Implementation of Flying, Scaling, and Grabbing in Virtual Worlds

Warren Robinett
Richard Holloway

Department of Computer Science
University of North Carolina
Chapel Hill, NC 27599-3175

ABSTRACT

In a virtual world viewed with a head-mounted display, the user may wish to perform certain actions under the control of a manual input device. The most important of these actions are flying through the world, scaling the world, and grabbing objects. This paper shows how these actions can be precisely specified with frame-to-frame invariants, and how the code to implement the actions can be derived from the invariants by algebraic manipulation.

INTRODUCTION

Wearing a Head-Mounted Display (HMD) gives a human user the sensation of being inside a three-dimensional, computer-simulated world. Because the HMD replaces the sights and sounds of the real world with a computer-generated virtual world, this synthesized world is called virtual reality.

The virtual world surrounding the user is defined by a graphics database called a *model*, which gives the colors and coordinates for each of the polygons making up the virtual world. The polygons making up the virtual world are normally grouped into entities called *objects*, each of which has its own location and orientation. The human being wearing the HMD is called the *user*, and also has a location and orientation within the virtual world.

To turn the data in the model into the illusion of a surrounding virtual world, the HMD system requires certain hardware components. The *tracker* measures the position and orientation of the user's head and hand. The *graphics engine* generates the images seen by the user, which are then displayed on the HMD. The *manual input device* allows the user to use gestures of the hand to cause things to happen in the virtual world.

BASIC ACTIONS

An *action* changes the state of the virtual world or the user's viewpoint within it under control of a gesture of the hand, as measured by the manual input device. The hand gesture initiates and terminates the action, and the changing position and orientation of the hand during the gesture is also used to control what happens as the action progresses.

The manual input device may be a hand-held manipulandum with pushbuttons on it, or it may be an instrumented glove. In either case, the position and orientation of the input device must be measured by the tracker to enable manual control of actions. The input device must also allow the user to signal to the system to start and stop actions, and to select among alternative actions.

Certain fundamental manually-controlled actions may be implemented for any virtual world. These actions involve changing the location, orientation or scale of either an object or a user, as shown in Table 1.

	User	Object
Translate	fly through the world	grab (and move) object
Rotate	tilt the world	grab (and turn) object
Scale	expand or shrink the world	scale object

Table 1. Basic actions

Flying is defined here as an operation of translating in the direction pointed by the hand-held input device, with steering done by changing the hand orientation. This is different from the type of flying available in a flight simulator, where the user can not only translate but can also cause the virtual world to rotate around him by banking. However, translation-only flying is appropriate for a HMD because the user has the ability to turn and look in any direction, and to point the input device in any direction. We believe that keeping the orientation of the virtual world locked to that of the real world helps the user to navigate while flying through the virtual world.

Tilting the world is the ability to re-orient the virtual world relative to the user's orientation; that is, to turn the surrounding virtual world sideways. This is implemented by rotating the user with respect to the virtual world, which is subjectively perceived by the user as the entire virtual world rotating around him.

Scaling the world is the capability to shrink or expand the world relative to the user, as occurs to Alice in Wonderland when she drinks from the little bottle or eats the little cake. By setting up the action code properly, the user can shrink and expand the world while manually steering the center of

expansion. This enables a powerful method of travel in very large virtual worlds: the user shrinks the world down until the *destination is within arm's reach and then expands the world*, continuously steering the center of expansion so as to arrive at the correctly-scaled destination.

Grabbing an object is picking up and moving a simulated object that appears in the virtual world. By analogy with real-world grabbing of objects, this includes the ability to rotate the held object before releasing it.

Scaling an object is just shrinking or expanding an individual object alone.

This paper seeks to answer the following question: How can the basic actions of flying, grabbing, scaling and tilting in a HMD system be specified and implemented?

