Design of Hand Gestures for Manipulating Objects in Virtual Reality

Wanhong Lin[1,2](✉), Lear Du[3], Carisa Harris-Adamson[3],
Alan Barr[3], and David Rempel[3]

[1] China Astronaut Research and Training Center, Beijing, 100094, China
acclwh@hotmail.com
[2] National Key Laboratory of Human Factors Engineering,
China Astronaut Research and Training Center, Beijing, 100094, China
[3] Department of Bioengineering, University of California, Berkeley, USA
lxd20@berkeley.edu,
{Carisa.Harris-Adamson,Alan.Barr,david.rempel}@ucsf.edu

Abstract. Virtual reality requires high levels of interaction with the user, a type of human computer interaction. Interactions that match the way humans usually interact with their surroundings should improve training effectiveness. A 3D hand gesture based interface allows users to control the position and orientation of 3D objects by simply moving their hands, thereby, creating a more naturalistic interaction process. The design of hand gestures should be evaluated to determine design features that are the most effective and comfortable for the user. The purpose of this study was to evaluate parameters for the design of 3D hand gestures for object manipulation in virtual reality to optimize productivity and usability. Twenty participants completed object manipulation tasks while wearing an Oculus Rift headset with a mounted Leap Motion depth sensor camera to capture hand gestures. Independent variables were distance from hand to object, hand posture threshold for grab and release, and grab locations on object. The dependent variables were time for task completion and subjective measures of control, fatigue, motion sickness, and preference. The preferred gesture design parameter was related to better control and reduced time to complete the tasks. In conclusion, this study identified important gesture design features that can be optimized to improve usability and throughput for an object manipulation task in Virtual Reality.

Keywords: 3D hand gestures · Human-computer interaction · Hand postures

1 Introduction

With the rapid development of computer technology, Augmented and Virtual Reality (VR) environments have become an integral part of professional training. Since training systems based on virtual reality technology demand more attention and interaction from the user, a type of human computer interaction (HCI) that adopts the way humans usually interact with their real surroundings may be an important component to an effective system. Future training systems will utilize new technologies that require input devices

M. Kurosu (Ed.): HCI 2017, Part I, LNCS 10271, pp. 584–592, 2017.
DOI: 10.1007/978-3-319-58071-5_44

to be easily accessible and instantly available. Such technologies pose several challenges to the GUI (Graphical User Interface) paradigm designed for desktop interaction, particularly with current input devices such as keyboard and mouse [1–4].

An effective system requires control with high degrees of freedom; thus, direct manipulation of a 3D object with a user's hands may provide an improved experience compared to using a mouse or keyboard [3–5]. With this approach, the user can control the position and orientation of a 3D object with their hands similar to how they manipulate objects in the real world. For training VR, this approach more closely approximates the tasks that users will ultimately be performing.

Although wearable gloves can track hand postures, a computer vision based gesture recognition system is a more natural alternative since it is part of the environment and operates remotely. A gesture based interface eliminates the need for pointing devices, saving equipment, time and effort. The purpose of this study was to evaluate various parameters for the design of 3D hand gestures for object manipulation in virtual reality to optimize productivity and usability. Building on prior research [2], we propose using a new technique for estimating the 3D gesture based on the 3D pose of a user's hand detected by a depth camera in real-time. In addition, because performing precision hand gestures in VR for long periods of time may be physically strenuous the study also evaluated the effect of design parameters on user comfort.

2 Gestures Lexicon and Action States Transition

In terms of hand pose, the user controls a virtual hand provided by the Leap Motion hand sensing device to manipulate a virtual object just as humans would do in the real world. There are numerous hand gesture poses that are recognized by the Leap Motion system. The virtual system captures, identifies and associates hand poses with specific gesture commands used to control virtual objects or menus.

One technique to implementing hand based interaction with virtual objects is to model real world physics using physical collision detection. In this study, the collision detection process is simplified by assuming that all objects are intangible. That is, the virtual hand does not manipulate the object from its surface; instead, the hand can pass through the object and attach to the object at various points on the inside or surface of the object. In terms of object manipulation, there are several kinds of actions that need to be carried out by virtual training systems leading to the commands generated by hand pose and gestures. These commands include Grab, Rotate, Move and Release.

