PinchMove: Improved Accuracy of User Mobility for Near-Field Navigation in Virtual Environments

Yun Suen Pai
Keio University Graduate School of Media Design
Yokohama, Japan
yspai1412@gmail.com

Zikun Chen
Keio University Graduate School of Media Design
Yokohama, Japan
zkzk.chen@gmail.com

Liwei Chan
National Chiao Tung University
Hsinchu, Taiwan
liweichan@cs.nctu.edu.tw

Megumi Isogai
NTT Media Intelligence Laboratories
Tokyo, Japan
isogai.megumi@lab.ntt.co.jp

Hideaki Kimata
NTT Media Intelligence Laboratories
Tokyo, Japan
kimata.hideaki@lab.ntt.co.jp

Kai Kunze
Keio University Graduate School of Media Design
Yokohama, Japan
kai.kunze@gmail.com

ABSTRACT
Navigation and mobility mechanics for virtual environments aim to be realistic or fun, but rarely prioritize the accuracy of movement. We propose PinchMove, a highly accurate navigation mechanic utilizing pinch gestures and manipulation of the viewport for confined environments that prefers accurate movement. We ran a pilot study to first determine the degree of simulator sickness caused by this mechanic, and a comprehensive user study to evaluate its accuracy in a virtual environment. We found that utilizing an 80°tunneling effect at a maximum speed of 15.18°per second was deemed suitable for PinchMove in reducing motion sickness. We also found our system to be at average, more accurate in enclosed virtual environments when compared to conventional methods. This paper makes the following three contributions: 1) We propose a navigation solution in near-field virtual environments for accurate movement, 2)we determined the appropriate tunneling effect for our method to minimize motion sickness, and 3) We validated our proposed solution by comparing it with conventional navigation solutions in terms of accuracy of movement. We also propose several use- case scenarios where accuracy in movement is desirable and further discuss the effectiveness of PinchMove.

ACM Classification Keywords
H.5.m. Information Interfaces and Presentation (e.g. HCI): Miscellaneous; See http://acm.org/about/class/1998/ for the full list of ACM classifiers. This section is required.

MobileHCI '18, September 3-6, 2018, Barcelona, Spain.

ACM ISBN 978-1-4503-5898-9/18/09$15.00
https://doi.org/10.1145/3229434.3229470

Figure 1. PinchMove being used bimanually, where the user's translation and axis of rotation are about the hands to achieve accurate navigation virtually.

Author Keywords
Virtual environment; accurate navigation; pinch gesture; simulator sickness; near-field

INTRODUCTION
In today's virtual (VR) and augmented reality (AR) systems, it is nearly impossible to create a perfect navigation mechanic suited for all types of applications and scenarios. Large virtual environments may benefit from more instantaneous and quick navigation, whereas small confined spaces may prefer to just use actual physical walking, depending on the hardware support. However, for spaces that are larger than that, yet still confined and requires accuracy, a suitable navigation mechanic is still unknown. The currently preferred method for VR, teleportation, works well in mitigating simulator sickness and navigating the user in large virtual environments. However, it is not possible to perform accurate movements; which is acceptable since it was never designed to achieve that in the first place. On the other hand, physically walking is arguably still the best method, but requires a physical area as big as its virtual counterpart, equipped with precise trackers. Therefore, what is the best form of navigation for a room-spaced environ-

ment, like an office, workshop, or lab, where the user requires accurate control of his or her position for professional use or simulation?

PinchMove aims to address this by using a pinch gesture to grab a position in space and drag the user based on the change in position of the user's hand. This form of manipulation of the viewpoint is accurate and allows the user to move in four directions freely, as well as rotate depending on the angle of a pinch. It is based on a discrete style of navigation as opposed to controlling the rate of movement (joystick controls). Furthermore, it is also suitable for users who are in a confined physical space, yet still, require accurate control of his or her location in the virtual world since only arm movements are required.

However, the primary concern for any form of navigation is the degree of simulator sickness it may cause. In our pilot study, we performed a comparative study between three tunneling speeds; slow, medium and fast, which are based on the smoothing time of the visual effect. We found that a medium tunneling speed with a maximum of 15.18 degrees per second was suitable for PinchMove, since it causes the least amount of nausea, oculomotor and disorientation. In our main user study, we aim to determine the degree of accuracy our system can achieve compared to more conventional methods. We compared between conventional gamepad input, unimanual pinch navigation, and bimanual pinch navigation based on three levels of preset accuracy, 6 positions, and 3 orientations. We found that on average, bimanual navigation performed the fastest for all three levels of accuracy.