PRIOR WORK

The first HMD was built in 1968 by Ivan Sutherland [8], but since it had no manual input device other than a keyboard, it did not allow actions controlled by manual gestures. At the University of Utah, a tracked manual input device called a "wand" was added to the system [9]. The tip of the wand was tracked in position but not orientation. The wand was used to deform the surfaces of virtual objects composed of curved patches [2].

In 1985 at NASA Ames Research Center, McGreevy and Humphries built a HMD which was later improved by Fisher, Robinett and others [3]. Under contract to NASA, VPL Research provided an instrumented glove, later named the "DataGlove," which served as a manual input device. The position of the hand and head were tracked with a Polhemus 3Space magnetic tracker. In 1986 using the glove input device, Robinett implemented on this system the actions of flying through the world, scaling the world, rotating the world, and grabbing objects.

Some of these actions, particularly flying and grabbing objects, have since been implemented on HMD systems at several sites. VPL Research began in 1989 selling commercially a HMD system that used a glove to control the actions of flying and grabbing [1]. At the University of North Carolina [7][5], the actions of flying, scaling and grabbing were controlled with a hand-held manual input device with pushbuttons on it which was made from a billiard ball.

COORDINATE SYSTEMS DIAGRAM FOR A HMD

Various coordinate systems co-exist within a HMD system. All of these coordinate systems exist simultaneously, and although over time they may be moving with respect to one another, at any given moment each pair of them has a relative position and orientation. The instantaneous relationship between two coordinate systems can be described with a transform that converts the coordinates of a point described in one coordinate system to the coordinates that represent that same point in the second coordinate system.

Although transforms exist between any pair of coordinate systems in the HMD system, certain pairs of coordinate systems have relative positions that are either constant, measured by the tracker, or are known for some other reason. These are the *independent transforms*, which are shown in relation to one another in Figure 1. In this diagram, each node stands for a coordinate system, and each edge linking two nodes stands for a transform between those two coordinate systems.

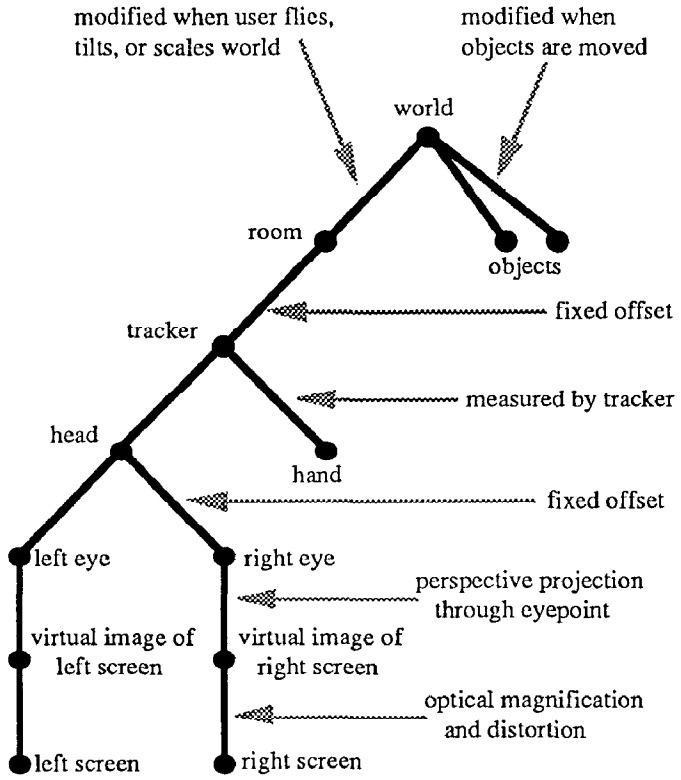

Figure 1. Coordinate systems diagram for a single-user HMD system

NOMENCLATURE FOR TRANSFORMS

We abbreviate the coordinate systems with the first letters of their names. The World-Object transform may be written as T_{WO}. Transform T_{WO} converts a point P_O in coordinate system O to a point P_W in coordinate system W.

$$P_W = T_{WO} \cdot P_O$$

This notation is similar to that used in [4]. Notice that the subscripts cancel nicely, as in [6]. Likewise, the composition of the transform T_{WO} going from O to W with the transform T_{RW} going from W to R gives a transform T_{RO} from O to R, with the cancellation rule working here, too:

$$T_{RW} \cdot T_{WO} = T_{RO}$$

The inverse of transform T_{WO} is written T_{OW}.

