A VR Study on Freehand vs. Widgets for 3D Manipulation Tasks

Robin Schlünsen
6schluen@informatik.uni-hamburg.de
Universität Hamburg
Germany

Oscar Ariza
ariza@informatik.uni-hamburg.de
Universität Hamburg
Germany

Frank Steinicke
steinicke@informatik.uni-hamburg.de
Universität Hamburg
Germany

ABSTRACT

We present in this article the results of a study based on a 3D user interface toolkit we developed which can be easily adapted and reused in VR applications featuring hand-tracking. We evaluated hand-based and widget-based manipulation techniques as well as different multimodal cues for 3D manipulation and system-control tasks. Our study compared the techniques in terms of performance and user acceptance. We found that free-hand manipulation is faster and preferred by the participants. We also analyzed the influence of the multimodal cues, finding valuable insights to integrate these cues and improve the user experience in 3D manipulation tasks.

CCS CONCEPTS

• **Human-centered computing** → **Virtual reality**; **Gestural input**; **User interface toolkits**; *User studies.*

KEYWORDS

freehand manipulation, 3D interaction, widgets, VR, multimodal cues

ACM Reference Format:
Robin Schlünsen, Oscar Ariza, and Frank Steinicke. 2019. A VR Study on Freehand vs. Widgets for 3D Manipulation Tasks. In *Mensch und Computer 2019 (MuC '19), September 8–11, 2019, Hamburg, Germany*. ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3340764.3340791

MuC '19, September 8–11, 2019, Hamburg, Germany

ACM ISBN 978-1-4503-7198-8/19/09…$15.00
https://doi.org/10.1145/3340764.3340791

1 INTRODUCTION

In traditional 2D desktop applications, user interfaces (UIs) have evolved over some time and became standard. Nowadays these interfaces do not develop much further, because they work very good for most purposes and computer users are already very familiar with the standard components and techniques. They can be used intuitively in most cases and are easy to learn for new users. In contrast to that, in 3D user interfaces we have no standards and such well-known user interfaces and interaction techniques do not exist. There has been a lot of research on 3D user interfaces, but a well-known set of standards has yet to develop. Often, 3D user interfaces are more complex because they have more degrees of freedom (DOF). However, this has also a lot of advantages, because it is possible to develop much more powerful interfaces. Also, they can be more natural then desktop user interfaces, that are controlled with input devices like keyboard and mouse. In 3D, it is for example possible to use grasping to select and manipulate objects, which is a very natural way of interaction that we know from the real world. In addition, it is possible to develop completely new user interfaces that would not be possible in the real world, which can feel like magic for the user, because the developers are not bound to physical restrictions in 3D applications. In general, developers of 3D user interfaces have to think about the advantages and disadvantages of different methods very well. In most cases, there is a trade-off between two different alternatives and, since there are no standards, and also a lot of different approaches, user interfaces are specialized for a certain application only.

In this article we present a reusable 3D user interface toolkit with different interaction techniques, that can be used in a variety of 3D applications and is easy to integrate in VR environments. We start with our selection technique that is based on ray-casting and hand-tracking. It has some additional features supported on a set of 3D widgets that we developed. One widget we look into is the hand menu, used to trigger different actions and we use it as the central menu of our toolkit. Furthermore, we present our multimodal cues (e.g., introduced as helpers to the participants) and take a closer look at the benefits they offer. We conducted a study

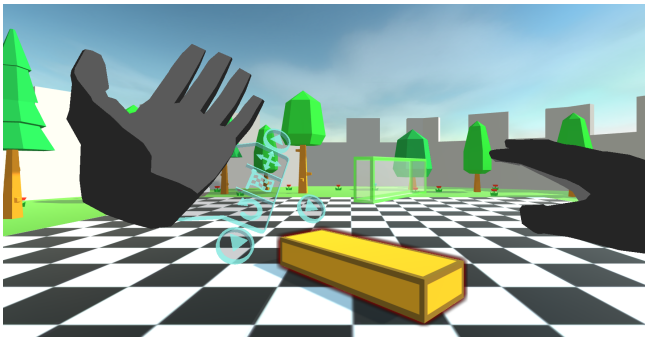

Figure 1: The experiment task: Move and rotate a yellow building block to match a semi-transparent (green) building block. The hand-menu was used to switch between manipulation widgets and tools.

(see section 4) to compare two different techniques for moving and rotating objects in VR applications. The first one uses 3D widgets and the second one is a free hand manipulation (FHM) technique (e.g., similar to a grasping technique). We also analyzed the influence of the cues, provided as combinations of visual and auditory feedback, and compared the different techniques in terms of performance, and usability. At the end (section 6) we sum up our results and draw a conclusion providing useful insights for 3D user interfaces involving hand-tracking, FHM, and widgets.

2 RELATED WORK

This section shows related research that was an inspiration for developing the 3D user interface toolkit. Our goal was to make a variety of interaction techniques available bundled in our toolkit. We start by looking on what impact the posture of the user has. After that we talk about different hand-based and vector-based selection techniques. Afterwards we take a look at different manipulation techniques. We deal with indirect manipulation like widgets, two-handed manipulation and gestures.

It is important to consider the posture a user has to take for using 3D applications, as he or she often stands the whole time. In contrast to that, desktop application users are sitting in most cases. The position of the arms and hands is also important. If the user must hold his hands far away or in a high position over time, the shoulder and arms get tired fast. This is also referred to as Gorilla arm syndrome by many researchers. This can be reduced strongly by lowering the arms and moving them closer to the body as [7] stated.

