Fluid VR: Extended Object Associations for Automatic Mode Switching in Virtual Reality

Mayra Donaji Barrera Machuca, Junwei Sun, Duc-Minh Pham, Wolfgang Stuerzlinger*

SIAT, Simon Fraser University

ABSTRACT

Constrained interaction and navigation methods for virtual reality reduce the complexity of the interaction. Yet, with previously presented solutions, users need to learn new interaction tools or remember different actions for changing between different interaction methods. In this paper, we propose Fluid VR, a new 3D user interface for interactive virtual environments that lets users seamlessly transition between navigation and selection. Based on the selected object's properties, Fluid VR applies specific constraints to the interaction or navigation associated with the object. This way users have a better control of their actions, without having to change tools or activate different modes of interaction.

Keywords: Interaction techniques. 3D selection. 3D navigation. 3D interfaces, Virtual reality.

Index Terms: H.5.2. Information interfaces and presentation (e.g., HCI): Interaction styles

1 INTRODUCTION

Thanks to recent technical advances, virtual environments are now accessible to a broader public. Regardless of the display device, users need to be able to perform three basic interactions: selection and manipulation, navigation, and system control [4] for a completely immersive experience. Yet, completely unconstrained interaction can frustrate users as they often end up with unexpected results [6]. On the other hand, most constrained interaction methods focus only on one of the three basic interaction modes. Examples include Bowman and Hodges [1] for manipulation, or Gleicher and Witkin [3] for navigation. However, if users want to change between interaction modes, they need to manually switch input modes/mappings/methods through buttons, gestures, or by using two hands instead of one. This need for a switch increases the potential for mode errors.

In this paper, we present Fluid VR, a new interaction method for 3D environments that maps user input differently depending on the properties of the object the user is interacting with. The idea of using object properties to constrain the user actions originates with Bukowski & Sequin [2], but they do not change between interaction modes. The basic idea of Fluid VR is to use the properties of each virtual object to activate specific user interface mappings and constraints that limit user actions. For example, if the user selects a glass, they can move it somewhere else, if they select a door, they open it, and if they select the frame of an open door, they will navigate through it. More precisely, Fluid VR integrates three different constraint-based interaction techniques: 1) a manipulation method, 2) a navigation method that relies on pre-programmed interactions, and 3) a flying method. After selecting an object, the system automatically identifies which of these three interaction methods needs to be activated, which affords users a substantially simpler user interface, while still providing the accuracy of constrained interaction. Our contributions are as follows:

- Fluid VR, a new efficient way to interact with 3D virtual environments by constraining the user action depending on the object category.
- A new variant of a constraint-based manipulation method, Shift-Sliding [5], adapted for use in head-mounted displays.
- A new navigation method that seamlessly transitions between flying and orbiting.
- A gesture-based method to efficiently activate preprogrammed navigation.

2 FLUID VR

The objective of Fluid VR is to simplify interaction with virtual 3D environments, while at the same time being easy to use and affording high accuracy.

Figure 1: Controller User Interface Mappings.

2.1 Basic Algorithm

A fluid interaction technique ideally keeps the input mapping natural while exploiting constraints on the actions. This can be achieved using the selected object's properties. Here we rely on three object properties: fixed objects, movable objects, and repetitive action objects. Fixed objects are static and can be used for navigation. Movable objects are dynamic, and users can move them around. Finally, repetitive action objects have a specific purpose and users can only use them in a certain way.

To start an interaction, the user first selects the object they want to interact with. To do this, we use the trigger button of the Vive controller to ray-cast from the head through the controller position to generate a virtual cursor. Using ray-cast also provides a large reach for interaction. Then the algorithm identifies the properties of the object and activates the corresponding input mapping and constraints. Based on these constraints, the user can activate specific interactions with the object, always through the same buttons and/or movements. Figure 1 shows the mapping for interaction.

* {mbarrera, junweis, ducminhp, w.s}@sfu.ca

2018 IEEE Conference on Virtual Reality and 3D User Interfaces
18-22 March, Reutlingen, Germany
978-1-5386-3365-6/18/$31.00 ©2018 IEEE

2.2 Movable Objects

We map the movement of movable objects so that the manipulated object moves along the surface behind it that it is currently in contact with, like Shift-Sliding [6]. We extended Shift-Sliding to work in a head-mounted display environment, where the user can move and rotate the controller freely by creating a virtual cursor. As in Shift-Sliding, we use an orthogonal camera that originates from the user's head for the computations in the sliding algorithm. While an object is selected, we temporarily turn physics off. This avoids the situation that the user could knock the stack over during dragging. When the user places the object through releasing the trigger button, we re-activate physics.