SPECIFYING ACTIONS WITH INVARIANTS

An action in a virtual world is performed by activating the input device, such as by pushing a button, and then moving the input device to control the action as it progresses. As an example, grabbing a simulated object requires, for each frame while the grab action is in progress, that a new position for the object be computed based on the changing position of the user's hand.

It is possible to precisely define grabbing and other actions with an *invariant*, which is an equation that describes the

desired relationship among certain transforms involved in the action. The invariant is typically stated as a relation between certain transforms in the current display frame and certain transforms in the previous frame. In the case of grabbing, the invariant to be maintained is that the Object-Hand transform be equal to its value in the previous frame while the grab action is in progress; in other words, that the object remain fixed with respect to the hand while it is being grabbed.

Starting from the invariant and a diagram of the coordinate systems involved, a mathematical derivation can be performed which produces a formula for updating the proper transform to cause the desired action to occur. For grabbing, this would be updating the Object-World transform to change the object's position and orientation in the virtual world.

Rigorously deriving the update formula from a simple invariant is much easier and more reliable than attempting to write down the update formula using the coordinate systems diagram and informal reasoning. Also, the matching of adjacent subscripts in the notation helps to check that the transforms are in correct order.

GRABBING AN OBJECT

To derive the update formula for grabbing, we first look at the relevant part of the coordinate system diagram, shown in Figure 2.

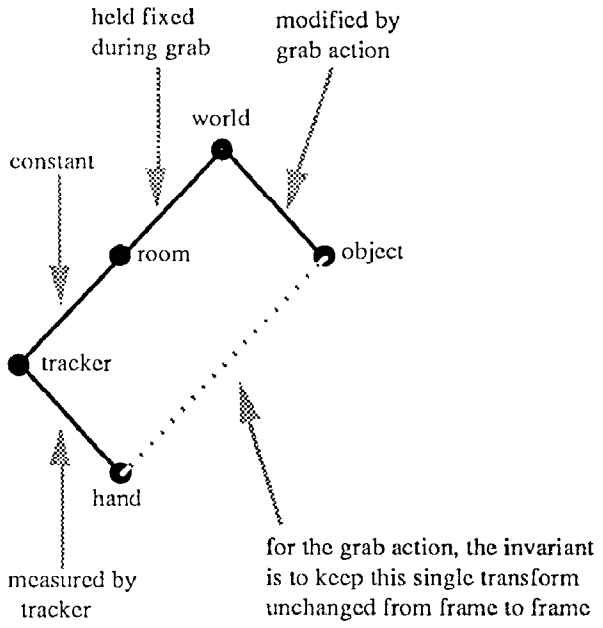

Figure 2. Coordinate systems diagram for grabbing an object

A way of describing the action of grabbing is that the Object-Hand transform T_{OH} remain unchanged from frame to frame, which is expressed by the invariant

$$T_{OH}' = T_{OH}$$

where the apostrophe in T_{OH}' indicates a transform in the current frame which is being updated, and no apostrophe means the value of the transform from the previous frame.