Regarding 3D object selection in VR, hand-based approaches use a trigger to select objects at the position of the virtual hand. This trigger can, for example, be a button on a tracked controller. Finger-based approaches often use gestures or certain finger configuration to trigger the grasp. This is possible through devices that allow full hand tracking like data-gloves or visual finger tracking (e.g. Leap Motion). An important aspect of finger-based approaches is how the fingers are displayed. They can either be allowed to pass through objects to always match the real fingers of the user or they can stop when colliding with objects. Some different approaches are Rigid-Body Fingers [3], Soft-Body Fingers [13] and God Fingers [28]. The simplest implementation of grasping methods maps the input of the user one-to-one onto the virtual hand. This is a very natural way of interacting. The big issue of this grasping technique is that the user can only select objects in his area of reach. Even if the input of the user would be scaled by a fixed factor the area of reach would still be finite. The Go-Go technique [23] tries to solve this issue by using a nonlinear transfer function. The idea is to scale the motion of the user real hand if it reaches a certain threshold. With that technique, the user can interact precisely with objects that are close to him and can interact with objects that are further away than his natural area of reach (but with less precision). Also, several studies ([4], [24]) showed that users had no problem to understand the a technique with nonlinear transfer functions.

Vector-Based techniques where developed a long time ago. One of the earliest examples of this category is the put-that-there interface [2], where the user points on an object and triggers the selection via speech commands. The ray-casting approach is limited in precision to the angular precision of the tracking sensor and little changes in the angle can make a big difference further away. This was for example observed by [9] [18] and a study from [24]. Another interesting approach by Noah Zucker [32] sends the ray from the users head through a cursor inside of a circle, that was formed by the user's thumb and index finger. Another approach is to use selection volumes to select multiple objects with one trigger action. A more advanced method is to define selection volumes dynamically. This would be the counterpart of the rectangle selection with the mouse in file explorers on desktop computers. [29] developed such a technique of defining a 3D cubic volume dynamically for selection purposes, using a bi-manual technique to define the cubic volume.

Selection often has the problem of inaccurate tracking and jittery. Furthermore, there can be many small objects close together and far away from the user. This makes selection a very hard task. Progressive refinement can be used to overcome this problem by adding multiple selection levels. The user selects a subset of all objects first and can then choose one of these objects to select it (with a selection technique that requires less accuracy). For example, SQUAD [15], uses a QUAD-menu for the refinement, Expand [6], is an extension of the SQUAD approach, but with better performance results, because in every case only two clicks are needed for any selection, and Double Bubble [1], which fixes a problem from

Expand, that there might be too many selection candidates after the first step.

Direct manipulation can cause occlusion issues like the fat finger problem [27]. Indirect manipulation solves this problem because the user does not directly touch the object he manipulates. Another advantage of indirect manipulations is that the developer can decide how much freedom the user should have in the manipulation. Often it is useful to reduce the complexity, by for example, using less DOFs leading to better performance (e.g., precision). As an alternative, widget techniques are a powerful way of interacting with virtual objects. Instead of manipulating a proxy of the selected object the user uses specialized and solely for that purpose designed widgets. For example, many 3D desktop applications have specialized widgets for moving, rotating and scaling objects. The user can simply switch between different widgets for different purposes. One disadvantage of widgets is that it adds additional objects to the scene, that could occlude or disturb the user in any way. An interesting widget technique for 3D rotations is the Virtual Sphere technique. There are several similar approaches by [8], [10], [26], and [12]. The idea of this technique is to reduce the complexity from three to two DOFs. This is achieved by drawing a virtual sphere around the selected object, the sphere can then be rotated by dragging it, which then rotates like a trackball and the object rotates together with the sphere.

So far, most of the techniques described before required only one hand for the interaction. This is useful in cases where only limited trackers or other hardware is available. Since tracking hardware for VR applications, becomes cheaper and is produced for the mass market, it is usually no problem anymore to track two hands. This opens new opportunities for two-handed or bi-manual interaction techniques [30]. In the same way, hybrid interaction techniques [17] enable technique integration as in HOMER [4]. It combines different techniques for selection and manipulation. The user selects an object via ray-casting and then the interaction technique switches to manipulation mode. The user's hand moves towards the selected object and he can manipulate it like with grasping techniques. The manipulation is scaled depending on the distances of the user to the virtual object. There are also approaches that dynamically combine well-known selection and manipulation techniques so the user can grab objects directly with his hands or can use a ray for objects that are further away. He can use both hands for that task, so he can grab two objects at the same time or grab one object with two hands.

Finally, gestures can also be used for system control in 3D applications. The input devices for gestural input become more powerful, which leads to better tracking and gesture recognition. The big advantage of gestural input is that it can be performed without having to hold a controller in your hand. This can increase the feeling of presence since it feels more natural than holding a controller and perform all actions with a controller. The XBox Kinect can be used for full-body tracking and the Leap Motion can be used for precise hand tracking in VR/AR applications. An example application that only uses gestural input is Polyshop [20]; navigation and manipulation cannot really be separated because the interaction mode changes on-the-fly.

3 TOOLKIT

In this section we describe the main components of the toolkit we developed to create our VR application and conduct the experiment. We made the toolkit publicly available for other researchers and developers under the GPLv3 License [1]. This was the main goal of developing the toolkit. We want to make the implemented adaptable interaction techniques available for a large amount of researchers and want to encourage other researchers to use and adapt the toolkit for further research.

We start this section by explaining two important gestures that are used for multiple purposes in the toolkit. After that the selection technique that we implemented. It is based on ray-casting approaches and was extended to support multiple-object-selection. After that we introduce our implemented 3D widgets. A core part of the widget is the hand menu. It is used to switch between the different interaction techniques. Afterwards we explain the manipulation widgets and the free hand manipulation technique.

