Interaction Design for Mobile Virtual Reality

Daniel Brenners

I. Abstract

Mobile virtual reality systems, such as the GearVR and Google Cardboard, have few input options available for users. However, virtual reality technology affords almost limitless possibilities for interacting with immersive worlds. Because there are only a limited number of input options available, most applications currently consist of highly specialized tasks using control methodologies unique to that application. This prevents users from establishing a set of expectations across mobile VR and increases the learning curve for each new application. In order to establish a possible standard for interaction methodologies, this paper observes what general cases should input devices be able to handle, and how they should handle them. It identifies object manipulation and navigation as core abilities that a virtual reality platform must afford. It then examines an important secondary application, text entry, and how mobile virtual reality may be able accomplish this. Finally, it examines ways to augment existing devices to handle such cases.

II. Literature Review

a. Physiological considerations

Human limitations form many constraints that govern the creation of interaction methodologies and user interfaces in virtual reality. These form a few basic rules that all virtual reality design should be grounded in so that the user is not uncomfortable.

i. Depth Constraints

Distance from the user to objects in a 3D scene must be considered in order to not cause a vergence-accommodation conflict, one of the main causes of visual fatigue.[1] This occurs when the eyes converge at various depths in order to focus on different 3D virtual objects, but the lenses do not accommodate this change, as they are constantly accommodated for the depth of the screen. In order to prevent this, it is best to keep most static UI elements at a virtual distance of 10-20 meters, while some fleeting effects can be shown at up to .5-10 meters away.[2,3] Further than 20 meters away results in a flat looking object.

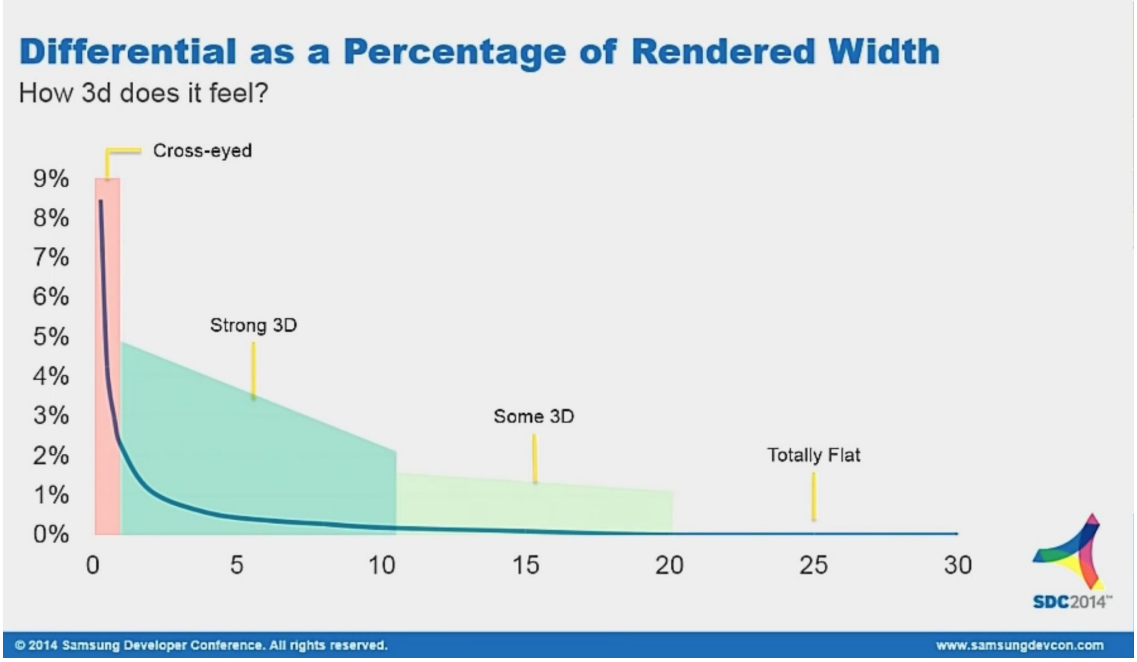

Figure 1: Difference between images sent to left and right eyes changes as a function of distance (shown in meters) from Samsung Research America

ii. Field of View (FOV) Constraints

In order to feel fully immersed in virtual reality, the system being used must surpass several technical specifications in order to establish a sense of presence. One of these is a stereoscopic field of view of at least 90 degrees left to right and up and down, which is achieved in most modern VR system. [4] Position of elements in the field of view plane must also consider neck ergonomics. The head can be comfortably rotated to one side about 30 degrees, with a max of 55 degrees, and there are also well defined constraints for looking up or down for a typical user. [2]

iii. Movement Constraints

Changes in velocity are known to cause issues in many users of virtual reality systems.[5] This is because any mismatches between the visual system and vestibular system may cause extreme discomfort and/or nausea. In this case, the visual system is communicating to the brain that there is an acceleration that the vestibular system is not detecting. Due to this, acceleration in any direction is best avoided when designing navigation systems.

b. Input constraints

The device used in this paper is the Samsung GearVR. The inputs that this device provides are directional tapping inputs that go “up,” “down,” “left,” and “right.” It also supports a main “select” button in the middle of the input surface. The device has a “back” button, but is usually reserved for exiting applications. Head orientation is also tracked, which may be used as a type of input.

c. Approaches to 3D User Interaction

3D user interaction is split primarily into object manipulation and navigation.[6,7] The taxonomies provided by Bowman and Poupyrev are used to ensure a virtual reality system can perform the essential tasks of interacting with a 3D environment. These taxonomies provide basic components that provide the building blocks for more general cases of manipulation and navigation.

i. Manipulation

When attempting to generalize manipulation tasks, we can decompose each task into a subset known as the canonical manipulation tasks. These simulate "acquisition and positioning movements that we perform in the real world."[8] These include selection, positioning, and orientation. [6,7,9]

Selection is the act of acquiring one target from a set of one or more targets. Notable selection methods in the field of 3D interaction include the cubic silk cursor, bubble cursor, and ray casting.[10,11,12,13] The cubic silk cursor uses a translucent cube to encapsulate an object in order to select it without occluding it. The bubble cursor is similar to the cubic silk cursor, but uses a sphere that dynamically changes volume as you move around objects. Although effective at selection tasks, cubic silk cursors and bubble cursor methods are generally less immersive than ray casting due to large translucent objects being placed in front of the users field of view. [8]

Ray casting projects a ray from the user to collide with an object being selected, selecting the object that is closest to user out of the subset that is intersected by the ray. An adaptation from the ray cast method, known as the aperture method, attempts to correct for the high precision needed at far distances by projecting a cone instead of a single ray.[14] Selection methods that use a selector with volume may unavoidably select more than one object at a time. The SQUAD offers a technique that progressively refines the selection until only the desired object remains.[15]

