3D Interaction in Virtual Environment

Jan Flasar
flasar@fi.muni.cz

Faculty of Informatics
Masaryk University
Brno / Czech Republic

Abstract

Human-Computer Interaction in 3D space should enable users to interact freely with virtual objects. Interaction in 3D and direct manipulation with virtual objects bring new problems. Forthcoming virtual environment applications require efficient techniques for faster 3D interaction.

In this paper we present short review of interaction techniques, which are used in VE applications. Then we state goals of our research in 3D interaction and list user interfaces used in applications of that type. We mention experiments realized in our HCI laboratory.

KEYWORDS: 3D interaction techniques, 3D user interfaces, virtual environment, manipulation techniques, selection, grabbing

1 Introduction

In virtual environment (VE) various type of interaction can be used. Typical task in VE is a manipulation with virtual objects. Each VE task can be realized in different manners. Direct manipulation in three dimensions is difficult when we are limited by 2D conventional graphical interfaces such as mouse or tablet.

One of the reasons that makes human-computer interaction (HCI) in VE so laborious, is that user has no haptic contact with objects situated in virtual world. This imperfection causes problems particularly in cases when we access VE using a *real-world metaphor*. In real world users can touch only objects that are within the reach of their arms.

Therefore we seek *interaction techniques*, which help us to interact in VE applications correctly, rapidly and without unnatural limitations. Most of these techniques fall in four categories:

- system controls
- grabbing and manipulation of remote objects
- travel in virtual environments

- user interfaces with constraints

In subsequent sections we will describe individual ways of adduced types of interaction which ease work within VE. We will review user interfaces that are used for interaction with virtual reality. Especially, we will aim at various kinds of trackers and devices for visual and force feedback.

2 System control techniques

Virtual environment applications use different kinds of control. We often need to stimulate the change of system state, leading to appropriate change of currently running action to another needed at given time. Following real-world metaphor, in VE we must move close to object to be able to interact with it. For example, if we want to manipulate object, that is located out of our reach distance, then at first we must come near to object, to such a distance that allows us to grab object and manipulate it. To facilitate it we have to stop processing of current event and invoke travel action. Similarly we may need to set some other parameters of VE application. It means that besides movement we also have to communicate with VE application (eg. announce that we want to transfer object to a new position).

There are many methods how to control VE-based applications. At present, most of VE applications are controlled via Graphical User Interface (GUI), voice and gesture commands combined with some logical tools. In the following we will adduce the taxonomy of system control techniques and then describe GUI-based methods.

2.1 Taxonomy

According the way we perform the change of system control, these techniques are divided in four categories.

- **GUI based system -** the most frequent solution to reach the change of system control. Using menu and others GUI instruments, we can invoke a change or to influence some settings, that we need at given time. This method is widely used in 2D desktop applications. In virtual environment we implemented 1D menu, 2D and 3D GUI. Each have advantages and also disadvantages.
- **Voice command -** system is driven by commands that result from voice recognition process.
- **Gestural interaction -** uses gesture recognition made i.e. for fingers position or hand motion.
- **Tool -** may be either *physical* or *virtual*. Physical tools take e.g. the form of pedal or wheel, logical tools mostly in graphical form, offer many different

solutions (virtual avatars of physical tools or specific tools for motion, fly etc.)

Naturally, we can use any combination of these techniques to improve interaction with VE application, because , depending on concrete situation, each technique may be more useful then others.

2.2 GUI system

First of these techniques is one-dimensional (1D) menu [3]. It enables to select one item from limited count. Input can be realized eg. by detecting the change of position or orientation of user's hand, but in this configuration we can use only one degree of freedom. Item is selected when the position/orientation of user's hand is situated in selecting region, which is defined as region bounded by two positions or angles. We emphasize, that the size of selecting region must be such, that we could differentiate safely between separate selecting regions. Considering above criteria, 1D menu has disadvantage in limited count of options/items, which could be handled this way.

If the number of options/items to select grows above reasonable count, then it cannot be fitted within 1D menu. The next logical step is to implement 2D GUI. In two dimensions, we can use all GUI instruments. For selection user's hand positions in 2D plane may be used. This interaction method is analogous to interaction commonly adopted in conventional 2D desktop applications.

The last step is to add the third dimension which brings both power and problems. On the one hand, using 3D GUI allows users to create 3D widgets or sliders, which can be employed for helpful and better manipulation with VE objects. On the other hand, adding the third dimension makes it more difficult for user to interact with 3D GUI.