Gestures

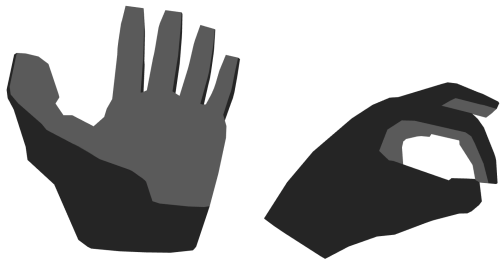

Figure 2: Gestures used in our toolkit: Palm gesture (left), and pinch gesture (right)

Gestures can be complicated and sometimes need some training for the user to perform them consistently. We tried to keep the gestures as easy as possible and tried to use as less different gestures as possible. Since we only use two simple gesture (on each hand) for our application, the user does not have to remember a large number of gestures. The first gesture is called the palm gesture, the user must open

[1] https://github.com/Hexoka/3DUI-Toolkit

and turn the hand so the palm is clearly visible in front. This is used for opening the hand menu and to start and end the selection mode (see figure 2). The second gesture we used for many 3D widgets and other actions is called the pinch gesture. It is triggered when the user presses his index finger and thumb together. It is triggered at a certain distance between the index fingertip and the thumb tip. This gesture is used to trigger different action like selecting objects and to drag and manipulate 3D widgets, they were easy to understand and learn, can be performed and tracked consistently, and do not lead to fatigue. Gestures recognition can often lead to false positives, to prevent that, we developed an algorithm that enables or disables the gesture recognition, based on the confidence of the tracking data.

Selection Technique

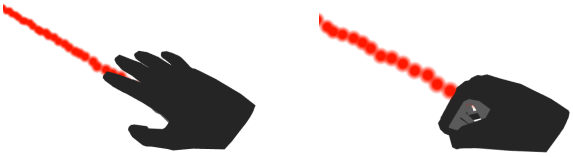

Figure 3: Selection ray-casting is attached to the user's palm or shifted to the side of the index finger if the user is pointing.

We implemented a selection technique that can be adapted based on the developers needs. It is based on ray-casting. The selection mode can be enabled and disabled by turning the palm of the right hand to the user's eyes (see Figure 2). After the user enabled the selection mode by executing this gesture, the ray becomes visible. The selection mode can be changed in a way that the ray is not based on the hand of the user but instead the selection is gaze-based. For this selection mode, the base of the ray is at the chest, instead of directly at the eyes, to prevent occlusion problems.

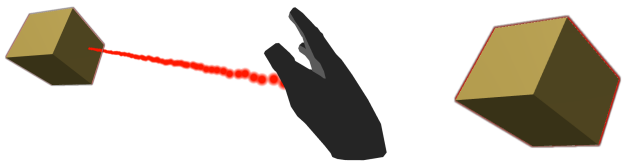

Figure 4: Selection of an object. The object gets a outline to show that it can be selected (left). After a pinch gesture the object is selected and the outline changes the color (right).

To select objects with the ray the user has to point on an object with the ray. This is then highlighted with an outline (see Figure 4). If the user then wants to select the highlighted object, must perform a pinch gesture between the user's thumb and index finger (see Figure 2). The gesture can be performed on both hands to trigger the selection. That has the advantage that the user can focus on the more fine motion of directing the ray with the right hand and can use his left hand for triggering the selection. Doing both with the right hand is also possible, but we found that trying to keep the ray on the same position while performing the gesture is sometimes difficult. This is especially an issue if the target object is small or far away. When the user selected an object the outline of the object becomes red and keeps visible as long as the object is selected. If the user performs the same action on an already selected object, the selected object gets deselected. We also support multiple-object-selection (MOS) with a Sphere-Volume-Selection method, holding the pinch gesture to change the sphere volume (see figure 5).

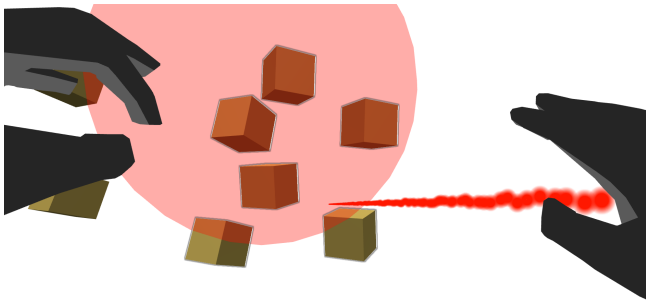

Figure 5: Multiple-object-selection (MOS) with our Sphere-Volume-Selection technique

3D Widgets

We developed a range of different 3D UI widgets. Most of the developed widgets are anchored to the head of the user. The only exception is the hand menu that is attached to the left hand of the user.

But instead of just attaching the widgets to the users head we developed a little more advanced type of head anchoring. We positioned the widgets in front of the user's body, but deeper than directly in front of the face. This has the advantage that it does not occlude the center of the view and that the widgets are easier to reach with the hands. The user is not forced to extend his arms over a longer time, because that would lead to fast fatigue [7]. The user is able to support his arms with his hips and interact with the widgets with only moving his forearms and hands. If the user rotates his head the widgets only follow this rotation on the yaw axis (the rotation when the user looks to the left or the right). This has the benefit that the user can, for example, look up without the widgets occluding the view. The widgets behave more like a control desk around the user. Furthermore, the widgets do not follow the head rotation strictly. A scaling

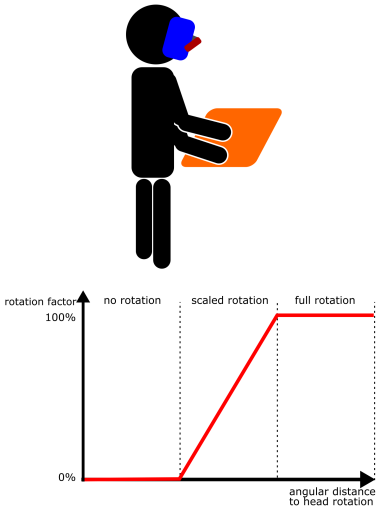

Figure 6: User posture when interacting with widgets (left). The factor of the widgets rotation depends on the angular distance to the head rotation of the user (right).