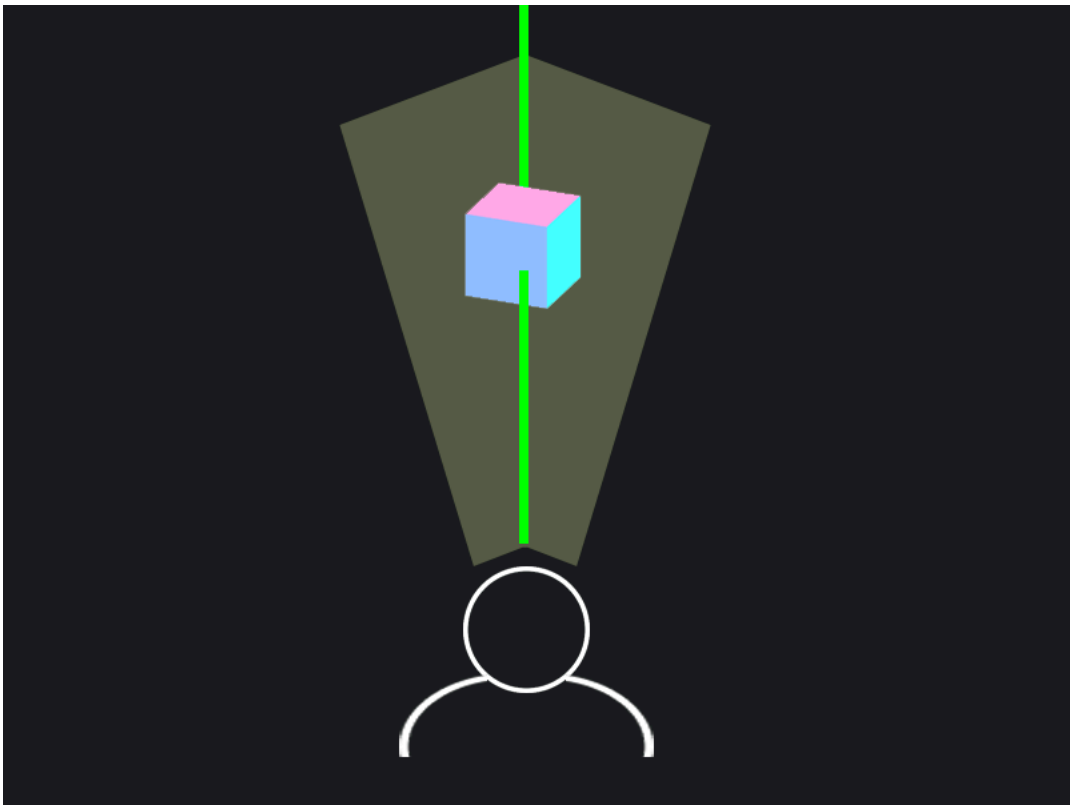

Figure 2: Ray cast (green) intersects a cube

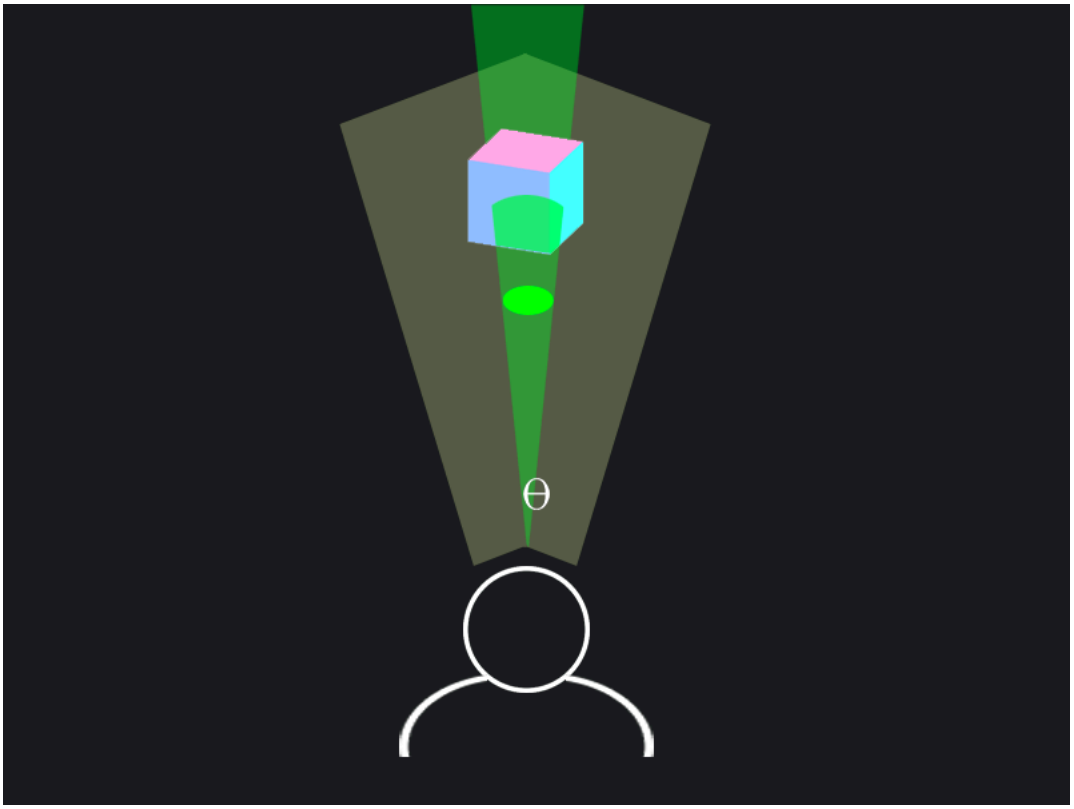

Figure 3: Aperture method, with apex angle θ

Previously studied positioning and orientation techniques largely rely on hand tracking to manipulate an object.[16] These techniques are divided into isomorphic and non-isomorphic. Isomorphic refers to a geometric one-to-one relationship between the real world motion and the manipulation occurring in the virtual world. Isomorphic manipulation provides much more natural, and hence immersive, manipulation, but suffers from devices that cannot provide such a mapping and human limitations that prevent such a relationship.[17]

ii. Navigation

Navigation refers to moving in space within a virtual environment. Task decomposition for navigation includes direction selection, velocity, and input conditions. [6] Direction selection is the way a user determines which way he or she will travel. Velocity is the speed at which the user travels through virtual space, and the input conditions specify how the user starts and stops a navigation task.

A common navigation technique is gaze-directed steering, which moves the user in the direction he or she is looking. Although seemingly restrictive, it has been found that strafing (moving side to side) or moving up and down is uncomfortable compared to moving in the direction of the user's gaze.[8] Non-gaze based methods include hand pointing, eye tracking, and torso-directed steering. [8,18] An alternative to steering techniques is route planning. Route planning techniques provide a path that the user will automatically follow once created. Techniques to create the path include drawing the path or defining single points that create a path.[19]

A final navigation technique is teleportation, which immediately places the user to a desired location once he or she selects the destination. Some teleportation methods include selecting the location with a ray cast, pointing to the location on a miniature version of the world, and explicitly entering the coordinates of the location. [8,20] While quick and efficient, these methods are less immersive and tend to "reduce the user's spatial awareness." [6]

d. Approaches to Text Entry

Effective text entry is crucial for mobile virtual reality to become not just an entertainment technology, but a viable computing platform. Text entry, in particular, is crucial for tasks such as messaging, entering a URL, and command line entry. While using speech-to-text is useful in certain areas, it fails in many scenarios. It suffers in noisy areas and has been shown to be inefficient in text-editing and manipulation.[21] Users also may find that audibly speaking their input is a breach in privacy and socially awkward.[8] Furthermore, it has been shown that performing mentally taxing actions, such as composing a message, can be done much more effectively by typing while thinking than by talking while thinking.[21]

Current text entry in mobile virtual reality relies on a hunt-and-peck method, using the user's gaze to select, which can be taxing on the users neck and suffer from the same deficiencies as ray casting small objects.[8] An alternative is to provide a separate input device to handle text entry. There have been many attempts to solve the problem of input in immersive

environments through input devices, and they generally fall into five separate categories: chorded keyboards, pen and tablet interface, gloves, natural hands, and navigation based. Chorded keyboards have a steep learning curve, which is not ideal for wide adoption. Pen-based methods require a user to hold their hands up for long periods of time, which can be tiring. Using positional tracking for hands over a virtual keyboard requires the user to hold keep their hands up, tiring the user and offering no haptic feedback. Navigation techniques, like Dasher and Hex, represent a very interesting alternative to text entry, but result in low words per minute, no haptic feedback, and need a user's complete visual attention at all times.[22,23]

Glove based methods, on the other hand, offer a possible solution with varying results. In general, these methods offer haptic feedback, the use of natural hand gestures, the possible inclusion of other device inputs, and are light and mobile. There are a couple of prominent glove based keyboards, but both have drawbacks that prevent wide adoption. The pinch keyboard requires large movements by the user to select a row on the keyboard and has been shown to not take advantage of muscle memory from conventional keyboard use.[24] The VType keyboard lacks a virtual representation of the keyboard and the ability to correct word predictions.[25]

e. Extension to Lower Input Options

One hope of this paper is to begin establishing a standard of input interactions, but it is clear the most popular form of consumer virtual reality, Google Cardboard, does not provide enough input options to become a viable standard. Instead of completely ignoring this immensely prevalent platform, we attempted to cheaply augment it so that it can afford the same input options the Samsung GearVR can.

It has been shown that "extension stickers" can be developed using silver conductive ink on paper or plastic. [26] These stickers can extend the abilities of capacitive touch screens and allow for interesting tangible interfaces. Innovations like these offer possible low cost solutions for augmenting an existing platform to meet a standard that requires more input possibilities

III. Methods

To develop and prototype designs, we used Unity 3D software with a Samsung GearVR and Samsung Galaxy S6. Custom plugins were developed to allow for rapid prototyping of different input events from the GearVR headset and Google Cardboard in Unity 3D.

Since 3D scenes were computationally expensive for our hardware, the device we used would frequently overheat and prevent user testing. To alleviate this, we developed an algorithm to reduce the polygon count of the scenes we had already created. This effect had the added benefit of an interesting artistic design that users thought was more immersive than attempting to be as realistic as possible. This could be from the increased performance, and possibly higher frame rate, or something similar to moving away from an uncanny valley.

To increase immersion, we also spent time creating audio cues in our virtual scenes. Campfires and natural sounds proved to be immensely important for users to feel more immersed in an

application. Three dimensional audio, in particular, increased immersion in. Anecdotally, we found that this was partly because it helped prevent users from hearing sounds from the real world, which confused a few users.

Before we analyzed any selection tasks, we first had to develop a way to demarcate where a users gaze was while being subtle enough to not break immersion. To indicate where the gaze of a user currently is, we used a reticle to mark the point in space directly in the middle of the gaze. This was to be used for all selection tasks. To create a reticle in 3D space, we had a flat circle to follow the user's field of view. The reticle is projected onto the first object in the field of view as to avoid any depth related issues (seeing two reticles at once). Because the reticle is projected at a distance, it also must scale linearly with the distance from the user in order to create the illusion that it is maintaining the same size. The reticle must also not react to any light sources in the environment or be occluded by other objects, so a unique shader must be created to draw the reticle last in the scene at each frame.

At close distances to objects, users reported that the reticle portrayed a false sense of distance, creating the illusion that the object was in fact further away than it was. To alleviate this problem we scaled the size of the reticle with distance from an object at close distances. The best performing formula that relates size according to distance was $y = 1 + 2e^{-x}$, with x being the distance in meters and y being the relative scale of the reticle.

To develop the text entry data gloves we applied copper tape to the finger tips of nylon gloves. Jumper wires were used to connect these copper pads to an Adafruit Bluefruit EZ-Key device. The thumb was connected to ground, which when connected to other copper pads on the glove would send a signal to the Bluefruit EZ-Key to send a BlueTooth signal of a specific key to a laptop. Each finger (except the thumbs) corresponded to a certain letter, which would later correspond to a certain set of keys. The left pinky, left ring finger, left middle finger, and left pointer finger correspond to "A," "S," "D," and "F" respectively. Conversely the right pinky, right ring finger, right middle finger, and right pointer finger correspond to ";," "L," "K," and "J" respectively. There were three separate, non lettered inputs available. The side of the right pointer finger corresponds to an "ENTER," the side of the right middle finger corresponds to "SWAP WORD," and the side of the left pointer corresponds to "DELETE."