Hand motions are continuous and different hand pose commands need to be distinguished by constantly tracking hand pose and movement in real time. This paper proposed two non-contact action determining rules and an action state transition method based on finite state machine (FSM). Finite state machine (FSM) [6, 7] is an effective tool for managing a series of hand action events. The interactive model of FSM for virtual hand action is shown in Fig. 1, where S represents the state, and C represents the states transition conditions.

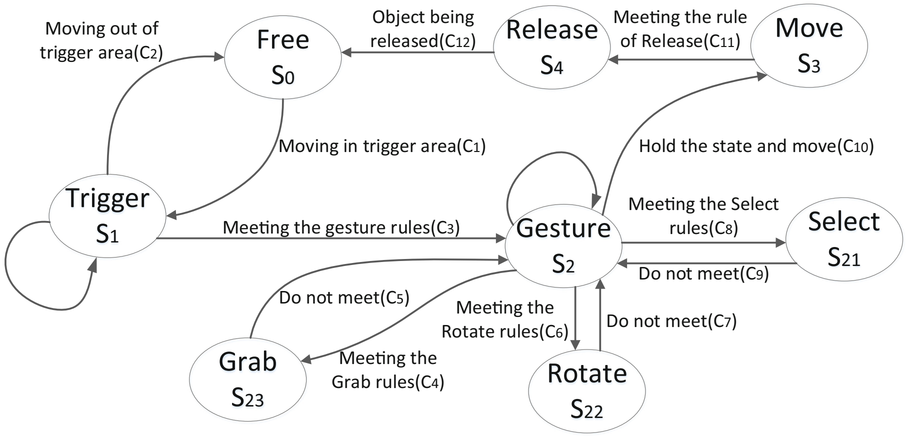

Free
S0
Release
S4
Move
S3
Trigger
S1
Gesture
S2
Select
S21
Grab
S23
Rotate
S22
Moving out of trigger area(C2)
Object being released(C12)
Meeting the rule of Release(C11)
Moving in trigger area(C1)
Hold the state and move(C10)
Meeting the gesture rules(C3)
Meeting the Select rules(C8)
Do not meet(C9)
Do not meet(C5)
Meeting the Rotate rules(C6)
Do not meet(C7)
Meeting the Grab rules(C4)

Fig. 1. The finite state machine of action

Rule One: The distance between the object and the virtual hand must be less than a certain critical value.

Rule Two: The feature value of the hand gesture must be less than a certain critical value, which is called the object action trigger threshold.

It was determined that an action was successful only if the virtual hand and the object met the condition of both rules at the same time. After a successful action (i.e. both rules simultaneously satisfied), the object would latch onto the virtual hand and move with it in virtual space. The object was released when the virtual hand did not satisfy rule two (i.e. the object action trigger critical value was exceeded).

According to the principle of FSM, the constituent elements of FSM are defined as: states (*S*), input events (*X*), output (*Y*), states transition function (*f*) and output function (*g*).

States (*S*): There are five action states of virtual hand: Free State, Trigger State, Gesture State, Move State and Release State. Each state has its own set of rules that govern how the virtual hand interacts with its surrounding environment.

Input events (*X*): These events correspond to the five states shown in figure one. There are six conditions which are the virtual hand moving into the object action trigger area, the virtual hand moving out of the object action trigger area, the virtual hand meeting the rules of gesture, the virtual hand meeting the rules of move, the virtual hand meeting the conditions of release.

Output (*Y*): This renders the results displayed to the operator.