To move an individual object, the Object-World transform T_{OW} must be updated each frame in a way that preserves the invariant. To derive the update formula for grabbing, we start with the invariant and decompose the transforms on both sides based on the relationships among the coordinate systems as shown in the coordinate system diagram.

$$T_{OW}' \cdot T_{WR}' \cdot T_{RT}' \cdot T_{TH}' = T_{OW} \cdot T_{WR} \cdot T_{RT} \cdot T_{TH}$$

We then use algebraic manipulations to isolate the desired transform on the left side of the equation, remembering that these transforms are not commutative.

$$\begin{aligned} T_{OW}' \cdot T_{WR}' \cdot T_{RT}' &= T_{OW} \cdot T_{WR} \cdot T_{RT} \cdot T_{TH} \cdot T_{HT}' \\ T_{OW}' \cdot T_{WR}' &= T_{OW} \cdot T_{WR} \cdot T_{RT} \cdot T_{TH} \cdot T_{HT}' \cdot T_{TR}' \\ T_{OW}' &= T_{OW} \cdot T_{WR} \cdot T_{RT} \cdot T_{TH} \cdot T_{HT}' \cdot T_{TR}' \cdot T_{RW}' \end{aligned}$$

This is the update formula for grabbing, which updates the Object-World transform based on its previous value, the current and previous values of the Hand-Tracker transform (which changes as the hand moves), and the values of the intervening transforms between Tracker and World. The effect of executing this assignment each frame is to keep the object in a fixed position and orientation relative to the hand, even though the hand is moving around within the virtual world.

Another action which can be implemented in a similar manner is "grabbing the fabric of space." In this case, the user can grab and tilt the entire virtual world, rather than just a single object, by holding the World-Hand transform invariant while the hand rotates.

FLYING

The action of flying is translating the user through the virtual world in the direction pointed by the manual input device. The user steers by rotating the manual input device as the flight proceeds. A metaphor for this type of flying is that the user holds a rocket pistol in his hand, which drags him through the virtual world when he squeezes the trigger.

The manual input device is considered to point in a particular direction that is relative to its local coordinate system. This may be thought of as a 3D vector in Hand coordinates, where the vector's length specifies the flying speed and the vector's direction defines the direction the input device points. This vector defines a translation transform, $T_{HtranslateH}$, which moves a point in Hand coordinates to a new position in Hand coordinates. To implement flying, we first need to convert this transformation to operate on points in Room coordinates.

$$T_{RtranslateR}' = T_{RH}' \cdot T_{HtranslateH}' \cdot T_{HR}'$$

To make the user's position change within the virtual world, the World-Room transform must be modified each frame, so the invariant for flying is

$$T_{WR}' = T_{WR} \cdot T_{RtranslateR}'$$

which may be expanded to give the update formula for flying.

$$T_{WR}' = T_{WR} \cdot T_{RT}' \cdot T_{TH}' \cdot T_{HtranslateH} \cdot T_{HT}' \cdot T_{TR}'$$

SCALING THE WORLD

It is possible to shrink or expand the surrounding virtual world. This is comprehensible and effective because the user has direct

perception of the size of and distance to virtual objects through stereopsis and head-motion parallax, and can therefore easily perceive the concerted motions of the objects in the virtual world expanding around a center of expansion, or shrinking towards a center of contraction.

The type of scaling used is uniform scaling, in which all three dimensions are always scaled by the same factor. There is always a center of scaling when uniform scaling occurs, and for the manually controlled action of scaling the world, it makes sense to locate the center of scaling at the user's hand. When expanding the world, the center of scaling is the point that virtual objects move away from as expansion occurs, and so to end up at a specific desired location within a formerly-tiny virtual world, the center of scaling must be repeatedly re-centered on the desired location as it emerges during expansion.

Implementing this action requires a derivation similar to that used for flying. An incremental scaling transformation in Hand coordinates, $T_{HscaleH}$, will use the Hand origin as the center of scaling. Below we give the invariant for scaling the world, and the update formula derived from it.

$$T_{WR}' = T_{WR} \cdot T_{RscaleR}'$$
$$T_{WR}' = T_{WR} \cdot T_{RT}' \cdot T_{TH}' \cdot T_{HscaleH} \cdot T_{HT}' \cdot T_{TR}'$$

GENERAL FORM

Upon examining the invariants for flying and scaling, we see a strong similarity between them: both invariants are of the form:

$$T_{WR}' = T_{WR} \cdot T_{R<transform>R}'$$

In fact, these two invariants for updating T_{WR} are examples of a more general technique for updating a transform between two coordinate systems based on a transform that occurs in a third coordinate system. The general form for updating the transform T_{AB} in terms of an action in coordinate system K is:

$$T_{AB}' = T_{AB} \cdot T_{BK}' \cdot T_{K<transform>K} \cdot T_{KB}'$$

where there may be an arbitrary number of coordinate systems between B and K, and T_{BK} is the product of the transforms that go between the two coordinate systems.