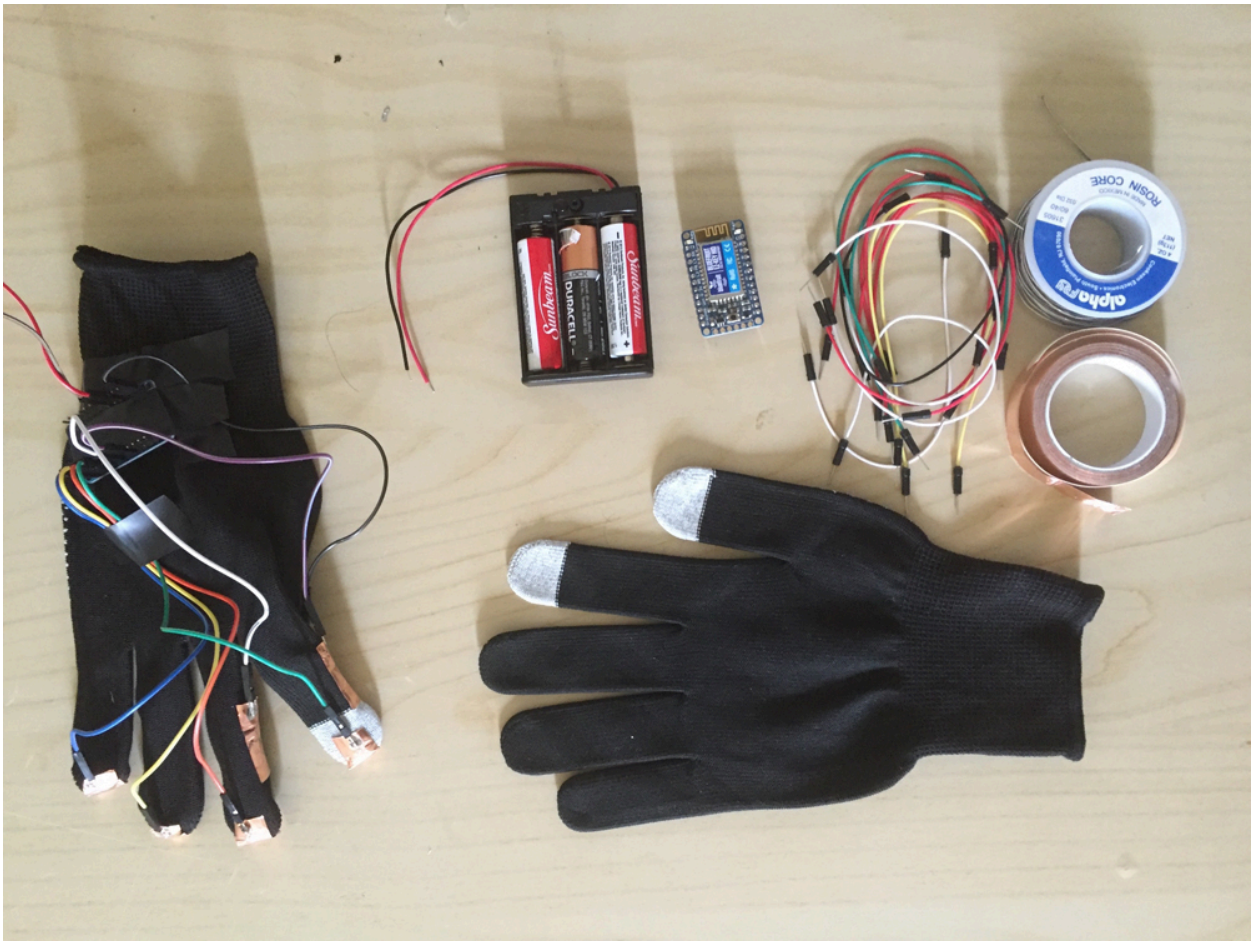

Figure 4: Gloves, batteries, jumper wires, copper table, and Bluefruit EZ-Key

The laptop would receive the various keys for a word, and when the enter key is received, it would send this word to a Flask server to process which word the user intended. Each letter sent to the Flask server was assigned a set of other letters that the user could have intended. For instance, if an “A” was received, that letter could have been intended to be “Q,” “A,” or “Z.” The final implantation also accounts for the letter possibly being “W,” “S,” or “X,” since these keys are assigned by a neighboring finger, although these letters were assigned a lower probability.

To determine which word was intended, we used a text corpus provided from Leipzig Corpora Collection, consisting of 300,000 sentences from 2015 news articles. This became our dictionary of unigrams, bigrams, and trigrams. The collection of unigrams was also made into a trie data structure. To determine what word a collection of letters could refer to, each combination of letters were checked against the trie. In order for the combination of letters to not increase geometrically with time, each substring was checked against the trie and possibilities were pruned accordingly.

For example, if the combination of letters entered was “sda,” the first step would be to analyze the “s” and determine that the first letter could be “w,” “s,” or “x.” There are valid words that

begin with each of these letters, so nothing is eliminated. The next letter is “d,” so now the possibilities grow to “we,” “wd,” “wc,” “se,” “sd,” “sc,” “xe,” “xd,” and “xc.” We can immediately eliminate “wd,” “wc,” “sd,” “xd,” and “xc.” This process continues until we have only valid letters left. We then check these words against our list of unigrams to determine which of these possibilities is the most popular word. Lastly, if possible, we check the previous two words to check for bigram and trigram frequency and weigh these into our decision to finally display a word on the screen. The list of words is kept in a prioritized list, so that if need be, a user can flip through possibilities in descending prominence.

For the frontend of the application, we developed a visual cue for users to test. The cue was an interactive keyboard that reacts to key presses. The keyboard hah suggestive spacing for users to infer which finger is accountable for which keys.

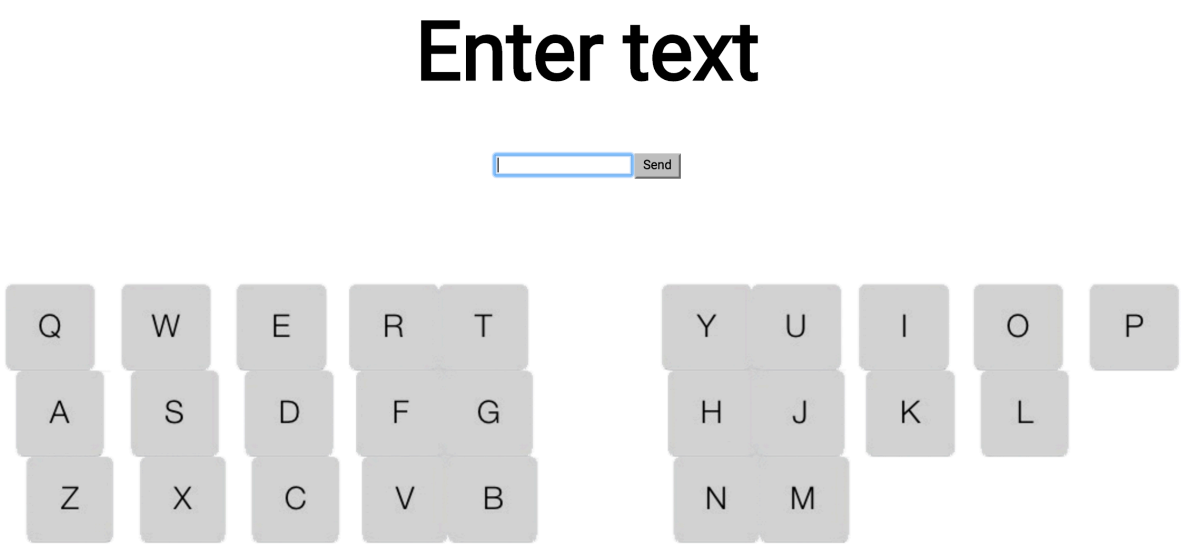

Figure 5: User interface for text entry input device

We developed a printable schematic for developers to use that will allow those with Google Cardboards or similar mobile phone based virtual reality systems to use “up,” “down,” “left,” “right,” “select,” and “back” buttons. The designs were tested with paper and a conductive silver ink pen. Prototypes can be printed on paper using regular ink, which can be traced with a conductive ink pen, or users can fit inkjet printers to print conductive ink directly on the paper itself and fold the paper as defined in the instructions. This project made us more comfortable with creating a possible standard with this input layout. If this, or a similar, type of methodology were to spread, it would be at least possible to retrofit these devices to match the

standard.

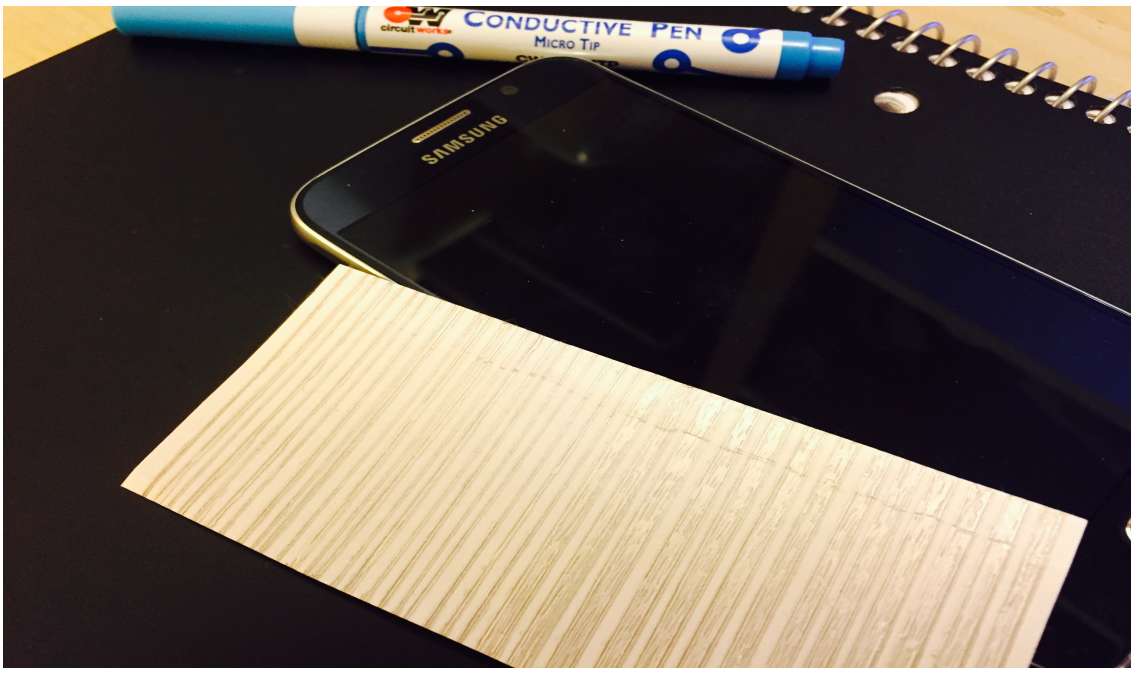

Figure 6: Testing optimal thickness for conductive lines on a capacitive touchscreen phone.