States transition function (*f*): This function determines the state transition from the current state to the next one. Equation (1) shows its relationship with the states and time, where $X(t) \in X$,$S(t) \in S$.

$$S(t + 1) = f(X(t), S(t)) \tag{1}$$

Output function (g): This function maps the relationship between the current state and the output. Equation (2) shows its relationship with the states, where $Y(t) \in Y$.

$$Y(t) = g(X(t), S(t)) \tag{2}$$

As an example, consider the user transition from the trigger state to the gesture state. The state transition function f determined the state of the hand in the next time frame. If the input event at the current time frame, $X(t)$, met the rules of the grab gesture as defined by rule one and two, and the current state, $S(t)$, was the trigger state, S_1, then the next state $S(t + 1)$ would be the gesture state S_2. The output function g rendered the interface displayed to the user based on the input event and the current state. Again, if the user transitioned from the trigger state to the gesture state, the output $Y(t)$ would be an audio sound that let the user know that he or she successfully grabbed the object. Here the states are explained in more detail:

Free State (S_0): The virtual hand did not touch any object. In this condition, the virtual hand moved freely and the finger joints bend freely

Trigger State (S_1): The virtual hand moved into the object action trigger area, but did not meet the action determining rules

Gesture State (S_2): The virtual hand manipulated the object stably by following the gesture determining rules. The object latched onto the virtual hand in the gesture state

Move State (S_3): The virtual hand manipulated the object stably by following the two determining rules. The object rotated and translated with the virtual hand

Release State (S_4): The virtual hand transitioned to the release state when it did not satisfy Rule Two of the action determining rules after manipulating the object. In this state, the virtual hand released the object and then transitioned to the free state

3 Task and Experimental Setup

There are many devices that provide hand pose data such as Intel RealSense, Leap Motion, Kinect etc. Due to the accuracy of Leap Motion and its compatibility with the Oculus Rift [8], we chose the pose data provided by the Leap Motion for gesture recognition. The system set-up is shown in Fig. 2 and the objects that the user sees are indicated in the box.

A Leap Motion hand tracking device was mounted on an Oculus Rift VR headset using a custom mount that oriented the Leap Motion device to point 13° below a line perpendicular to the headset surface. The task involved manipulating a virtual hand to grab a virtual dice from a starting location, move the dice to a precise target location and orientation, and release it. The study was approved by the university Committee on Human Research.

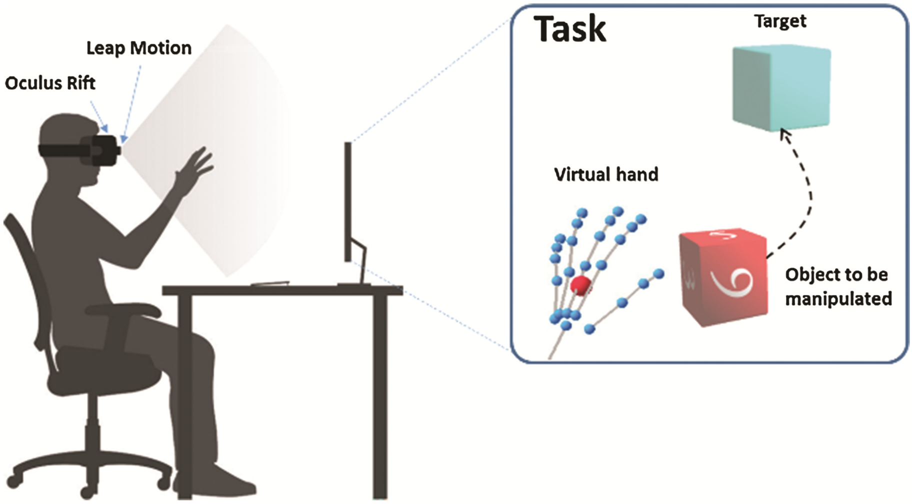


Fig. 2. The manipulation system

As shown in Fig. 3, the virtual skeletal hand model mimics the posture and movements of the user's real right hand based on hand detection by the Leap Motion sensor. Following the FSM model explained in Sect. 2, subjects manipulated the virtual hand to capture, move, rotate, and release the red dice (100×100×100 mm) until the red dice fit precisely inside the blue-green target box. The two requirements were:

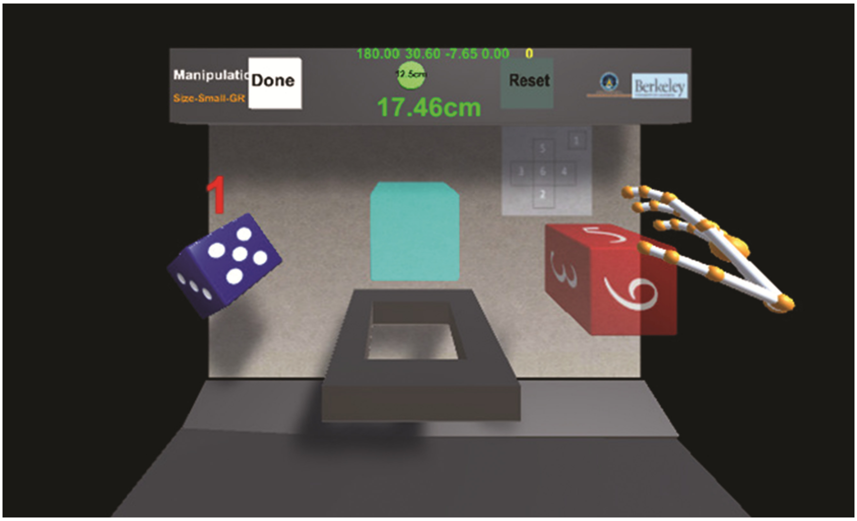


Fig. 3. The manipulation scene (Color figure online)

Position: The center of the red dice had to be within 2 mm of the center of the target box.

Orientation: The dice must be correctly orientated so that the number facing the subject matched the number on the smaller dark blue dice and was upright. The orientation of the red dice had to be within 3° of the target box orientation.

The subject then virtually pressed the DONE button on the upper left corner with their left hand to complete the task. The dice dropped down through the grey hole and appeared back at the original position. The task was repeated with the next number appearing on the front of the blue dice. The task was repeated 6 times (once for each number on the dice) for each parameter-level. The final scene is as Fig. 3.

The gesture commands for object manipulation included: push, grab, rotate, move, and release. The grab gesture modeled a pinch posture, in which the user brought the fingertips together. The threshold for the grab command (G_T) was based on the distance between fingertips and was the length of the subject's thumb metacarpal bone (m; extracted from the Leap Motion data) multiplied by a grab threshold constant, α.

$$G_T = \alpha \cdot m \tag{3}$$

Hence, the grab threshold value was different for each user and was based on their hand size. The release threshold (R_T) determined when the hand pose transitioned out of the grab state and was similar to the grab threshold, except $\beta \geq \alpha$.

$$R_T = \beta \cdot m \tag{4}$$

The release threshold could be set greater than the grab threshold to reduce the chance that the user might accidently transition out of the grab state and unintentionally release the dice.

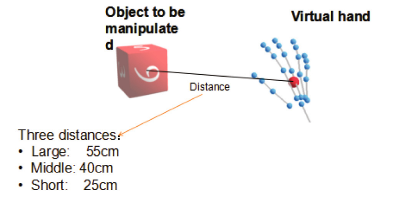
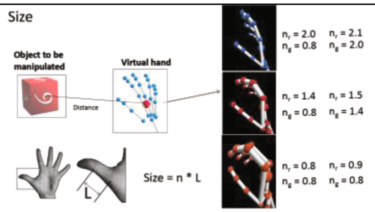
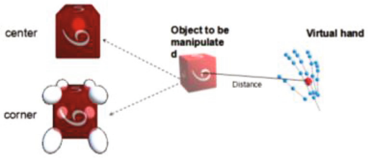
4 Experiments

Twenty subjects participated in the study; eight were male. The mean age was 22. Only two had familiarity with virtual reality devices.

Four different experiments evaluated 4 parameters on usability, performance and comfort. The parameters were grab distance (distance from palm to object before it can be captured), grab release difference (difference between grab threshold and release threshold: $R_T - G_T$), grab size (difference in α) and grab location (center of dice vs corners of dice). Each parameter was tested at 2 or 3 levels. e.g., Grab Distance was evaluated at 3 levels: short, middle, and large. The parameters and their tested levels are summarized in Table 1. For each parameter, the test order of levels was randomized. For each level, the task was repeated 6 times.