2.2.1 3D GUI implementation issues

One of the most important problems is proper GUI's location in virtual environment. The first possibility is to have GUI floating in the space [3] as other virtual objects. Anytime a user needs to change GUI's position, he must grab it and move to a new location. But a user can forget, where is 3D GUI situated and it means that he would spend too much time by searching for it in VE space. Useful enhancement is to limit GUI location so that it will be always situated in front of user view. This improvement will solve the problem when user loses knowledge about 3D GUI location, but on the other hand it shields view into scene. This disadvantage may be solved very simply by using some gesture with that we can show/hide GUI. But basically,there are many manners, how we may locate GUI and what gesture is more suitable for it's activation. The next problem is the selection of item in 3D GUI. This selection can be done using eg. 2D pointing, virtual touch (with visual feedback) or ray-casting technique (Figure 1). To confirm selection we can also use some gesture.

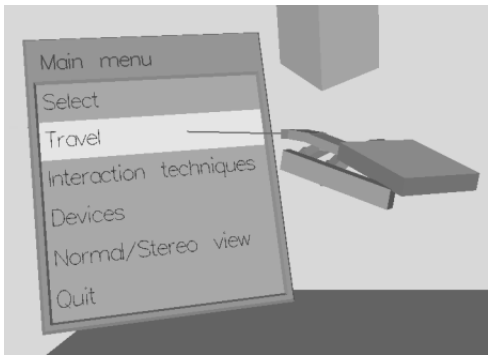

Figure 1: 3D menu - item selection with ray-casting technique.

3 Interaction techniques

Interaction techniques provide the means for easy work with objects in VE. It is also important for these techniques to be easy to learn and use. Their description will be presented in three parts: selection and grabbing, manipulation and travel techniques.

3.1 Selection and grabbing techniques

Before grabbing the object we want to manipulate with, it is necessary to select it. Selection process requires two things:

- instrument/technique with that we make selection
- signal/command to commit (eg. button press, gesture)

Selection techniques fall into following three categories: direct selection, remote selection and discrete selection.

3.1.1 Direct selection

In this selection mode, object are selected by contact of cursor (typically, cursor is user's hand) and object or it's close neighbourhood, which can be eg. bounding volume that is used as the first approximation for collision detection. In this case it is not necessary to confirm selection because visible collision between cursor and object can be regarded as unambiguous confirmation of selection.

If we work with object in virtual world in natural manner, we can interact easily only with objects, which are located within user's arm reach. Problem is coming up when user wants to interact with remote object. It happens when distance between object and user is larger than the (physical) length of user's arm. Under circumstances we must move more closely to object and grab it. Problem is solved using techniques, which are stated in literature as *arm-extension* or *local* selection

techniques. These techniques use non-realistic selection when user's virtual arm is able to dynamically grow to desired length (opposite to user's physical arm). Thus the selection of remote object can be made by virtual hand that is always long enough. The technique with visual feedback provides natural mapping (though nonlinear) of physical movement to virtual movement of arm. Basically, we can say, that arm-extension technique makes remote objects manipulation simpler and faster, because we may interact with them by natural hand and arm motions.

One of these techniques is *Go-Go technique*(Figure 2) that was published in [4]. A local region with perimeter D is defined around user. Until the user's hand stays in this region virtual hand moves in one to one correspondence with physical hand (using some linear mapping). If the physical hand leaves local region then the virtual hand begins to move outwards faster than the physical hand. Go-Go technique allows the user to interact with remote objects (without precise manipulation) and in a local region it allows delicate manipulation. Length of virtual arm R_v is calculated using non-linear mapping function F (see Figure 2), eg. that is explained in [4]:

$$R_v = F(R_r) = \begin{cases} R_r & if R_r < D \\ R_r + k(R_r - D)^2 & otherwise \end{cases},$$

where R_r is length of physical arm, R_v is length of virtual arm and D is distance where a linear part of function is applied This function is designed to ensure smooth transition between linear and non-linear part.