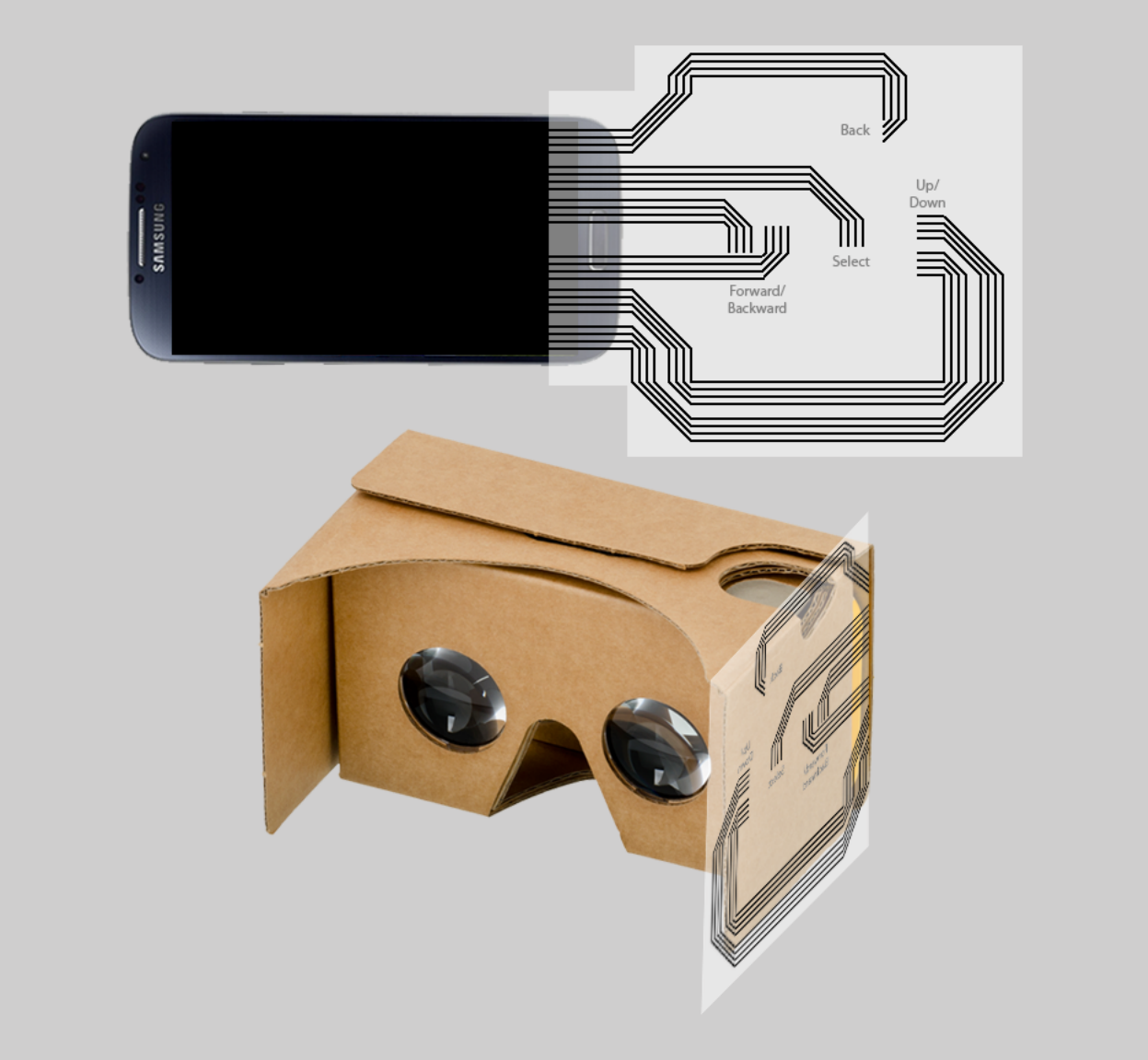

Figure 7: Printable or traceable design for augmenting Google Cardboard

IV. Experimental Design

The following section outlines the experimental design of determining the optimal mapping of canonical manipulation and navigation tasks to the Samsung GearVR. The GearVR has a track pad that offers an “up,” “down,” “left,” and “right” input, as well as a button we will call the “select” button. It also offers a “back” button, and tracks the user’s gaze, which can be thought of a type of input.

Two separate 3D environments were created to test manipulation and navigation based tasks. For these experiments, 3 subtasks were examined. Subtask 1 consisted of selecting a cube out of a group of cubes. Subtask 2 consisted of selecting the correct cube by rotating cubes to locate the cube with the correct symbol on the back. Subtask 3 consisted of selecting the correct cube, then moving it to the desired position and scaling it appropriately. At each

subtasks, users were split into 2 or more groups, with each group using a different methodology. The ideal methodology was determined, and then controlled for as the group moved on to build upon this subtask. After subtasks 4 was completed, certain cosmetic changes were then examined to optimize for immersion.

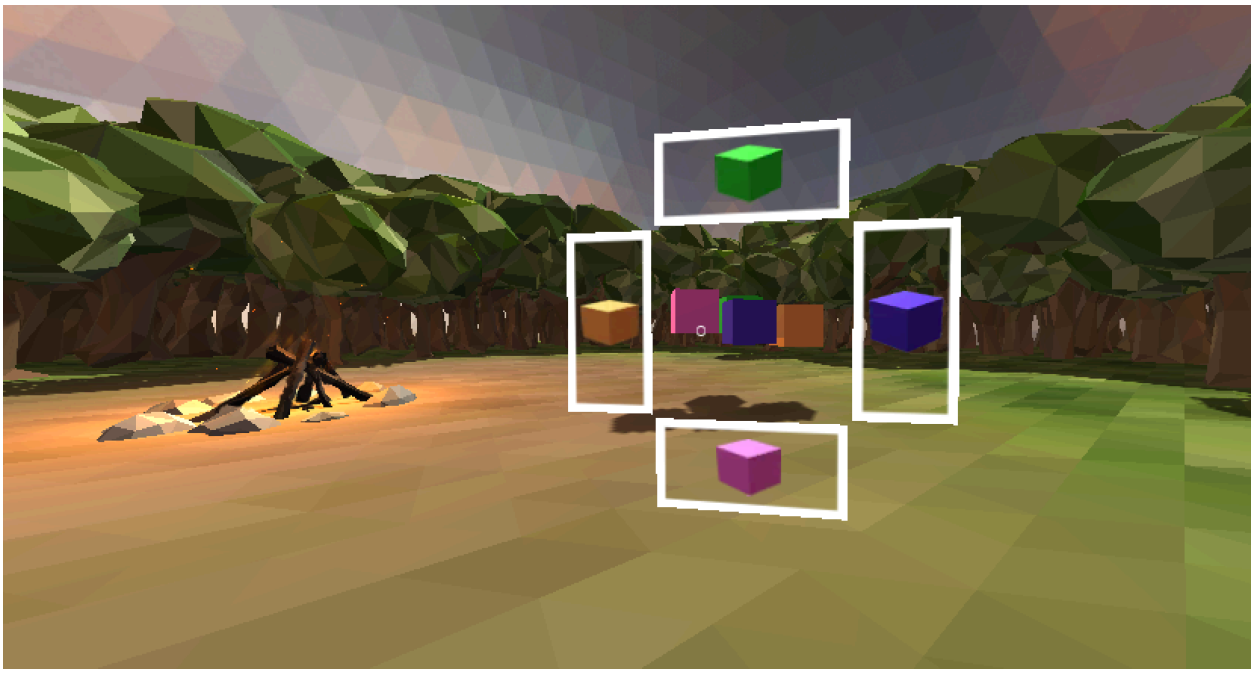

Figure 8: Subtask 1 in the manipulation experiment

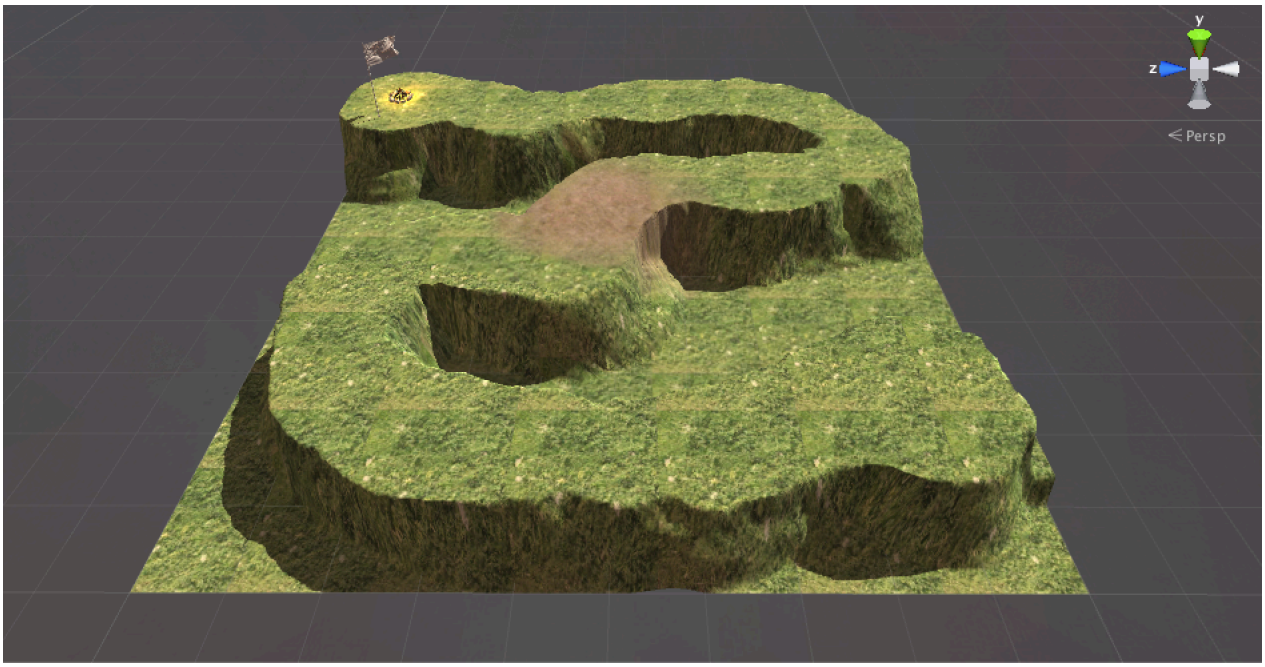

Figure 9: Short distance navigation world

Performance metrics used to evaluate each tasks were adopted from Poupyrev and Foley.[7,9] These included completion time, error rate, ease of use, sense of presence, and fatigue susceptibility. Completion time was measured in seconds from the beginning to the end of the task. Errors identified were missing a selection, dropping the cube, over rotating the cube, and hitting the container with the box. Ease of use, sense of presence, and fatigue were both evaluated qualitatively by a Likert-type scale in a survey after each trial.

Subtask 1 examined two groups (n = 7 for each group), in which group A used a simple ray casting method, and group B used an aperture method. Groups were reshuffled then used two different highlighting cues for selecting an object, testing for immersion. Each group then tested both types of highlighting for an informal survey on which they preferred. Groups were again created to test for selecting one cube out of a bundled group with the aperture method. Group A selected the group, but then used a dropdown menu to select the appropriate cube. Group B was given a radially oriented menu instead.