With the Grab Distance experiment, the dice could be grabbed and manipulated from different distances. When the palm pointed to an object and the object was within the proscribed grab distance: short (25–40 cm), middle (40–55 cm), large (55–70 cm), it would change color and could then be captured (grabbed) and moved and rotated. The dice rotated around the center of the palm, instead of around itself, so at the large distance, a small change in hand orientation would magnify the movement of the dice.

Table 1. Parameters tested in the 4 experiments

Parameter	Levels	Values	Concept Model
Grab Distance	Short	25 – 40	 Three distances: • Large: 55cm • Middle: 40cm • Short: 25cm
	Middle	40 – 55	
	Large	55 – 70	
Grab Release Difference	Small	0.8 Grab 0.8 Release	 Size Object to be manipulated Virtual hand Distance L Size = n * L $n_r = 2.0$ $n_g = 0.8$ $n_r = 2.1$ $n_g = 2.0$ $n_r = 1.4$ $n_g = 0.8$ $n_r = 1.5$ $n_g = 1.4$ $n_r = 0.8$ $n_g = 0.8$ $n_r = 0.9$ $n_g = 0.8$
	Middle	0.8 Grab 1.4 Release	
	Large	0.8 Grab 2.0 Release	
Grab Size	Small	0.8 Grab 0.9 Release	
	Middle	1.4 Grab 1.5 Release	
	Large	2.0 Grab 2.1 Release	
Grab Location	Center	Center	 center corner Object to be manipulated Virtual hand Distance
	Corner	Corner	

This is an application of Rule One from section three, in which the center of the palm had to be within a certain range from the center of the dice before it could be captured.

In the Grab Release Difference experiment, the distance for the grab threshold, α, was held constant at 0.8, but the distance threshold for the release, β, was tested at small (0.8), middle (1.4), and large (2.0). The 0.8 release threshold was the same as the grab threshold but the 2.0 release threshold was 2.5 times larger than the grab threshold of 0.8. In theory, a release threshold larger than the grab threshold might prevent the dice from being accidently released. This is an application of Rule Two, in which the spread between grab and release thresholds were different. For this experiment, applying Rule One, the palm center had to be within 12 cm of the dice center before the dice could be grabbed. Rotation was around the center of the dice not around the palm.

In the Grab Size experiment, the grab threshold, α, was varied between small (0.8), middle (1.4), and large (2.0) while difference with the release threshold was held constant ($\beta = \alpha + 0.1$). The purpose was to determine if grab size influenced usability or throughput. Again, for this experiment, the palm center had to be within 12 cm of the dice center before the dice could be grabbed and rotation was around the center of the dice.

The Grab Location experiment evaluated the difference between grabbing the dice at its center compared to grabbing the dice at any one of its 8 corners. The dice rotated around the center if the dice was grabbed at the center or around the corner if was grabbed at the corner. To capture the dice, the palm center had to be within 6 cm from any of the corners of the dice for the corner test, and within 12 cm from the center of the dice for the center test.

After each of the 4 experiments, subjects completed a survey evaluating subjective usability and comfort for each of the levels tested. The survey presented three statements and subjects rated each statement on a five-point Likert scale (1 was strongly disagree and 5 was strongly agree). The statements were (1) I had excellent control, (2) I had no shoulder fatigue, and (3) I feel motion sickness. Subjects also ranked the levels tested on overall preference, 1 being the most favorite and 3 being the least favorite.

Throughput for each level was calculated as the time to complete the 6 dice placement tasks.

Differences between subjective ratings were evaluated with the non-parametric Skillings-Mack test ($p < 0.05$) and differences in time to complete tasks were evaluated with repeated-measures ANOVA.

5 Result and Discussion

For each of the 4 experiments, differences between levels of a parameter had a significant effect on subjective usability, preference and time to complete tasks. For example, grabbing and manipulating an object at its center, as compared to its corners, improved the time to complete the task and the subjective usability rating of "control". Subjects reported greater rotation control with the center than the corner locations. The result is shown in Table 2.

Table 2. Grab location test results; mean (SD) subjective ratings, ranking, and time to complete task. (Likert scale: 1 = strongly disagree, 5 = strongly agree).