We further discussed other contributing factors that may influence the proposed solution, such as optimal velocity, different forms of gestures, as well as utilizing different hardware to achieve a similar result.

The following are the contributions of this work:

1. We developed a highly accurate navigation mechanic for virtual environments that are suited for near-field navigation.

2. We ran a user study to determine the best tunneling parameters in reducing simulator sickness with the proposed method.

3. We evaluated its accuracy compared to conventional navigation solutions in virtual environments and found that bimanual input achieved the highest speed in achieving the desired accuracy (6.849, $p = 0.004 < 0.005$).

RELATED WORK

Among the numerous available options for navigating in a virtual environment, most of them cater to either avoiding simulator sickness, being fast, being realistic, or all three at once. These are important factors to consider, yet at the end of the day, the best navigation method is highly dependent on the application itself. Furthermore, an additional dimension that we wish to consider in this work is the accuracy of navigation or the ability given to the user to navigate to the desired point, given moderately limited space.

Navigation in Virtual Environments

Mobility in virtual environments is one of the most important input, and arguably the most popular navigation mechanic for VR today is teleportation, which causes an instant change in position using a fade and blink animation [2]. A variation to this is Dash, where instead of teleporting, the user moves at high speed to the designated target location to provide a better sense of movement. This variation in speed and transition for teleportation could mean that a proper tweak in speed and transition can improve the experience [16]. However, these methods of navigation do not allow fine-tuning of position. For that, the user needs to walk to the desired position after teleporting, adding to additional navigation time.

Other more immersive forms of navigation also exist, such as those that require movements of legs and arms, akin to real walking [30, 21]. However, the main issue with such methods is that even though they take less physical space than actually walking, they require the user to stand and perform gestures that can both be tiring and imprecise. Accuracy is further sacrificed due to under and overshooting of position when the user stops since the system needs to detect stopping.

One of the currently available solutions most related to our proposed method is the grabbing locomotion, where the user grabs the space in front of them to traverse. This can often be seen in climbing VR games like Climbey[15] and The Climb[5]. However, these games were made specifically for vertical navigation only and do not provide other means of navigation such as rotation.

Based on the related work, the factors to consider in an implemented navigation mechanic usually boils down to simulator sickness, quickness, and sense of realism or immersion. However, one other factor less mentioned is the consideration for accuracy, or how much degree of control a user can be given to reposition himself or herself as accurately as possible.

Simulator Sickness

Simulator sickness, cybersickness or motion sickness, are terms used particularly in VR environments to mean nausea or dizziness, often caused by the 'sensory conflict theory' that refers to how our vision is augmented to receive motion signals, yet our non-vestibular proprioceptive senses don't, resulting in a variance that causes said sickness [24, 10]. To help curb this issue, there has been an established guideline for VR development, such as how movement acceleration should be linear and not too long, avoid rotation on the forward-axis, avoid yaw-axis rotation, and avoid direct control of the main camera view [17]. Among the current solutions are adding a motion platform, performing direct vestibular stimulation, and implementing rest or static frames, of which only the last option can be considered to avoid any third party peripherals or custom sensors [14].

Furthermore, over the years, VR researchers have begun implementing several visual tricks, one of them being 'grounding', where part of the foreground remains static [1]. Essentially, it reduces the immersion level of the user through methods like reducing the field-of-view (FOV) [9], or adding a static visual frame for the user. This can greatly reduce the effects

of visual update delays and simulator sickness compared to a wider FOV [6]. Finding an optimum FOV can be tricky because reducing it can greatly effect the immersion and requires the user to perform a bigger head and eye movement to view content at the periphery [31].

However, it was also mentioned that a smaller FOV is more acceptable for the smaller virtual environment compared to larger ones. Fernandes et al. found that a FOV of 80°was the limit of an acceptable FOV before it started to detract the experience [9]. Among some of the other methods explored by researchers are using an independent visual background (IVB) which is similar to Google Daydream's implementation of a virtual meta world, though it was only proven to work in driving simulations at this point of time [8].