Subtask 2 examined two groups, which controlled for the selection method but differed on how the groups orientated the cube. Group A used the “push” metaphor which had them drag the cube using the select button and a side swiping gaze, while the group B used a method that changed orientation according to gaze angle (or head orientation). Groups were reshuffled and group A and B were controlled for the latter orientation method, but differed on input conditions: group A would need to continually hold the “select” button to keep the object selected, while group B did not. Groups were then recreated to test for different speeds of cube rotation. This was a more iterative approach that tested different neck angles to reach a 90 degree turn, from 15, 20, 25, 30, and 35 degree neck turns.

Subtask 3 controlled for the aperture method with the gaze-based orientation method. Group A used “left” and “right” controls to move the object forward and back, with “up” and “down” scaling the object. Group B used “up” and “down” controls to move an object forward and back, with “left” and “right” scaling the object.

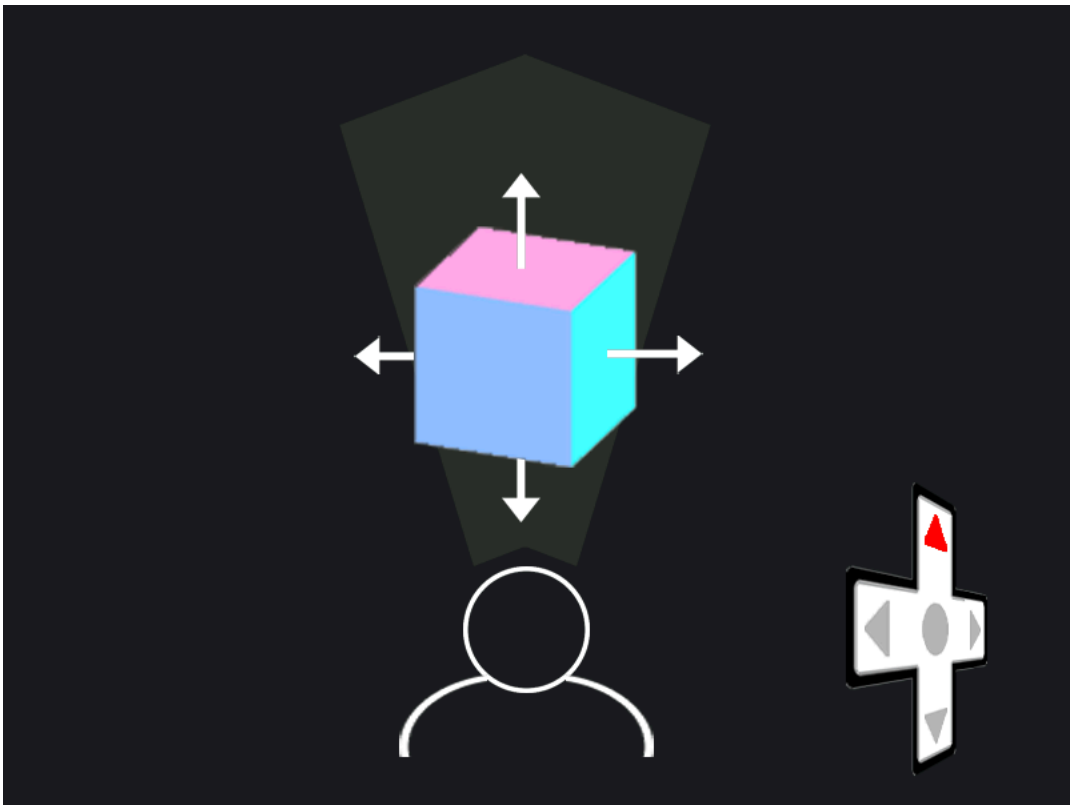

Figure 10: Using the touch pad to increase the scale of an object

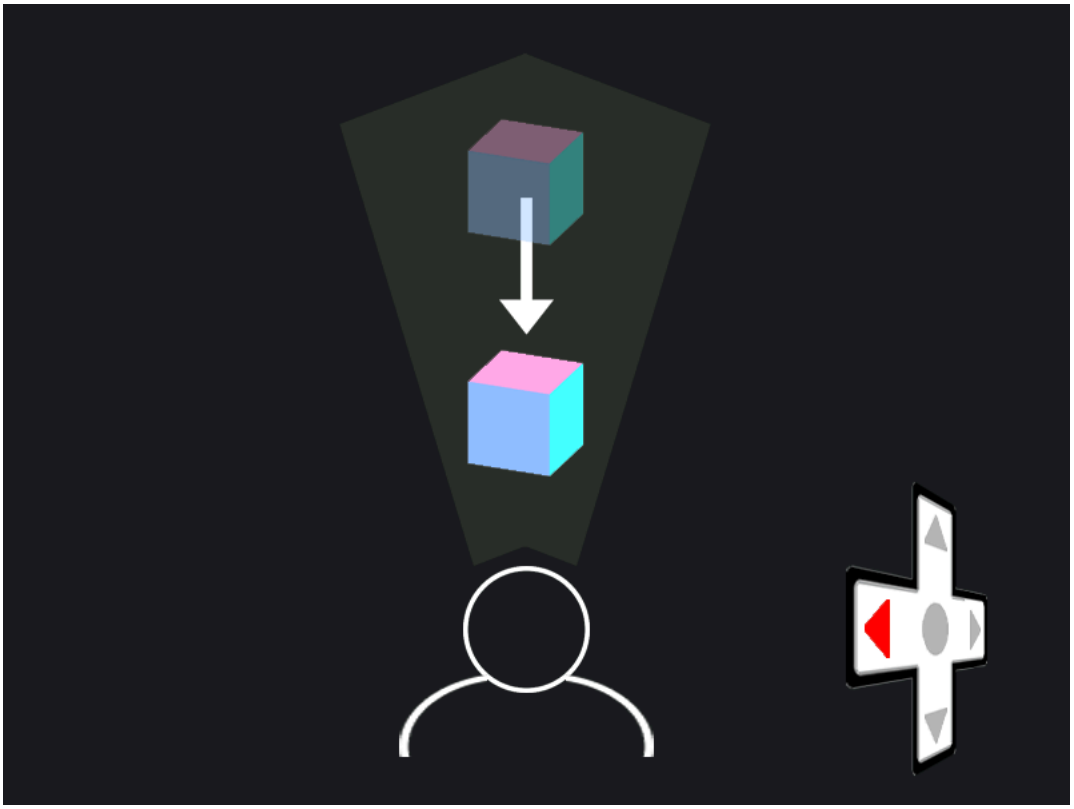

Figure 11: Using the touch pad to control the z-depth of an object

For the navigation experiments, a separate 3D environment was created. Users were given two subtasks, one involving short range navigation and one requiring long range navigation. Performance metrics included completion time, ease of use, sense of presence, and fatigue susceptibility. Navigation techniques included gaze-based steering, teleportation, navigation objects, and terrain manipulation. Gaze-based steering had the users press the “up” button, which would propel them forward at a constant rate until they released the button. Teleportation methods were implemented by having a user select a point on the ground, which would instantly bring them to that point. To use navigation objects, users select a teardrop shaped object and from the user interface select the “Travel Here” option. Terrain manipulation had users manipulate terrain itself to push it away, and the user could then zoom back in at a different location.

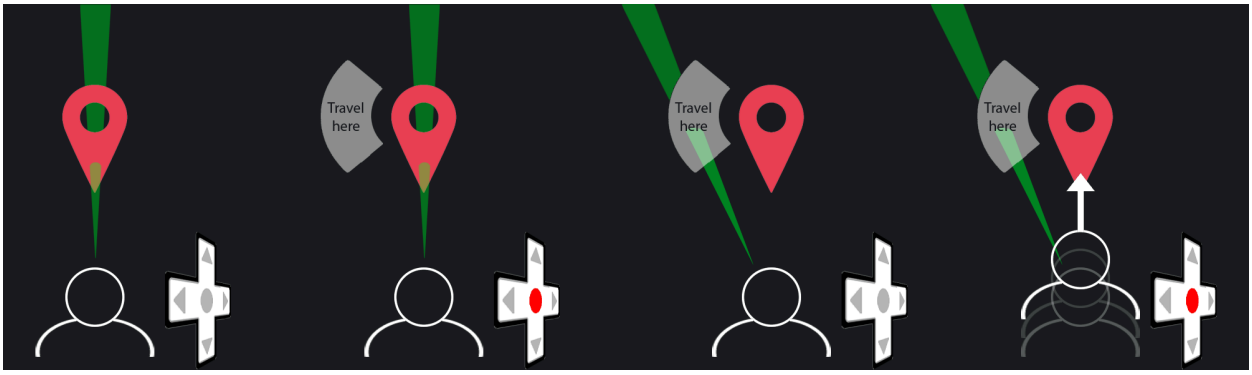

Figure 12: Using navigation objects

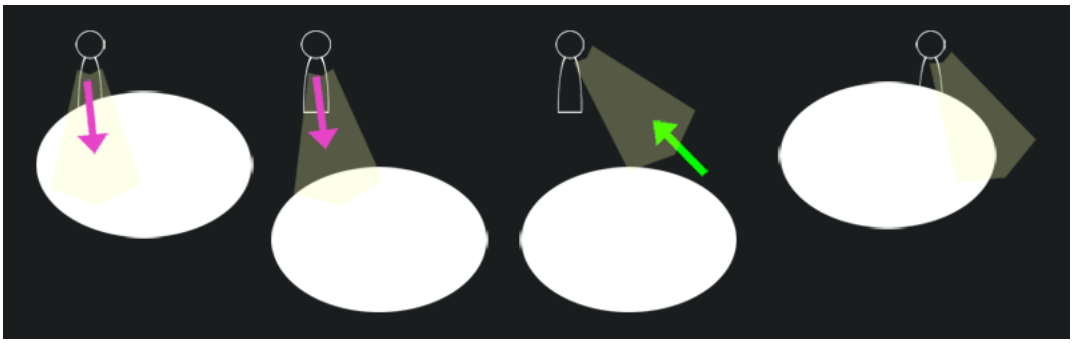
Figure 13: Terrain manipulation as a means of navigation

For short range navigation, we first tested everyone tested three methods: gaze-based steering, teleportation, and automated routes with navigation objects. Each group traversed a virtual terrain using their method. After this task was completed, all participants were tested again with gaze-based steering, but using three groups to test for ideal forward velocity and spin velocity. We then split the group into two groups (n = 7) to test input conditions. The first group needed to continuously hold the “up” button to travel, while the second group pressed it for starting and then again for stopping.