factor is multiplied with the heads rotation before it is applied to the rotation of the head-anchored widgets. Figure 6 shows how much the widgets follow the head rotation of the user depending on the angular distance to the head rotation of the user. There is a small area, where the widgets do not move when the head is rotating. We implemented this because when the user, for example, wants to interact with a button on the right side of his view, he will usually look at that button by rotating his head while moving his hand to that position. If the widgets would follow this rotation strictly, the user can get confused and can miss the button. Furthermore, it would be impossible to center this part of the widget in the user's view. The factor, that slows down the widgets, increases linearly depending on how large the angular distance to the head rotation is, and at a certain angular distance, the widget will follow the head rotation strictly.

Hand Menu

The hand menu is a simple menu with a one-level-hierarchy. Our main inspiration for the menu was a video of a similar menu attached to the user's hand in an AR application from Keiichi Matsuda [21]. Another example of a menu attached to the user's hand was proposed by [22]. The user can open the hand menu with the palm gesture (see Section 3). After performing the gesture an animation is shown while the hand menu is opened. The hand menu appears from behind the hand and rotates around the hand until it reaches the final position next to the hand. The menu is attached to the user's left hand and located next to the pinky finger. It is operated with the index finger of the user's right hand. So, the hand menu uses an asymmetric synchronous bi-manual interaction technique. The user always sees three menu items at a time and can scroll through the menu items with dedicated buttons above and below the hand menu (see Figure 7 label 1). Also, it is not possible to press two buttons at the same time. This prevents that the user presses multiple buttons when he accidentally hit the edge between two buttons.

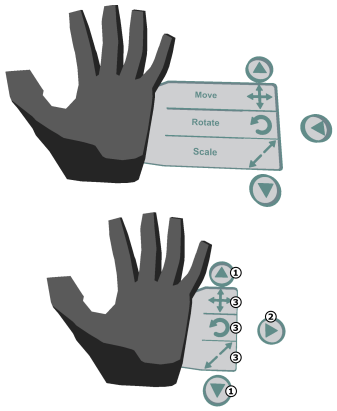

Figure 7: Hand-menu in beginner mode shows a label next to the buttons (left). Hand-menu is expert mode is more compact and does not show labels next to the button (1) Scroll buttons (2) Switch mode button (3) Menu item buttons.

Manipulation Widgets

An important task in many 3D applications is to change the position, orientation, and scale of objects. Houde ([11]) was one of the first who used widgets for manipulating objects in 3D applications. We developed a set of 3D widgets to provide this functionalities, adjusting values as the user operates the widget handles along the world axes.

Figure 8: 3D Widgets used to move, rotate and scale selected objects.

The widgets are inspired by traditional widgets in desktop 3D applications like 3D modeling tools, using handles along the three axes (two handles per axis, one in each direction). The handles are attached to a sphere in the middle (see figure 8). Every widget is located in front of the user in a comfortable position. The orientation of the widget is fixed to match

the world axes at all time. The user can interact with the widgets by performing the pinch gesture (see Section 3) within one of the handles. The widget is attached to the hand of the user until he releases the pinch gesture. The user can drag the widget along the selected axis. This movement, rotation or scaling is then applied to objects in the VE. As soon as the user stops the pinch gesture, the move widget will move back to its initial position, while the selected objects keep their new position. The user can then again drag an axis to move the objects further. This gives the user the ability to move objects far away without having to travel or making very large movements (e.g., as in a clutching technique). In addition to that, the widget provides auditory feedback. It uses the drag-clicker feedback (see Section 3). Furthermore, the move widget has a functionality to scale the translation in a non-linear way. The developer can choose between one of three different transfer functions, which are: Linear, quadratic and exponential. He can also adjust the transfer function with a constant.

Free Hand Manipulation

We also developed an interaction technique to perform manipulation tasks, but without widgets. Instead, we are using a free hand manipulation (FHM) technique. The selected object performs the same transformation as the hand of the user. It is similar to grasping techniques, but the rotation is not performed around the hand position. Instead, it is performed around the objects local coordinate system. When the user moves his hand, the selected object will be moved in the same direction. The amount is scaled by a transfer function. This gives the user the ability to perform translations in a much larger range. This works like the move widgets transfer function and the Go-Go technique [23]. The rotation works similarly. When the user rotates his hand, the object performs the same rotation, but in his own local coordinate system. This means the object will only change its rotation, but will not be translated, because the point of rotation is the origin of the object. The rotation is not scaled. This is not necessary, because the rotation is not limited as much as the translation. The user can use a pinch gesture with one hand to toggle if the object is moving or fixed. This gives the user the ability to do clutching and keeping the object fixed while re-positioning his hand for further manipulations.

Multimodal cues

We provide some visual and auditory feedback (cues) to increase the usability and make the techniques easy to learn.

Hand-proximity coloring. The toolkit provides a color cue on the fingers of the displayed hands based on the proximity to UI elements (e.g., when hovering an interactive component of a widget). It seamlessly change the strength of the coloring between zero (not colored) and one (fully colored) for each individual finger. An example of the colored fingers is depicted in figure 9.

Figure 9: The fingers of the user's hand can be colored individually using our finger color shader. The index finger is 100% colored and the ring finger is 50% colored.