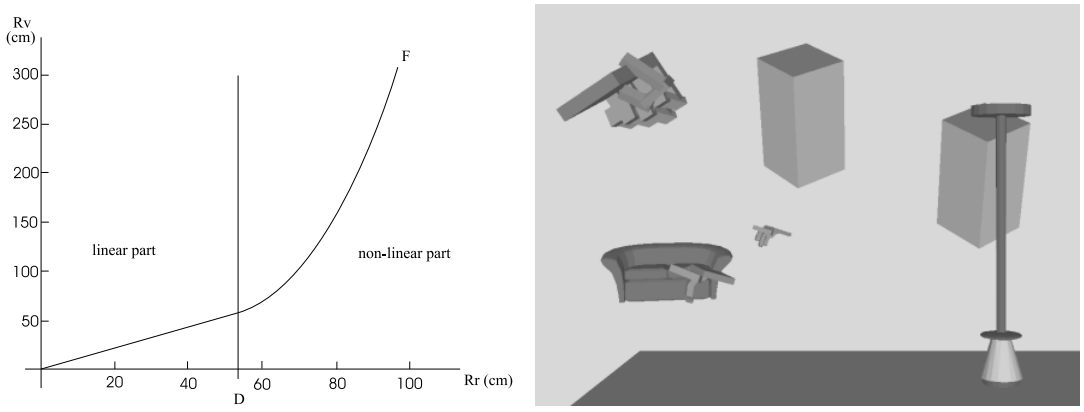

Figure 2: *Go-Go technique*: example of mapping function (reproduced from [4]) and snapshot of our experiment. Hand miniature is in position that users hand would take in real-world. Virtual hand translates to position computed with non-linear function F applied to world coordinates of hand.

Modified version of previous method is technique *"fast" Go-Go* [2], which has no local region and more rapidly growing mapping function. However, Go-Go technique has a finite range still, which is defined by mapping function. Then for various VEs we must correct function F so that it would be possible to grab most of the objects in scene. This disadvantage is solved using other modification, which is

called *"stretch" Go-Go*, that divides area surrounding user into three concentric regions. In the inner region, arm length is retracted, in the middle region it remains the same and in the outermost region arm stretches out with constant speed. The next modification can be the installation of signal, which allows to stretch and retract arm length (eg. using two buttons). This technique is called *indirect stretch Go-Go*.

3.1.2 Remote selection

This method attacks the selection problem by means of direct selection. Remote object selection can be realized using ray [2, 3], which is directed out of user's hand. Ray direction can be determined by user's finger, hand or head orientation. Selection is realized if ray intersects some object and the system receives command confirming object's selection. Again, this technique has disadvantage in the case when we want to select an object of very small size, that is very difficult to locate with ray. This disadvantage may be solved using so-called *spotlight*[1]. It employs a light cone allowing user to select very small objects aimed by directed and visible light cone.

3.1.3 Discrete selection

Next alternative is the selection based on preliminary specification of object identity e.g. with some *name*. This explicit information (usually in non-graphical form) is used for direct object identification [3], without any contact with to-be-selected object. This identification can be realized by following ways:

- list selection - it can be made and included in GUI instrument
- usage of voice recognition
- direct textual input

The advantage of this method is the possibility to select object that cannot be seen.

3.1.4 Grabbing techniques

When we use a direct selection then we can interpret the moment of selection also as the statement to grab object. In the case of two following selection techniques we must use some signal to confirm the grasp of object. With grabbed object we have a possibility to manipulate it in three ways:

- *remote manipulation* - hand and object necessarily do not touch in virtual world

[1] This method is also known as *cone-casting technique*

- *object transfer method* - object (and its neighbourhood part of scene) is moved to cursor position, manipulated, and when released, it moves back (see Figure 3)
- *hand transfer method* - virtual hand moves to object's location and manipulation is done there in proper scene context (see Figure 4)

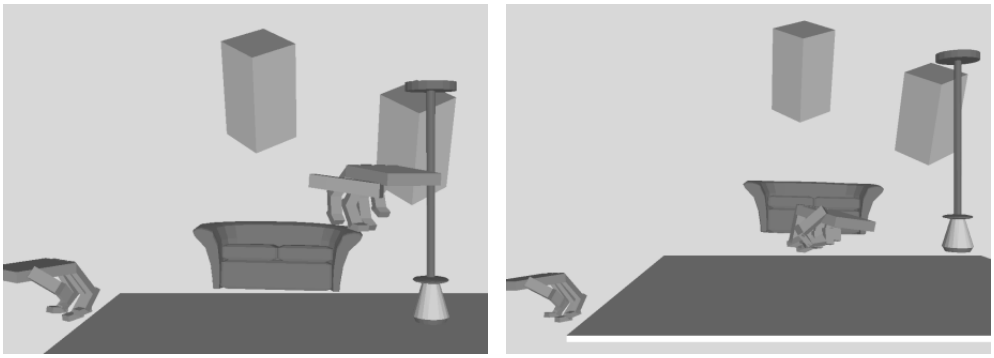

Figure 3: Left figure shows situation before object seizing using *object transfer method*. Right figure shows situation after object seizing. Object and scene were transfered close to users hand.