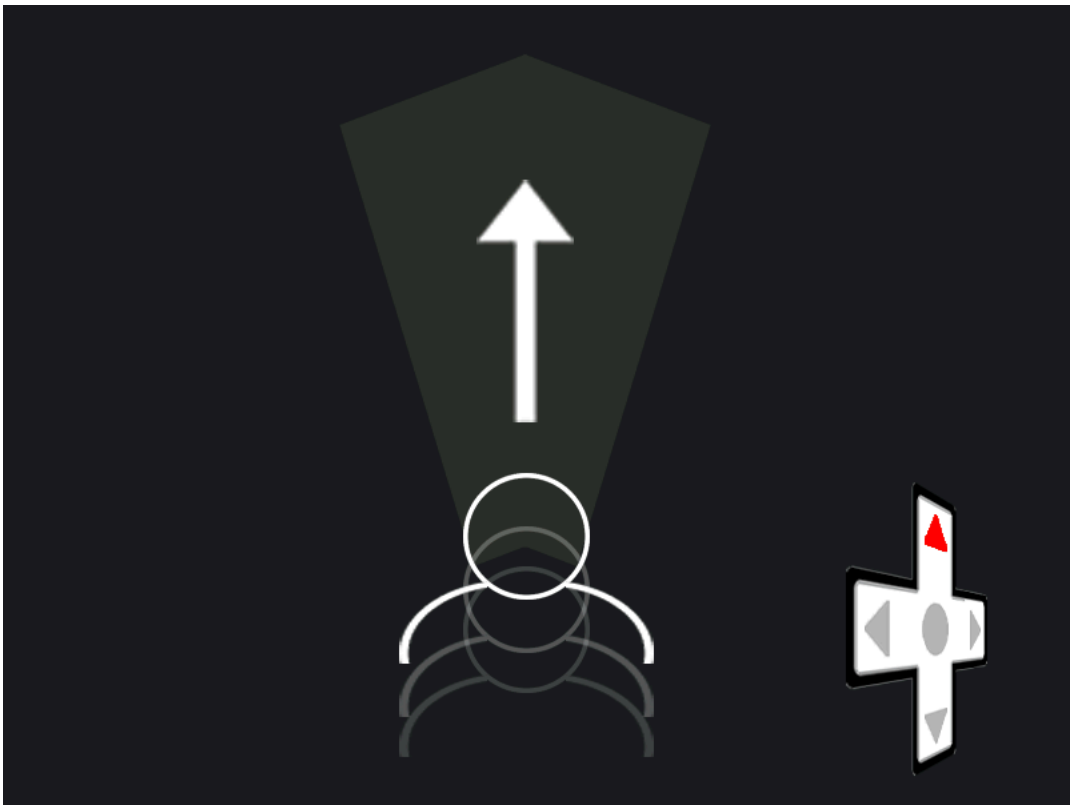

Figure 14: Using the touch pad to initiate gaze-based steering navigation

For long range navigation, we tested two different methods: teleportation and terrain manipulation. We tested all participants on teleportation and placed them into two groups (n=7). Group A had instantaneous jump cuts between locations and group B had fade to black before the transition. We then placed participants into two groups (each n = 7) to test teleportation versus terrain manipulation methods for long distances.

To test the text entry input device, we first oriented users with the device and had them type 5 sample sentences, while keeping track of their words per minute count. We then had users type the sentences with both gaze based methods and with the text entry device (n=7 for each group). We also tested the text entry input device with visual feedback and no visual feedback (n=7 for each group). Performance metrics include error rate, completion time, and comfort level.

V. Results and Analysis

a. Manipulation

i. Selection

While pointing the reticle at an object is a relatively straightforward selection mechanism, we tested two different subtleties. The first was the simple ray casting method. This method projects a single line from the users field of view. To select, the ray must intersect a selectable object. We also used an aperture technique using an conic volume with an apex angle of 3 degrees. Our experiments showed that while with an appropriate apex volume of 3 degrees, the aperture technique was more accurate than the ray casting counterpart, especially at objects at longer distances.

The aperture method necessitated a way to account for query to result in multiple selected objects. To remedy this, we used a similar method to Kopper et al, and after a selection occurs, the user chooses the appropriate item from a user interface.[15] This user interface can be presented in a number of ways, and may be unique to the situation. However, we did test two different types of UI orientations. The first was a typical dropdown menu and the second was a radially oriented menu. In our tests, users reported a preference for the radial menu, as it required much less steering than the drop down menu. However, they performed similarly well in accuracy and error rate.

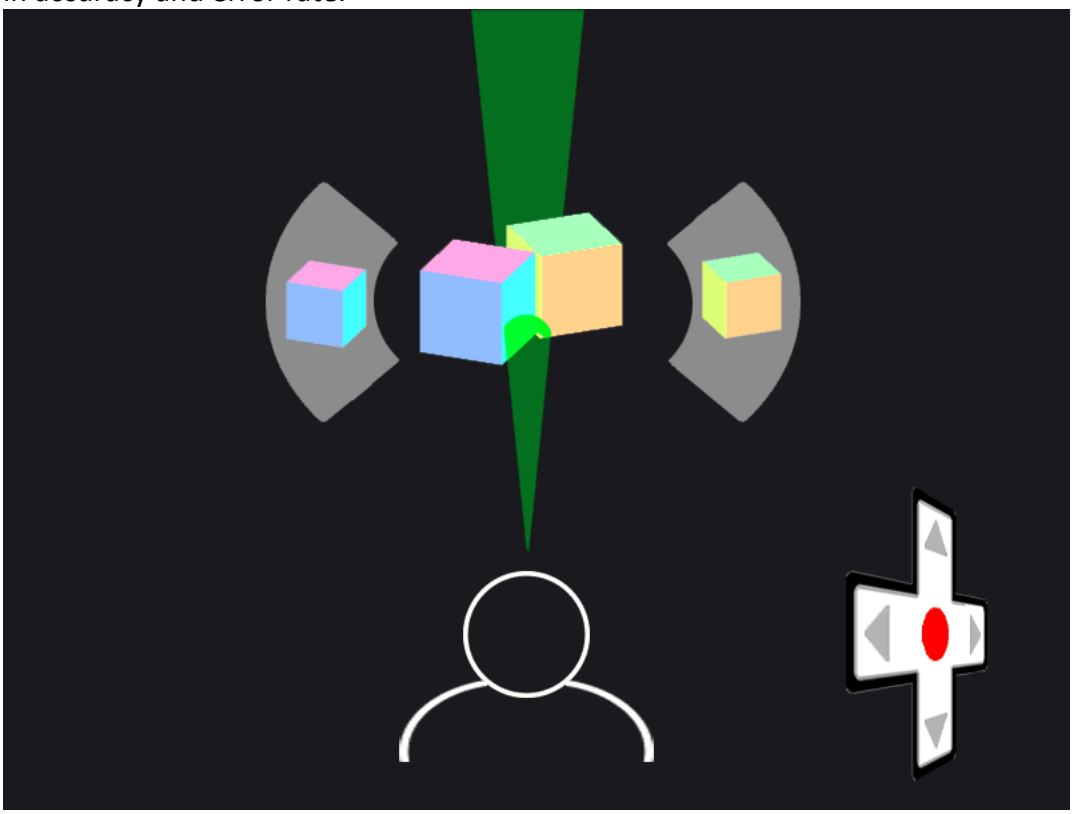

Figure 15: Selecting group initiates a radial user interface to choose from

We preferred the radial menu because it ensured the shortest possible path for all types of selections, and minimizes narrow steering of dropdown menus. This optimizes according to Fitz law and the corollary Steering law developed by Accot and Zhai.[27]

To select an object, we used the main button or "select" functionality in our input schematic. This was intuitive for users, as it is very similar to conventional mouse-like input hardware. There are two types of selection for objects. One tap reveals an objects user interface, to show any actions a user may perform with that object. The second way to select is by holding down the select button to enter "transformation mode." With few input options available, we had to rely on using the same button in multiple ways while being similar in use. In this case, the "select" button was selecting an object, but in two different ways.

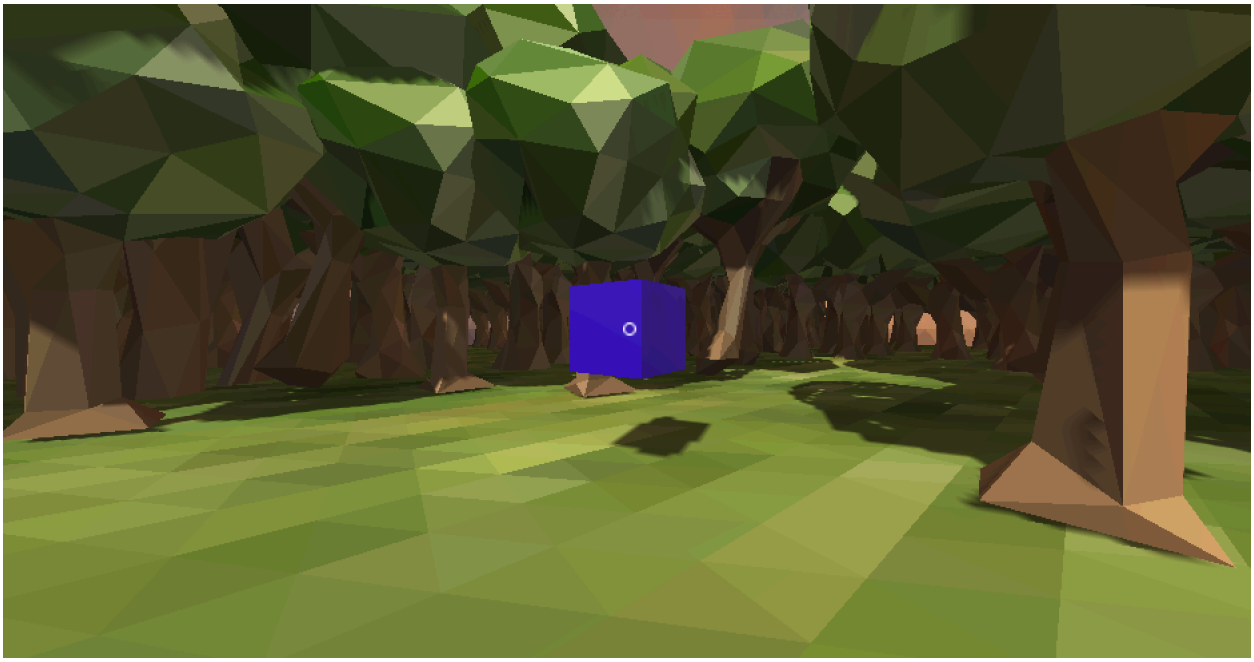

Figure 16: A user in transformation mode

Selection feedback was used in order to convey that an object is selectable or currently selected. This was done using audio and visual feedback. For visual feedback, we tried using highlighting or a change of color, but this broke immersion for many users. Instead when an object could be selected, the reticle changed to a shade of gray, instead of typical white. When selected, the object would reveal its user interface. If there is no user interface, the object could still be manipulated with by holding down the "select" button and entering transformation mode.

ii. Transformation

Transformation in virtual reality consists of a performing affine transformations on the object. In particular, we attempted to incorporate object translation in space, rotation, and changing

scale. To do this, we needed to reuse many of the inputs that we will use for navigation. This necessitated a sort of “transformation mode” that a user could enter and exit at will.