Auditory Feedback. The auditory feedback has the advantage that the user can hear it even if he is looking somewhere else. Visual feedback stimuli can be missed easier because the view of the user is limited. This can be caused by either looking in another direction or occlusion from other objects. The auditory stimuli can be perceived even if its source cannot be seen. We used different types of auditory feedback. The first one is a confirmation for a successful action. The user will hear a click sound after a performed action (e.g., after pressing a button). The second type we called the *Drag-Clicker* feedback [16]. We used this for all widgets that have a dragging component (e.g., manipulation widgets). While dragging the appropriate part of the widget the user will hear a very short click sound after a certain small distance. When the user drags the widget, he will usually hear a lot of small clicks. From the number of clicks in a certain time, the user can estimate the velocity, with which he is currently dragging. Furthermore, the velocity determines the volume of the click. Furthermore, we have auditory feedback every time the user opens or closes a widget. This shows the user that the widget was opened or closed successfully. The last auditory feedback we provide is a sound every time a currently hovered button is pressed.

Hovering. We added different feedback to make the interaction with the hand menu as easy as possible. The first one is a hover functionality. If the index finger is within a certain distance of a menu button a hover value between zero and one is calculated. This value depends on the distance of the index finger to the menu button. After all hover values are calculated the button with the highest hover value is highlighted. The closer the finger is to the button (the higher the hover value is), the stronger is the visual feedback of

the hovering. The visual feedback is provided by coloring the background of the hovered button. This visual feedback helps the user to decide, which button he is going to press and helps to aim with the finger at the target button. Also, the user knows how far away he is from pressing the button, because of the dynamic strength of the hovering. Furthermore, we added animations for pressing a button. This gives a visual feedback that the button press was successful to the user.

Utility UI

We also developed a set of additional widgets for system control and symbolic input. Our toolkit offers a clock widget that can be used to input a time of the day with instant changes on the sky mapping of the VR environment; changing lighting conditions, sun or moon position as well as cloud visualizations. We created a color picker widget, it uses the HSV (hue-saturation-value) color model and is used to change the appearance of the selected objects. A 3D keyboard was implemented to input choices and answers during the experiment. We also provide traditional UI controls like sliders, push buttons, and toggle buttons, which were mainly used to change visual appearance settings.

4 EXPERIMENT

We conducted an experiment to compare two different interaction techniques for object manipulation. In the same way, we used different multimodal cues (e.g., combinations of visual and auditory feedback introduced as helpers to the participants) to make the interaction as easy as possible. We had conditions with the cues enabled and disabled in order to check how cues and techniques influenced the participants in terms of performance, accuracy and subjective preference. The multimodal cues were:

(1) Hover and button press effects in the hand menu
(2) Blurring of the environment when hand menu is open
(3) Hover effect on the widgets
(4) Coloring of the finger in the FHM technique
(5) Confirmation auditory feedback in the hand menu
(6) Hover change auditory feedback in the hand menu
(7) Drag-clicker auditory feedback in widgets

Participants

We had 12 participants, which are students in the fields of human-computer interaction or computer science (2 female, ages 20-29, M=23.1, SD=2.75). All of them had normal or corrected vision. None of our participants reported a disorder of equilibrium or a motor disorder such as an impaired hand-eye coordination or any other vision disorders. 10 participants were right-handed and 2 left-handed. 10 participants have participated in a study with an HMD before and 5 have participated in a study with a Leap Motion before. The experiment lasted about 60 minutes including instructions and questionnaires. The participants wore the HMD for about 45 minutes. They had a small break after each condition to answer the questionnaires.

Materials

The participants wore an HTC Vive Pro with an attached Leap Motion devices, that was tilted downwards by 45 degrees. The computer we used for the experiment has the following specifications:

- CPU: Intel Core i7-4930K, 3.40 GHz
- GPU: Nvidia GeForce GTX 1080 Ti
- Main Memory: 16 GB RAM

Methods

The participants were located inside a VE with some trees and flowers that was surrounded by a wall. The task was to move and rotate a building block to match another semi-transparent building block as good as possible (see figure 1). The participants were standing while performing the tasks. We measured the time that the user used the move and the rotate tool to perform the task as well as the Euclidean distance and angular distance to the goal configuration. The participants were told to work as fast and precise as possible. They decided when to finish the current task and they could open the movement or rotation widgets using the hand menu. The experiment had a 2x2 within-subject factorial design. To avoid unintended effects caused by the order, we used a Latin square to determine the conditions sequence for each participant. The participants also had a training phase at the beginning of each condition to reduce learning effects.

In total we had the following four conditions:

(1) FHM with Cues
(2) Widgets with Cues
(3) FHM without Cues
(4) Widgets without Cues

Before the experiment, all participants filled out an informed consent form and received instructions on how to perform the task. The participant had to answer a simulator sickness questionnaire (SSQ) [14] before the experiment and after the experiment. Furthermore, the participants answered a questionnaire to measure the system usability scale (SUS) [5] [25] after each condition. At the end, the participants had to answer the Slater-Usoh-Steed presence (SUSP) questionnaire [31]. In addition, we asked the following questions:

(1) With which technique you were faster?
(2) With which technique you were more precise?

(3) With which technique would you perform the task again?
(4) Do you think the cues improved your performance?
(5) Do you think the cues improved your accuracy?

The times was measured by how long the participant used the move or rotate tool in total to finish one task. For the Euclidean error distance we measured the distance from the center of the moving block to the center of the semi-transparent block in the goal position. For the angular error we summed up the three angular differences around the coordinate axis in comparison to the goal configuration.

Results

We analyzed the results with a repeated measures ANOVA at the 5% significance level. Degrees of freedom were corrected using Greenhouse-Geisser estimates of sphericity when Mauchly's test indicated that the assumption had been violated.