	Grab location		p-value
	Center	Corner	
Excellent control	4.3 (0.8)	3.2 (1.1)	0.007
No shoulder fatigue	3.7 (1.3)	3.6 (1.3)	0.65
Motion sickness	1.4 (0.7)	1.4 (0.7)	1.00
Preference (1 = favorite)	1.1 (0.3)	1.9 (0.3)	0.0008
Time (s)	14.5 (8.6)	34.2 (23.5)	< 0.0001

For the other parameters studied, there was significantly better performance and usability for the short grab distance (e.g., 25 to 40 cm), the middle grab release difference ($R_T - G_T = 0.6$) and the largest grab size ($G_T = 2.0$).

Subjects generally reported lower levels of shoulder fatigue for settings that were easier to control. However, significant difference in shoulder fatigue occurred with grab distance ($p = .046$). Participants reported little motion sickness during the experiments (mean rating 1.31 on 1–5 scale). The tasks required relatively little head rotation.

Overall, preference for a gesture design parameter was related to better control and reduced time to complete the tasks.

Some study limitations should be noted. The task was designed to be a relatively high precision task requiring placing the dice to within 2 mm and 3° of the target. In addition, the distance the dice was moved was relatively short. Other tasks will require higher or lower precision and larger movements of virtual objects. These differences in task demands may influence optimal gesture design. The gesture evaluated for object capture was based on distance between finger tips. Other gestures for object capture could be evaluated.

In conclusion, this study identified important gesture design features that can be optimized to improve usability and throughput for an object manipulation task in VR. Further refinement of this optimization may be useful. For example, the interaction between grab size and grab release difference should be explored. In addition, other hand gestures should be compared to the ones used in this study on usability and throughput. Overall, if properly designed and evaluated, 3D hand gestures have the potential to provide very functional human-computer interaction in VR.

Acknowledgments. This work was supported by the Ergonomics Lab at Berkeley, the Office Ergonomics Research Committee, and the China Astronaut Research and Training Center within the following research programs: Advanced Space Medico-Engineering Research Project of China (2013SY54A1303, 9140c770204150c77318, 060601).

References

1. Sen, F., Diaz, L., Horttana, T.: A novel gesture-based interface for a VR simulation: re-discovering Vrouw Maria. In: 2012 18th International Conference on Virtual Systems and Multimedia (VSMM), pp. 323–330 (2012). http://ieeexplore.ieee.org/xpl/mostRecentIssue.jsp?reload=true&punumber=6341172
2. Sato, Y., Saito, M., Koike, H.: Real-time input of 3D pose and gestures of a user's hand and its applications for HCI. In: Proceedings of the Virtual Reality 2001, pp. 79–86. IEEE (2001). http://ieeexplore.ieee.org/xpl/mostRecentIssue.jsp?punumber=7269
3. Lee, C.-S., Oh, K.-M., Park, C.-J.: Virtual environment interaction based on gesture recognition and hand cursor. Electron. Resour. (2008)
4. Dardas, N.H., Alhaj, M.: Hand gesture interaction with a 3D virtual environment. Res. Bull. Jordan ACM **2**(3), 186–193 (2011)
5. Ming, A., Yuqing, L., Bohe, Z., Fuchao, H.: Study on real-time interactive simulation of rotating top in weightlessness. Manned Spaceflight, March 2014. http://zw.yakjfw.org.cn:8081/kns55/loginid.aspx?uid=VlFpZ3Ivb2VqVno5&p=Navi/Bridge.aspx?LinkType=BaseLink&DBCode=cjfd&TableName=CJFDbaseinfo&Field=BaseID&Value=ZRHT
6. Chen, Y., Lin, F.: Safety control of discrete event systems using finite state machines with parameters. In: Proceedings of the American Control Conference, pp. 975–980. American Auotmatic Control Council, Arlington (2001)
7. Feng, Z., Yan, B., Xu, T., et al.: 3D direct human-computer interface paradigm based on free hand tracking. Chin. J. Comput. **37**(6), 1309–1323 (2014)
8. Weichert, F., Bachmann, D., Rudak, B., Fisseler, D.: Analysis of the accuracy and robustness of the leap motion controller. Sensors **13**(5), 6380–6393 (2013)