As mentioned before, to enter transformation mode, a user simply selected an object by holding down the main button. When transformation mode is enabled, the object is placed directly in front of the user for the user to observe. To exit transformation mode, the user again holds the main button down for a short period of time, and the object is released from the user’s gaze (it may return to the original position or stay where it currently is, depending on the application).

We tested an object attachment mode that required the user to keep holding down the button for the entire duration of transformation mode and release upon exit. For periods longer than 30 seconds, users reported arm fatigue. We thought the first option was more ergonomic and opted for object attachment that did not require any user button interaction for the duration of transformation mode.

To move the object closer or further, experiments showed that users prefer the two lateral buttons. Due to the orientation of these buttons, they orient themselves to a natural forward and backward positions that felt intuitive for users. To translate an object along the viewing “plane” (up, down, left, and right) the user needs only to move his or her head in that direction, as the object is attached to the gaze. This isn’t actually a plane that the objects is moving across, but a sphere. But it was sufficient to move the object anywhere in space that a user wanted to, with the added use of forward and backward movement.

Changing object orientation was difficult to achieve. Especially given the input real estate already given up by previous transformations. We tried several different methodologies, such as a dragging gesture that served as a metaphor for spinning the object with a gaze-based input. With this method, the user presses and holds down the “select” button and moves their head in a swiping motion, then releasing to spin an object. We tested this against a gaze-based orientation method. This method allows the user to rotate the object while the user moved his or her gaze. For instance, if the user wants to look at the right side of a box (from the perspective of the viewer) currently in transformation mode, the user would look left. Instead of the box rotating with the viewer so that the front side of the box is always facing the user, the box reveals the right side, essentially rotating the opposite direction of the viewers head.

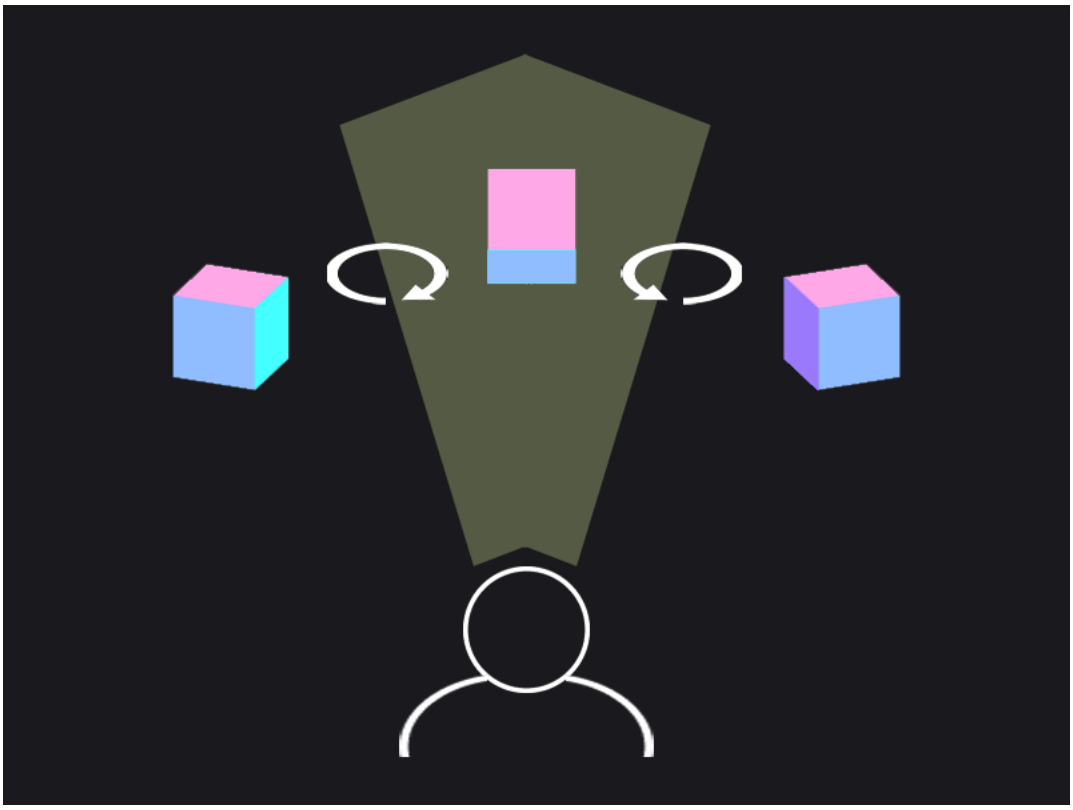

Figure 17: Gaze-based orientation changes

Users widely preferred the gaze-based method. It caused less fatigue and users reported having greater control of the object's spin. This solution also created a great unintended side effect of emulating positional tracking. Mobile virtual reality does not yet support the user being tracked in space, but with this methodology, the act of rotating the neck felt like peeking around the object. We also found that the speed of the rotation could vary within a certain range. The user need not rotate the neck 90 degrees to see the entire side of an object. Rather, angles between 25 to 30 degrees were sufficient to create the immersive positional tracking effect and not rely on high angular precision to inspect an object without necessitating the user to stretch their necks.

b. Navigation

Navigation methods we observed were gaze-based steering, teleportation, terrain manipulation, and automatic movement with navigation objects.

Gaze-based steering was the preferred method of short range transportation by all subjects. Although it was the slowest method, users felt it gave them the most control and was the most immersive. For short range navigation, users felt that the teleportation method was disorienting, although the task was finished in a fastest time with this method. Navigation objects were sufficient to navigate the course and utilizing the object's user interface was

intuitive. However, users felt restricted and that they could not explore the environment as much as they wanted.

To allow for direction selection in 360 degrees around a user who is constrained to a typical non-spinning chair, we used the “left” and “right” buttons on the control pad to spin the user while they are being pressed. Some users were uncomfortable with the spinning mechanisms. To alleviate motion sickness for prolonged spinning, we tested placing artificial reference objects in front of the user and darkening the background slightly during a spin. This had minimal effect. Reducing spinning speed to around 10 rotations a minute helped alleviate motion sickness in all cases while being fast enough to not cause frustration. Prolonged spins (>180 degrees) continued to be uncomfortable for some users. For reference, spinning that was comfortable for most users takes three seconds to rotate 180 degrees.

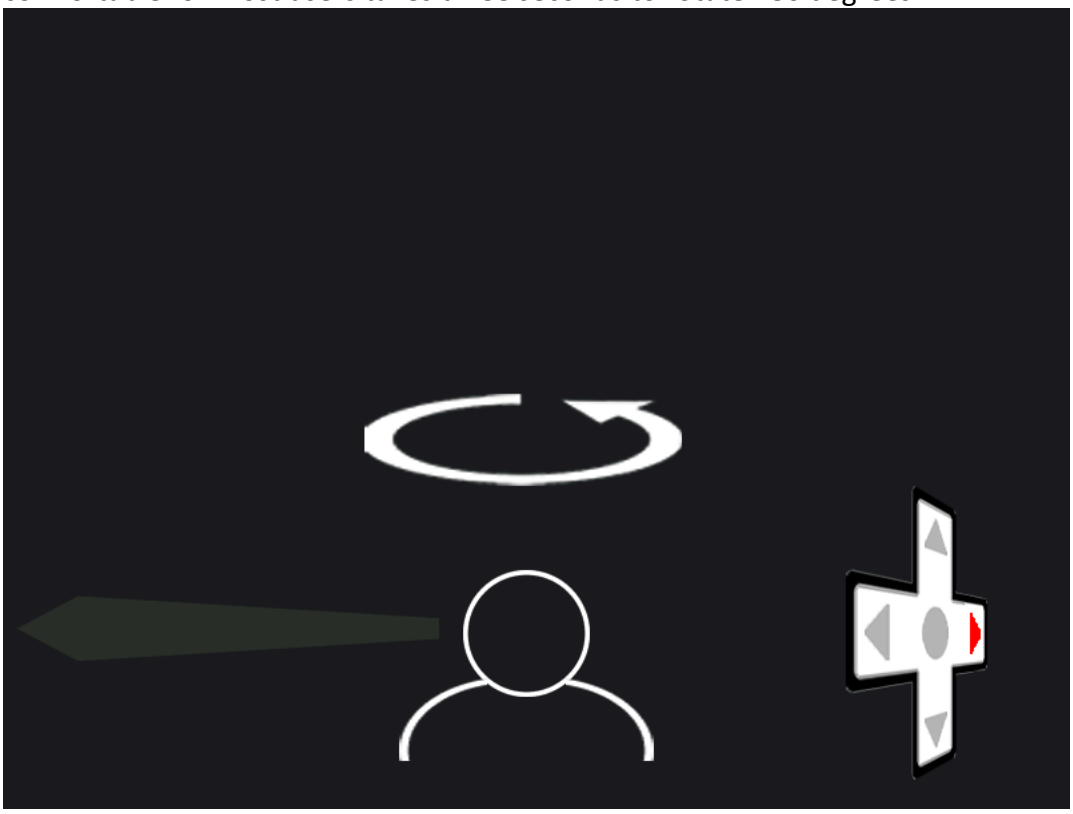

Figure 18: Using the touchpad to spin to the user’s left

We chose to use constant velocity when moving in virtual reality (except when spinning, as technically speaking, spinning is a change in velocity). We briefly tested, on ourselves, whether creating a slight ambulating head bob would add to the immersion of navigating in virtual reality. This was extremely uncomfortable and was quickly discounted in our methods. Since we eliminated the ability to change velocity we tested various speeds to have our users use in virtual reality. We found that about 1 meter per second moving forward and .5 meters per second moving backward were comfortable and immersive in small and large areas.