Movement Time. The movement time is the summed up time the participant was using the movement widget. Figure 10 shows the movement time for each condition. Table 1 shows the results of the statistical analysis. Technique (FHM and Widgets) presented a significant effect on the movement time ($F=70.923$, $p<0.001$).

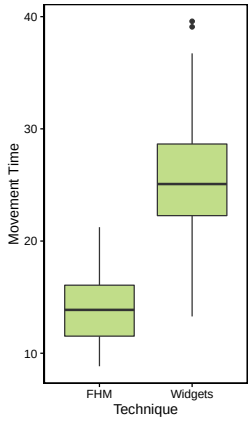

Figure 10: Plot of the movement time

Technique	Mean	SD
FHM	13.878s	6.247s
Widgets	25.636s	12.458s

Table 1: Results for movement time

Euclidean Error Distance. The euclidean error distance is the mean distance from the center of the movable block to the center of the target block when the participant finished the task. Figure 11 shows the mean error distance (Euclidean distance) for each condition. The Cues ($F=17.643$, $p<0.01$) and also the Technique ($F=12.382$, $p<0.01$) presented a significant effect on the euclidean error distance. Table 2 shows the results of the Post-hoc analysis.

Technique	Cues	M	SD
FHM	Yes	0.135	0.097
Widgets	Yes	0.190	0.148
FHM	No	0.086	0.052
Widgets	No	0.122	0.083

Post-hoc	p
FHM with Cues - FHM without Cues	0.019
Widgets with Cues - Widgets without Cues	0.019
FHM with Cues - Widgets with Cues	0.053
FHM without Cues - Widgets without Cues	0.021

Table 2: Statistical analysis for the error distance

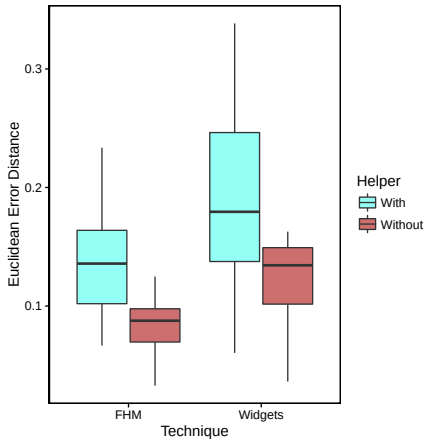

Figure 11: Plot of the euclidean error distance

Rotation Time. The rotation time is the summed up time when the participant was using the rotation widget. We found no significant differences for this variable.

Angular Error Distance. The angular error distance is the mean angular distance from the three rotation axes of the movable block to the three rotation axes of the target block when the participant finished the task. We did not found any significant differences.

Questionnaires

The participants filled in a System Usability Scale Questionnaire (SUS) after each condition, to rate their experience with the last technique to move and rotate the building block. The mean SUS scores and standard deviation for each condition can be seen in table 3. A plot of the results can be seen in figure 12.

Condition	M	SD
Free Hand With Cues	71.667	16.866
Widgets With Cues	58.125	22.465
Free Hand Without Cues	68.125	18.063
Widgets Without Cues	51.667	23.460

Table 3: Statistical analysis for the SUS scores

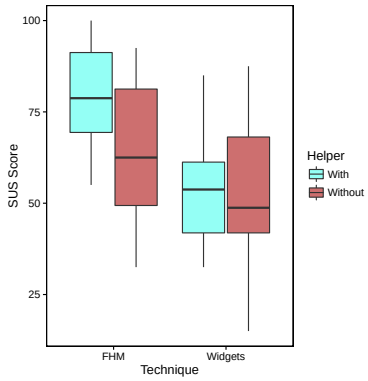

Figure 12: Plot of the SUS scores

An overview of the answers from the subjective questions can be seen in table 4.

We used a Simulator Sickness Questionnaire SSQ and measured a mean score of 8.73 (SD = 9.74) before and 19.64 (SD = 21.93) after the experiment. Presenting a significant difference (p-value<0.05) similar to related VR experiments (e.g., as reported by Lubos et. al.[19]). For the presence, we used a Slater-Usoh-Steed Questionnaire (SUSP) and measured a score of 4.36 (SD = 1.03), which indicates a high sense of presence [31].

Regarding the informal comments collected during the experiment, many participants stated that the movement of the object was easier than the rotation for both techniques. Some participants had difficulties with the widget rotation because of the limitation of the axes. Also, some participants said that the visual feedback and some auditory feedback helped a lot for using the interaction techniques, but some participants also stated that there were too much different sources of auditory feedback, felt as annoying and distracting.

Question	FHM	Widgets
In which technique you were the fastest?	91.67% (11/12)	8.33% (1/12)
With which technique you were more precise?	58.33% (7/12)	41.67% (5/12)
Which technique would use to do the task again?	83.33% (10/12)	16.67% (2/12)

Question	Yes	No
Do you think the cues improved your performance?	91.67% (11/12)	8.33% (1/12)
Do you think the cues improved your accuracy?	66.67% (8/12)	33.33% (4/12)

Table 4: Subjective questions. Red: <50%, Green: >50%

One participant stated that in his opinion, the cues were only indirectly relevant for performance and accuracy, but instead were more relevant for the usability of the interaction techniques.

5 DISCUSSION

The results show that the technique presented a significant effect on the movement time. The FHM is faster than the widgets for movement tasks. For the rotation time, the FHM was also faster, but there was no significant effect. From this results, we recommend using the FHM in cases where the required time for a manipulation task is the most important feature.