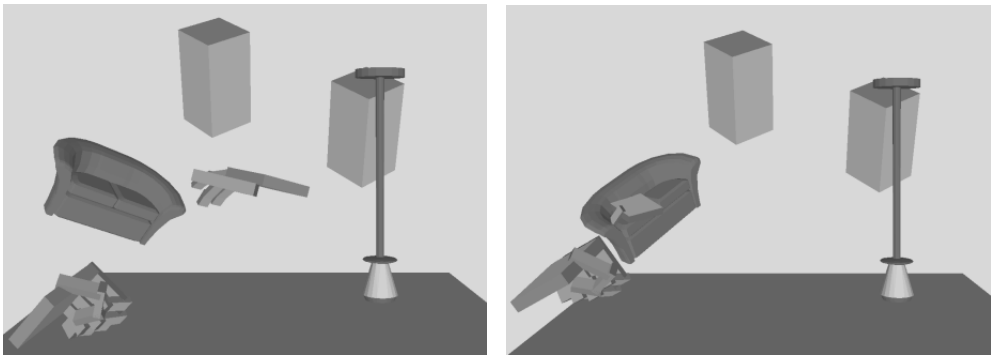

Figure 4: On the left is situation before object seizing using *hand transfer method*. On the right is situation after object seizing. Users hand moves close to object and manipulation is done there.

3.2 Manipulation techniques

One of the most important forms of interaction is object positioning in virtual world, mainly the specification of object position and orientation. This interaction can be realistic or non-realistic. With realistic interaction, user grabs object and manipulates it as well as if he manipulated it in real world (see Figure 5). Virtual hand moves in 1:1 correspondence with user's physical hand. Problem appears when the user wants to manipulate object that lies outside of the reach distance of

his hand. This problem is solved using arm-extension technique already described in section 3.1.1.

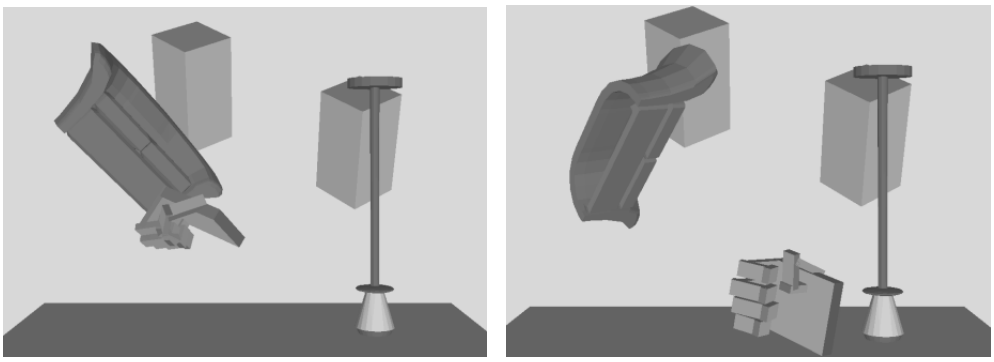

Figure 5: Object manipulation as if done in real world is shown in the left figure. Figure on the right side shows operation of turning object around its center controlled by rotation of user's hand.

The next technique is *ray-casting technique* [2, 3]. With object we manipulate by ray. If we want to translate object we can do it easily with hand motion. But object rotation is difficult with this technique. To rotate object is easy only around ray axis. If we want to rotate around more axes we must employ some signal again (i.e. two buttons, giving the choice to rotate around an additional axis).

The next technique is WIM (World In Miniature) [5]. User sees the miniature copy of original scene and manipulation is realized in this synoptic world. The technique is appropriate particularly when we work with large objects that block out its surroundings. For precise manipulation it is necessary to have the possibility to set a constraint that narrows the degree (range) of freedom. For example we want to restrict a motion only to XY plane or along Z axis and to set the possible range of movement limits. Good aids are co-located widgets and sliders, that allow to set the constraints easily.

3.3 Travel techniques

Before we begin to walk in virtual environment we must choose appropriate travel method. At first, we must determine if our tour will be driven by some goal of motion, where we want to transfer, or we just choose a direction of our motion. Then we must set speed/acceleration with which we will move. And finally, we must declare input conditions that specify beginning, duration and end time of travel motion.

Taxonomy of travel techniques can look like for example in this way [1].