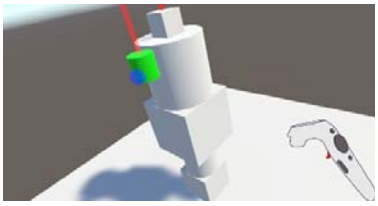

Figure 2: User moving the last object to the top of the stack.

To help with alignment during stacking, we show a red line through each object's center, see Figure 2. When an object's center is close enough to the centerline of the object it is in contact with, i.e., a base object, we snap the center of the top object to the bottom's centerline to align the objects. Using this method, we can stack the objects stably in different orders (within the limits imposed by Unity's physics engine).

2.3 Repetitive Action Objects

User gestures activate specific functions for repetitive action objects. Users can select multiple repetitive action objects at the same time and switch between them using the touchpad. After this the user only needs to do the gesture to activate the action when the specific object is active. After each flick, the system waits until the controller becomes stationary, before the next preprogrammed action can be initiated. This way we avoid false inputs. To release an object, users need to have it activated and then click the grip buttons to release them.

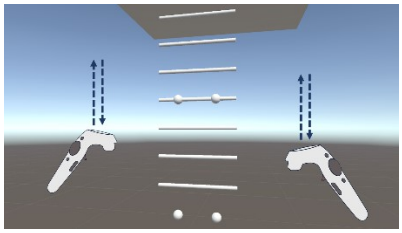

Figure 3: User control for the climbing task.

For the climbing task, we use both controllers to manipulate the movement of the virtual hands and feet. The left controller handles the virtual left hand and foot, and the right controller the right hand and food. The foot is moved if the touchpad is pressed and the hand if is not. To activate the preprogrammed up-movement function, the user makes an upward flick gesture, and correspondingly for downward motions. See Figure 3.

2.4 Fixed Objects

Users can navigate around fixed objects through two navigation methods, fly-to and orbiting. To switch between these methods, the user needs to press the grip button. After a fixed object is selected, the system automatically identifies if the object is a flat surface or a ring, using the object's geometry. Depending on the type of surface, the fly-to target position is where the ray hits the object (flat surface) or the center of the object (ring). If the orbiting mode is activated, the user makes a stroke with the controller to specify at which distance and which direction relative to the initially selected point they want to orbit. Based on the target distance to the object and the edges crossed by the stroke, the system calculates the orbiting (pivot) center and the distance to it. See Figure 4.

Figure 4: Example of fixed object orbiting navigation.

To navigate towards the target, users need to press the touchpad. The system then automatically flies the drone to the specified position by calculating the force needed to move the drone in each axis. Once they arrive at the target, the drone will stay in that position, which helps users reorient themselves, e.g., after they have passed through a target ring. If is an orbiting motion, the drone will automatically start rotating using the orbiting pivot and vector, until the user releases the touchpad.

3 Limitations

Currently, we manually tag each object to assign object properties. However, in the future we are planning to create an automatic method to achieve this.

4 Conclusion

Fluid VR simplifies user interactions in virtual environments by automatically changing between interaction and navigation modes. Then, users can focus less on interaction and more on their tasks. Moreover, as the three interactions methods behind Fluid VR constrain user actions, they reduce user frustration and increase accuracy and speed.

References

[1] D.A. Bowman and L.F. Hodges, 1997, April. An evaluation of techniques for grabbing and manipulating remote objects in immersive virtual environments. In Proceedings of the 1997 symposium on Interactive 3D graphics (pp. 35-38). ACM.

[2] R.W. Bukowski and C.H. Séquin, 1995, April. Object associations: a simple and practical approach to virtual 3D manipulation. In Proceedings of the 1995 symposium on Interactive 3D graphics (pp. 131-214). ACM.

[3] M. Gleicher and A. Witkin, 1992, July. Through-the-lens camera control. In ACM SIGGRAPH Computer Graphics (Vol. 26, No. 2, pp. 331-340). ACM.

[4] J. Jankowski and M. Hachet, 2013, May. A survey of interaction techniques for interactive 3D environments. In Eurographics 2013-STAR.

[5] J. Sun, W. Stuerzlinger and D. Shuralyov, 2016, October. Shift-Sliding and Depth-Pop for 3D Positioning. In Proceedings of the 2016 Symposium on Spatial User Interaction (pp. 69-78). ACM.

[6] W. Stuerzlinger and C.A. Wingrave, 2011. The value of constraints for 3D user interfaces. In Virtual Realities (pp. 203-223). Springer Vienna.