The cues and the manipulation techniques presented significant effects on the Euclidean error distance. For both techniques, the condition with the cues had significantly worse results. The participants were less accurate with the cues and we assumed that the cues would increase the performance and precision, however, these results might be related to the distracting auditory feedback some participants mentioned in the experiment. A possible explanation would be that the participants used to many cognitive resources to perceive and interpret the cues, which reduced the accuracy. For the rotation error, we did not measure any significant effects. However, most of the participants stated that they subjectively think they were faster and more accurate with the cues. We recommend using less or no cues if accuracy is very important, but use the cues in other cases to increase the usability.

We also ran with a non-parametric Mann-Whitney U Test for the SUS scores. The FHM had a higher score than the widgets and the variants with cues had a higher score than the variants without them. A SUS score of 68 is assumed to be average [25]. Our scores for the different conditions are

in a similar area. The FHM with cues was the best with a score of 71.7. However, the statistical analysis showed that the differences are not significant. This might be due to a low sample size and can be researched further in an additional experiment in the future.
The subjective data showed that most of the participants felt faster with the FHM in contrast to the widgets. This matches the statistical analysis for the performance. For the accuracy, there was no clear favorite. This was also underlined by the statistical analysis of the error distance.
Finally, the experiment showed that most participants preferred the FHM. They also had better results using this technique. Also, they liked the cues (except too much auditory feedback). According to the results, we would suggest the use of FHM over Widgets in order to have better manipulation times, while offering cues involving auditory feedback on the manipulation widgets only. However, further investigation is needed to determine why the cues decreased the movement accuracy. It is possible that this changes if some distracting auditory feedback would be removed and only the cues that were enjoyed by the participants would be used. Since some users also preferred the widgets over the FHM, it would be a good option to have both techniques integrated in a complementary way and let the users decide which technique they prefer.

6 CONCLUSION & FUTURE WORK

We compared a free-hand-manipulation technique with a widget-based technique on a 3D task. In addition to that, we analyzed the influence of cues like combinations of visual and auditory feedback. We found that the FHM technique is preferred by most users and has also better results in terms of performance and a higher SUS score. For the cues we had some controversial results. The participants stated that they liked the cues and had the feeling they improved the results, but our statistical analysis showed that the cues decreased the movement accuracy and had no significant effect on the other dependent variables.
All in all, we can say that we successfully developed a reusable, flexible, adaptable and easy-to-use 3D user interface toolkit, that has a wide variety of widgets and interaction techniques that are useful for many VR and AR applications. It integrates some well-known, some not publicly available and some completely new interaction techniques that can be integrated in any other project.
In the future, it would be possible to extend the toolkit with more widgets, interaction techniques, and other features. Also, it would be possible to improve existing features based on user feedback. Additional studies, evaluations and also informal feedback of developers and users would help to improve the toolkit. Furthermore, we want to add haptic feedback to the toolkit. This is a very interesting point because the user can use his tactile sense to feel the 3D widgets. In addition to that, it would be interesting to analyze the influence of haptic feedback on the performance of the user, as well as on the user acceptance.

REFERENCES

[1] Felipe Bacim de Araujo e Silva. 2015. *Increasing Selection Accuracy and Speed through Progressive Refinement.* Ph.D. Dissertation. Virginia Polytechnic Institute and State University.
[2] Richard A. Bolt. 1980. &Ldquo;Put-that-there&Rdquo;: Voice and Gesture at the Graphics Interface. In *Proceedings of the 7th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '80).* ACM, New York, NY, USA, 262–270. https://doi.org/10.1145/800250.807503
[3] C. W. Borst and A. P. Indugula. 2005. Realistic virtual grasping. In *IEEE Proceedings. VR 2005. Virtual Reality, 2005.* 91–98. https://doi.org/10.1109/VR.2005.1492758
[4] Doug A. Bowman and Larry F. Hodges. 1997. An Evaluation of Techniques for Grabbing and Manipulating Remote Objects in Immersive Virtual Environments. In *Proceedings of the 1997 Symposium on Interactive 3D Graphics (I3D '97).* ACM, New York, NY, USA, 35–ff. https://doi.org/10.1145/253284.253301
[5] John Brooke. 1996. SUS: A quick and dirty usability scale. *Usability Eval. Ind.* 189 (1996).
[6] J. Cashion, C. Wingrave, and J. J. LaViola Jr. 2012. Dense and Dynamic 3D Selection for Game-Based Virtual Environments. *IEEE Transactions on Visualization and Computer Graphics* 18, 4 (April 2012), 634–642. https://doi.org/10.1109/TVCG.2012.40
[7] D.B. Chaffin and G. Andersson. 1991. *Occupational Biomechanics.* Wiley.
[8] Michael Chen, S. Joy Mountford, and Abigail Sellen. 1988. A Study in Interactive 3-D Rotation Using 2-D Control Devices. In *Proceedings of the 15th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '88).* ACM, New York, NY, USA, 121–129. https://doi.org/10.1145/54852.378497
[9] Andrew Forsberg, Kenneth Herndon, and Robert Zeleznik. 1996. Aperture Based Selection for Immersive Virtual Environments. In *Proceedings of the 9th Annual ACM Symposium on User Interface Software and Technology (UIST '96).* ACM, New York, NY, USA, 95–96. https://doi.org/10.1145/237091.237105
[10] Andrew J. Hanson. 1992. Graphics Gems III. Academic Press Professional, Inc., San Diego, CA, USA, Chapter The Rolling Ball, 51–60. http://dl.acm.org/citation.cfm?id=130745.130757
[11] Stephanie Houde. 1992. Iterative Design of an Interface for Easy 3-D Direct Manipulation. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '92).* ACM, New York, NY, USA, 135–142. https://doi.org/10.1145/142750.142772
[12] Jeff Hultquist. 1990. Graphics Gems. Academic Press Professional, Inc., San Diego, CA, USA, Chapter A Virtual Trackball, 462–463. http://dl.acm.org/citation.cfm?id=90767.90901
[13] J. Jacobs and B. Froehlich. 2011. A soft hand model for physically-based manipulation of virtual objects. In *2011 IEEE Virtual Reality Conference.* 11–18. https://doi.org/10.1109/VR.2011.5759430
[14] Robert S. Kennedy, Norman E. Lane, Kevin S. Berbaum, and Michael G. Lilienthal. 1993. Simulator Sickness Questionnaire: An Enhanced Method for Quantifying Simulator Sickness. *The International Journal of Aviation Psychology* 3, 3 (1993), 203–220. https://doi.org/10.1207/s15327108ijap0303_3 arXiv:https://doi.org/10.1207/s15327108ijap0303$_3$
[15] R. Kopper, F. Bacim, and D. A. Bowman. 2011. Rapid and accurate 3D selection by progressive refinement. In *2011 IEEE Symposium on*