1. **Direction/Target Selection**
    - Gaze-directed steering
    - Cross-hairs mode

- Pointing/gesture steering (including props)
- Discrete selection
  - Lists (eg. menus)
  - Environmental/direct targets (objects in the virtual world)
- 2D pointing

2. **Velocity/Acceleration Selection**
   - Constant velocity/acceleration
   - Gesture-based (including props)
   - Explicit selection
     - Discrete (1 of N)
     - Continuous range
   - User/environmental scaling
   - Automatic/adaptive
3. **Input Conditions**
   - Constant travel/no input
   - Continuous input
   - Start and stop inputs
   - Automatic start or stop

4 User interfaces

In our HCI lab we work with several devices, that provide different means of interaction between human and computer. These devices facilitate space location, visual feedback, motion capture, hand gesture input and simple force feedback. Devices are:

- Gloves
  - 5th Glove - data glove sensoring the bending of 5 fingers (in range 0-255) plus pitch & roll information
  - Pinch Glove - left/right pair of gloves detecting contacts among finger tips
- Trackers
  - Polhemus Ultratrack with 4 6DOF sensors (motion tracking up to 10 meters)
  - Logitech Ultrasound Trackers - 3D mouse, triangles, Crystal Eyes VR (position tracking up to 2 meters)

  - trackers from IO glasses (3DOF - orientation in Earth space)

- Stereo Vision
  - Crystal Eyes Bundle, Crystal Eyes VR - polarised stereoglasses, optionally with Logitech ultrasound tracker
  - IO glasses - LED shutter stereoglasses, helmet with earphones and 3DOF Earth space orientation tracker
  - CyberEye helmet
- Force Feedback
  - PHANToM 1.0
  - torsion feedback - under development

5 Experiments in virtual environment

At present, to run VE applications we have been developing in our laboratory, we use PC (dual Celeron/400 MHz and accelerated graphics card Matrox G400) with Linux. Visualization part including stereo uses OpenGL. With this configuration we achieve average speed 20fps in stereo display mode of the scene with 8000 triangles. For interaction experiments we have used following hardware. Polhemus was used to track the position and orientation of user's hands. 5th Data Glove enabled to detect fingers motion. Original glove sensor was replaced by Polhemus sensor to get more precise data of hand orientation. Polhemus sensor was used to control origin of selection ray. Ray direction was derived from the direction of index finger on the right glove. Currently we have connected left and right glove and we may test more complex manipulation techniques in future. Crystal EYEs were used for visual feedback. They provide smooth and natural depth view into scene. Typical VR session is shown in Figure 6.

Figure 6: Experimental session in VR laboratory

6 Conclusions and future work

Interaction techniques are very important for better human-computer interaction. Without these techniques, constraints application would be controlled difficult. Interaction techniques described in this paper, are implemented in step-wise manner into our VE applications programming system.

In future we will continue the research of 3D interaction techniques introduced in this paper. We will focus on design and evaluation of new techniques and efficient combination of such techniques. We intend to find the easy-to-learn-and-use VR techniques. We plan to include other devices such as PHANTomM for force feedback, helmets or 3D mouse into VR manipulation system. Goal of our work is the creation of such instruments that would allow users to interact with virtual environment as naturally as in real world.

Acknowledgements

I would like to especially thank Jiří Sochor for his advice as well as his many helpful comments and suggestions. This work was supported by a research grant, from Grant Agency of Czech Republic, Contract No. GACR 201/98/K041.

References

[1] D. Bowman, D. Koller and L.F. Hodges. "Travel in Immersive Virtual Environments: An Evaluation of Viewpoint Motion Control Techniques." to appear in *Proceeding of Virtual Reality Annual International Symposium (VRAIS)*, 1997.

[2] Bowman, D. and L.F. Hodges. "An Evaluation of Techniques for Grabbing and Manipulating Remote Objects in Immersive Virtual Environments." *Proceeding of the 1997 Symposium on Interactive 3D Graphics*, Providence, RI, ACM: 35-38.

[3] M. Mine. "Virtual Environment Interaction Techniques." *Proceeding of the 1997 Symposium on Interactive 3D Graphics*, University of North Carolina Computer Science Technical Report TR95-018, 1995.

[4] I. Poupyrev, M. Billinghurst, S. Weghorst, and T. Ichikawa. "The Go-Go Interaction Techniques: Non-linear Mapping for Direct Manipulation in VE." to appear in *Proceeding of the ACM Symposium on User Interface Software and Technology (UIST)*, 1996.

[5] R. Stoakley, M. Conway, and R. Pausch. "Virtual Reality on a WIM: Interactive Worlds in Miniature." *Proceeding of CHI*, 1995, pp. 265-272.