Using this general form, scaling an object about the hand is analogous to scaling the world about the hand:

$$T_{OW}' = T_{OW} \cdot T_{WH}' \cdot T_{HscaleH} \cdot T_{HW}'$$

CONCLUSIONS

The foregoing examples of grabbing, flying and scaling show how actions can be implemented that operate under continuous manual control by the user. For each action, the relationship between the motion of the hand and the transforms to be modified was precisely specified with an invariant. These invariants not only provided a concise and precise specification of each action, but also provided a starting point for a formal derivation that produced update equations which could be used directly to implement the actions.

Using invariants and derivations to produce the code to implement grabbing, scaling and flying is greatly superior to the method which is often used, namely, to just write down a sequence of transforms that looks right based on the coordinate system diagram. It is easy to get some of the transforms in the wrong order. The notation used in this paper provides a check against misordering the transforms by requiring adjacent subscripts to match. The HMD software at UNC was implemented using this notation and the formulas derived in this paper, and serves as proof that they work.

ACKNOWLEDGEMENTS

We would like to thank many people for their contributions to this work, starting with the HMD and Pixel-Planes teams at UNC, led by Fred Brooks and Henry Fuchs. We thank Fred Brooks for useful discussions about coordinate systems diagrams and nomenclature. This work builds on earlier work at NASA, and we would like to acknowledge the contributions of Scott Fisher, Jim Humphries, Doug Kerr and Mike McGreevy. We thank Ken Shoemake for help with quaternions and Julius Smith for rules-of-thumb about writing technical papers. This research was supported by the following grants: DARPA #DAEA 18-90-C-0044, NSF Cooperative Agreement #ASC-8920219, and DARPA: "Science and Technology Center for Computer Graphics and Scientific Visualization", ONR #N00014-86-K-0680, and NIH #5-R24-RR-02170.

REFERENCES

[1] Blanchard, C., S. Burgess, Y. Harvill, J. Lanier, A. Lasko, M. Oberman, M. Teitel. Reality Built for Two: A Virtual Reality Tool. *Proc. 1990 Workshop on Interactive 3D Graphics.* 35-36.

[2] Clark, J. 1976. Designing surfaces in 3-D. *Communications of the ACM.* 19:8:454-460.

[3] Fisher, S., M. McGreevy, J. Humphries, and W. Robinett. 1986. Virtual Environment Display System. *Proc. 1986 Workshop on Interactive 3D Graphics.* 77-87.

[4] Foley, J., A. van Dam, S. Feiner, J. Hughes. 1990. *Computer Graphics: Principles and Practice* (2nd ed.). Addison-Wesley Publishing Co., Reading MA. 222-226.

[5] Holloway, R., H. Fuchs, W. Robinett. 1991. Virtual-Worlds Research at the University of North Carolina at Chapel Hill. *Proc. Computer Graphics '91.* London, England.

[6] Pique, M. 1980. Nested Dynamic Rotations for Computer Graphics. M.S. Thesis, University of North Carolina, Chapel Hill, NC.

[7] Robinett, W. 1990. Artificial Reality at UNC Chapel Hill. [videotape] *SIGGRAPH Video Review.*

[8] Sutherland, I. 1968. A head-mounted three-dimensional display. *1968 Fall Joint Computer Conference, AFIPS Conference Proceedings.* 33:757-764.

[9] Vickers, D. 1974. Sorceror's Apprentice: head mounted display and wand. Ph.D. dissertation, Dept. of Computer Science, Univ. of Utah, Salt Lake City.