3D User Interfaces (3DUI). 67–74. https://doi.org/10.1109/3DUI.2011.5759219

[16] Eugene Krivoruchko and Keiichi Matsuda. 2018. Leap Motion: Designing Cat Explorer. http://blog.leapmotion.com/designing-cat-explorer/. Accessed: 14.11.2018.

[17] J.J. LaViola, E. Kruijff, D.A. Bowman, R.P. McMahan, and I. Poupyrev. 2017. *3D User Interfaces: Theory and Practice.* Addison-Wesley.

[18] Jiandong Liang and Mark Green. 1994. JDCAD: A highly interactive 3D modeling system. *Computers & Graphics* 18, 4 (1994), 499 – 506. https://doi.org/10.1016/0097-8493(94)90062-0

[19] Paul Lubos, Gerd Bruder, Oscar Ariza, and Frank Steinicke. 2016. Touching the Sphere: Leveraging Joint-Centered Kinespheres for Spatial User Interaction. In *Proceedings of the 2016 Symposium on Spatial User Interaction (SUI '16)*. ACM, New York, NY, USA, 13–22. https://doi.org/10.1145/2983310.2985753

[20] Daniel P. Mapes and J. Michael Moshell. 1995. A Two-handed Interface for Object Manipulation in Virtual Environments. *Presence: Teleoper. Virtual Environ.* 4, 4 (Jan. 1995), 403–416. https://doi.org/10.1162/pres.1995.4.4.403

[21] Keiichi Matsuda. 2018. Twitter: Introducing Virtual Wearables. https://twitter.com/keiichiban/status/976850288949809154. Accessed: 14.11.2018.

[22] Wayne Piekarski and Bruce H. Thomas. 2003. Augmented Reality User Interfaces and Techniques for Outdoor Modelling. In *Proceedings of the 2003 Symposium on Interactive 3D Graphics (I3D '03)*. ACM, New York, NY, USA, 225–226. https://doi.org/10.1145/641480.641526

[23] Ivan Poupyrev, Mark Billinghurst, Suzanne Weghorst, and Tadao Ichikawa. 1996. The Go-go Interaction Technique: Non-linear Mapping for Direct Manipulation in VR. In *Proceedings of the 9th Annual ACM Symposium on User Interface Software and Technology (UIST '96)*. ACM, New York, NY, USA, 79–80. https://doi.org/10.1145/237091.237102

[24] I. Poupyrev, T. Ichikawa, S. Weghorst, and M. Billinghurst. 1998. Egocentric Object Manipulation in Virtual Environments: Empirical Evaluation of Interaction Techniques. *Computer Graphics Forum* 17, 3 (1998), 41–52. https://doi.org/10.1111/1467-8659.00252

[25] Jeff Sauro. 2011. MeasuringU: Measuring Usability With The System Usability Scale (SUS). https://measuringu.com/sus/. Accessed: 16.11.2018.

[26] Ken Shoemake. 1992. ARCBALL: A User Interface for Specifying Three-dimensional Orientation Using a Mouse. In *Proceedings of the Conference on Graphics Interface '92*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 151–156. http://dl.acm.org/citation.cfm?id=155294.155312

[27] Sven Strothoff, Dimitar Valkov, and Klaus Hinrichs. 2011. Triangle Cursor: Interactions with Objects Above the Tabletop. In *Proceedings of the ACM International Conference on Interactive Tabletops and Surfaces (ITS '11)*. ACM, New York, NY, USA, 111–119. https://doi.org/10.1145/2076354.2076377

[28] A. Talvas, M. Marchal, and A. LÃĺcuyer. 2013. The god-finger method for improving 3D interaction with virtual objects through simulation of contact area. In *2013 IEEE Symposium on 3D User Interfaces (3DUI)*. 111–114. https://doi.org/10.1109/3DUI.2013.6550206

[29] Amy Ulinski, Catherine Zanbaka, Zachary Wartell, Paula Goolkasian, and Larry Hodges. 2007. Two Handed Selection Techniques for Volumetric Data. 0 (01 2007).

[30] A. C. Ulinski, Z. Wartell, P. Goolkasian, E. A. Suma, and L. F. Hodges. 2009. Selection performance based on classes of bimanual actions. In *2009 IEEE Symposium on 3D User Interfaces*. 51–58. https://doi.org/10.1109/3DUI.2009.4811205

[31] Martin Usoh, Ernest Catena, Sima Arman, and Mel Slater. 2000. Using Presence Questionnaires in Reality. *Presence: Teleoper. Virtual Environ.* 9, 5 (Oct. 2000), 497–503. https://doi.org/10.1162/105474600566989

[32] Noah Zucker. 2018. Twitter: Exploring playful interactions in VR. https://twitter.com/Noah_Zr/status/1031931236447604736. Accessed: 14.11.2018.