jobs that usually cannot be done in the real world.

In the CAVE these tasks are done by selecting an object and then pressing a certain mouse button, which opens a window where the right command has to be selected (again bei pointing at the menu item and then pressing a mouse button).

Object creation can be done in very different ways. One would be that only a few special objects can be created (for example pieces od furniture for an interior decoration application). Or new objects can just be created out of some primitives (like spheres and cubes) which can be manipulated (scaled, colored) to get the desired compound.

The most complicated but also most flexible way to create a new object in the virtual world is by specifying its vertices in the tree dimensional space just by pointing at that position. (or even creating 3d-curves by sweeping the input device through the air)

5.5.3 General Commands

How can the user specify commands that are more abstract and not connected to any object in the VR. Commands for loading, saving or exiting the program.

Such commands are best achieved via two dimensional menues (e.g. virtual pull-down menues). When the user presses the designated button a menue (or even a menue structure) appears where these commands can be selected by pointing to the entry in the menue.

6 Passive Interaction

With passive interaction all interaction is meant that is not performed by the user via his input device. This interaction is mainly the movement of the user inside the CAVE which is tracked by the motion tracking system.

This interaction is done nearly all the time, everytime the user walks around or just turns his head. It is important to note the relevance of ths interaction as only by correct motion tracking and its correct evaluation an exact projection of the VR can be calculated.

An improvement to motion tracking could be accomplished by an eye tracking system. If the application could not only track the position of the user but also exatly where he is looking, even more realistic frames could be rendered.

7 Summary/Conclusion

This document described the possibilities we have to interact in the VR using a CAVE as the interface between reality and virtuality. We could see that there are a lot of possibilities to accomplish this interaction, but only a few are practical enough to be used.

Still there is much to be done to find better ways for the VR interaction, and to improve the interaction methods we have.

Of course there will be many new possibilitys when technology evolves, especially in voice recognition will be done very much. But the computer systems of today are just too limited to let the borders between our real world and the virtual reality vanish.

References

[1] Larry F Hdges Doug A Bowman. *User Interface Constraints for Immersive Virtual Environment Applications*.