Input conditions defined how a user was able to start and stop moving. In the case of gaze-based navigation, we tested two options. In the first option, the user could press “up” on the navigation path to start moving forward, and then tap anywhere on the control pad to stop navigation. The other case had the user continuously hold the “up” button while moving, then release to stop. We thought arm fatigue might be a detrimental factor for the second case, but in observing users, all kept their arms raised in both cases during the navigation. The second case provided much more control than the first, so it was mostly preferred among users. We created the option to move backward by pressing the “down” button, although this was very rarely used in our tests

We then tested the same techniques, as well as the additional terrain manipulation technique, for long distance travel. Users found the gaze-based system useful for exploration, although the task took the longest and became fatiguing for users to use for long distances. Conversely, navigation objects allowed the user freedom to relax while they navigated a long distance automatically, but was not exploratory.

Teleportation was again not preferred. In the case of long distances, users could not get a straight shot to teleport to the final destination in one teleport, and needed to perform multiple teleports. After each teleportation, users took a few seconds to reorient themselves and a few times would get completely lost after a teleportation. To alleviate the jarring nature of teleportation, we tested a “fade-to-black” technique where users could be given 2 seconds after the teleportation selection to orient where they will be going, while the screen fades to blackness, then reappears at the desired destination. This was universally met with discomfort, even after experiencing the effect multiple times.

Terrain manipulation was not very intuitive for users and some found it to be uncomfortable. The few who had immediate command over the mechanism could finish the long distance tasks in faster times than the three previous methods. After repeating the mechanism, most users became more comfortable with it and felt that was a good combination of exploratory as well as fast, although only useful in long distance travel.

Typing with the custom glove based input device had varying results. Participants who identified as touch typists had a much easier time with the system. These participants typed at a range of 20 – 36 words per minute after training (n=4). Those who were not touch typists ranged from 5 - 30 words per minute (n=11). Visual cues helped typing speed as well as lowered the error rate. Compared to gaze based typing, it was faster but more prone to error (gaze based typing averaged at 12 words per minute among all participants [n = 15]). One of the main pain points was dealing with an error. All participants had trouble remember which key was the “delete” touch point and which key was the “swap word” touch point. Visual cues did not help this. Another difficulty was dealing with large words. Because the intended word cannot be determined until a user is finished, there is no proper way to show word progress in the middle of typing a word. Because of this, users reported forgetting which letter they had just typed. Users express frustration when given no feedback on how they may have misspelled a word when they did not enter the word correctly.

VI. Conclusion

With user testing and an iterative design process, we established a usable interaction methodology for interacting in a virtual reality environment using limited inputs. We divided tasks a user would need to perform in a general case as being object manipulation and navigation. To select an object in space, users use a gaze-based method focusing their head in the direction of the selectable object, and pressing the “select” button. This will reveal a radially oriented user interface for users to perform miscellaneous tasks with the object.

In order to transform the object, users need to select the object by holding the “select” button. At this point, the object will hover in front of the user’s field of view, attached to their gaze without needing to keep the “select” button held down (holding the “select” button again will exit this transformation mode). While in this mode, users can press the “left” and “right” buttons on the touch pad to move the object forward and backward, respectively. To scale the object, users can press the “up” and “down” buttons to scale up or down respectively. Moving the head during transformation mode will rotate the object in the opposite direction of the users head, creating a illusion of positional tracking.

In order to navigate short distances, users can press the “up” button and keep it held down during the duration of navigation, while steering with their gaze. Pressing the “down” button moves the user backward. Users move at 1 meter per second forward, and .5 meters per second backward. To move long distances, users can use a terrain manipulation method by selecting the ground directly below them, enter transformation mode, move the world away from them, look at a different part of the world, and move the world toward them.

Having navigation objects does not use any separate input space, and have been shown to be comfortable. These could be used in order to navigate to nearby terrain that is not accessible via gaze-based steering. We decided to eliminate the teleportation method after finding it to be uncomfortable for users. In thinking about the overall system, we also realized that an implementation of teleportation could have users miss an object selection and accidentally teleport to a distant location.

The typing device seemed to be too reliant on touch typing abilities to seriously consider it a standard of typing in immersive environments. Although it proved to be better in terms of typing speed in than previous iterations of data gloves, there was too much variance in participant’s abilities to use the device. Perhaps this could be something useful for information workers or people who are used to touch typing. In the future, it would be interesting to see how quickly people could learn this device over a longer period of time.

Virtual reality is unique in its divergent set of input devices, and may be one of is strengths as platform. That being said, some amount of user expectations play a large role in the usability of technologies that we use every day. Standard user interface elements or interaction methodologies are crucial to a seamless user experience. This paper represents a step toward a standardization of the most widely accessible form of the newest computing platform.

VII. Works Cited

1. Hoffman, D. M., Girshick, A. R., Akeley, K., & Banks, M. S. (2008). Vergence–accommodation conflicts hinder visual performance and cause visual fatigue. Journal of Vision, 8(3), 33.1–3330. http://doi.org/10.1167/8.3.33 http://www.ncbi.nlm.nih.gov/pmc/articles/PMC2879326/

2. Chu, A. (2014) VR Design: Transitioning from a 2D to a 3D Design Paradigm. http://alexchu.net/Presentation-VR-Design-Transitioning-from-a-2D-to-a-3D-Design-Paradigm

3. Oculus. User Interface Documentation for Developers. https://developer.oculus.com/documentation/intro-vr/latest/concepts/bp_app_ui/

4. Abrash, M. (2014) What VR Could, Should, and Almost Certainly Will be Within Two Years http://media.steampowered.com/apps/abrashblog/Abrash%20Dev%20Days%202014.pdf

5. Reason, J. T.; Brand, J. J. (1975). Motion sickness. London: Academic Press.

6. Bowman D., Koller D., Hodges L. (1997).“Travel in immersive virtual environments: an evaluation of viewpoint motion control techniques.” In Virtual Reality Annual International Symposium, 1997, pp. 45–52. IEEE Computer Society Press, 1997.

7. Poupyrev I., Weghorst S., Billinghurst M., and Ichikawa T. (1997) “A framework and testbed for studying manipulation techniques for immersive VR.” In Proceedings of the ACM Symposium on Virtual Reality Software and Technology - VRST ’97, pp. 21–28. New York, New York, USA: ACM Press, 1997.

8. Bowman D., Kruijff E., LaViola J., and Poupyrev I. (2004). 3D User Interfaces Theory and Practice Addison Wesley, Boston, USA

9. Foley J., Wallace V., and Chan P. (1984). “The human factors of computer graphics interaction techniques.” Computer Graphics and Applications, IEEE 4:11, 13–48.

10. Zhai S, Buxton W. and Milgram P. (1994). “The “Silk Cursor”: investigating transparency for 3D target acquisition.” In CHI ’94: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, pp. 459–464. New York, New York, USA: ACM, 1994.

11. Grossman T. and Balakrishnan R. (2005).“The bubble cursor.” In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems - CHI ’05, p. 281. New York, New York, USA: ACM Press.

12. Bolt R. (1980). ““Put-that-there”: Voice and gesture at the graphics interface.” In SIGGRAPH ’80: Proceedings of the 7th Annual Conference on Computer Graphics and Interactive Techniques, pp. 262–270. New York, New York, USA: ACM.

13. Poupyrev I., Ichikawa T., and Weghorst S. (1998). “Egocentric object manipulation in virtual environments: empirical evaluation of interaction techniques.” Computer Graphics Forum 17:, 41–52.

14. Cockburn A., Quinn P., Gutwin C., Ramos G., and Looser J. (2011).“Air pointing: Design and evaluation of spatial target acquisition with and without visual feedback.” International Journal of Human-Computer Studies 69:6.

15. Kopper R., Bacim F., and Bowman D. (2011).“Rapid and accurate 3D selection by progressive refinement.” 3D User Interfaces (3DUI), 2011 IEEE Symposium on, pp. 67–74.

16. Pierce J., Forsberg A., Conway M., Hong S., Zeleznik R., and Mine M. (1997).“Image plane interaction techniques in 3D immersive environments.” In I3D ’97: Proceedings of the 1997 Symposium on Interactive 3D Graphics, pp. 39–ff. New York, New York, USA: ACM, 1997.

17. Knight J. (1987). “Manual control and tracking.” In Handbook of Human Factors, edited by Gavriel Salvendy. John Wiley & Sons, 1987.

18. Mine M., Brooks F. and Sequin C. (1997). “Moving objects in space: exploiting proprioception in virtual-environment interaction.” In SIGGRAPH ’97: Proceedings of the 24th Annual Conference on Computer Graphics and Interactive Techniques, pp. 19–26. New York, New York, USA: ACM Press/Addison- Wesley Publishing Co., 1997.

19. Igarashi T., Kadobayashi R., Mase K., and Tanaka H. (1998).“Path drawing for 3D walkthrough.” In UIST ’98: Proceedings of the 11th Annual ACM Symposium on User Interface Software and Technology, pp. 173–174. New York, New York, USA: ACM, 1998.Cohen 99

20. Zeleznik R., LaViola J., Feliz D., and Keefe D. (2002) “Pop through button devices for VE navigation and interaction.” In Virtual Reality, 2002. Proceedings. IEEE, pp. 127–134. IEEE Comput. Soc, 2002.

21. Schneiderman, B (2000) The Limits of Speech Recognition Communications Of The ACM, 43(9) 63-65 Electronic Copy

22. Ward, D , Blackwell, A , and D , M (2000) Dasher - a Data Entry Interface Using Continuous Gestures and Language Models In UIST 2000 The 13th Annual ACM Symposium on User Interface Software and Technology, pages 129-137, San Diego, CA, USA Electronic Copy

23. Williamson, J and Murray-Smith, R (2003) Dynamics and Probabilistic Text Entry Technical report, TR-2003-147 Dept Computing Science, University of Glasgow Electronic Copy

24. Bowman, D , Ly, V , and Campbell, C (2001) Pmch Keyboard Natural Text Input for Immersive Virtual Environments Technical report, Virginia Tech Dept of Computer Science

25. Evans, F , Skiena, S , and Varshney, A (1999) VType Entering Text m a Virtual World submitted to International Journal of Human-Computer Studies

26. Kato K., Miyashita H., (2014). “Extension Sticker: A Method for Transferring External Touch Input Using a Striped Pattern Sticker”. UIST ’14 October 5–8, 2014, Honolulu, HI, USA

27. Johnny Accot and Shumin Zhai (1997). Beyond Fitts' law: models for trajectory-based HCI tasks. Proceedings of ACM CHI 1997 Conference on Human Factors in Computing Systems, pp. 295-302. http://doi.acm.org/10.1145/258549.258760