Pinch Gesture Devices and Inputs

Glove-based devices like the Z-Glove and DataGlove [33] that uses ultrasonic positioning and magnetic positioning respectively were developed in the past to allow finger-based gesture recognition. Even though these devices allow whole hand interaction [26], pinching between 2 fingers is the most minimal form of performing a gesture that provides proper haptic feedback, when compared to other gestures like grasping or grabbing, or bimanual gestures like clapping.

A two-handed navigation system was developed using the PinchGlove where the relative vectors between the hands allows the user to determine the direction and pinching allows navigation, whereas a different pinch gesture allows velocity control [3]. Even though the technique was deemed flexible, no proper studies were conducted, and since it controls the rate of movement as opposed to discrete positioning, it suffered from accuracy. Furthermore, relying on various pinch gestures using multiple fingers may provide additional functions, yet at the same time make it less intuitive if the user needs to memorize each function. FingARtips focuses on the pinch between index and thumb for grabbing, pointing, and pressing in an AR environment [4]. However, this work was meant for urban design, and not physical movement of the user avatar which presents new set of challenges like movement accuracy and simulator sickness.

Another previous work used pinch gestures to manipulate computer-aided design (CAD) models using bimanual pinch gestures [25]. GaFinC combined gaze with pinch gestures for selection and manipulation, with different pinch movements resulting in translation and rotation of the 3D model. However, merely adopting object manipulation mechanics like GaFinC into navigation presents an issue; object manipulation only changes the position and orientation of an object with reference to its center point as pivot, whereas for navigation, we need to consider both the user's position and orientation, as well as the environmental cues for it to be natural and accurate. If GaFinC was directly used for navigation, the desired final position with respect to the environment cannot be achieved, which is properly explained in the Unimanual Navigation subsection with reference to Figure 2 that shows the difference between PinchMove and conventional methods (more similar to GaFinC).

Unlike the other previously mentioned related works, PinchMove is a software-based approach that uses the fine positioning of the user's hand for the user's self positioning. It relies on intermittent hand translational movements and subtle orbital movement for rotation that takes into account the space and the environment itself. It also uses very minimal to no pinch-gesture memorization (depending on unimanual or bimanual). Even though the pinch gesture itself can be substituted with other forms of gesture or even a button input, we were motivated based on the mentioned related work that pinching is the simplest input gesture for inducing self haptic feedback on the hand, making it possible to be used for gesture-only solutions.

IMPLEMENTATION

Our system was implemented using the Oculus Rift CV1 with the Touch controllers. The Touch controllers are 6-DOF input controllers for each hands, with each buttons equipped with a capacitive sensor. However, any device with 6-DOF hand tracking can be used, such as future peripherals for ARCore and ARKit on mobile devices. For our implementation, the user simply needs to rest the thumb on any of the face buttons or the analog stick (since the capacitive sensors can sense this without actually pressing any button), while pressing the index trigger. This creates a pinch animation on the virtual hands as well. We divided the implementation based on two methods; unimanual and bimanual, depending on the user's preference.

The pinch gesture works by "pinching" the viewport or any point in space and dragging it towards yourself, creating an inverted movement that causes the user to navigate to the direction the controller was previously. This inverted movement is metaphorically referred to as Camera-in-hand or eyeball in hand, which was deemed as a superior navigation mechanic [28, 32]. A similar example would be grabbing a rope and pulling it while sitting on a wheeled-office chair. Though not the most popular navigation method for VR, a variation of this method can be seen in VR games like Climbey [15] and The Climb [5], because it emulates the arm movements of actual wall climbing. Grabbing a point and lifting up moves the user upwards, akin to PinchMove. However, the games above differentiate in two factors; they allow only vertical movement, and rotation is not possible.

This particular form of navigation can potentially feel more natural than using a gamepad because the distance traveled and velocity of the arm gestures are directly reflected on the user for accurate positioning. This is also relatable to how we can move forward when grabbing and pulling an object of higher mass than us, or movements like crawling or climbing. For example, to be really accurate, we won't just say "I want to stand in front of a table", we say "I want to stand 10cm in front of a table". Therefore, PinchMove leverages this by allowing the user to place their hands on these environment references and navigate based on them. For our implementation, we choose to disable vertical movement or movement about the y-axis since we are focusing on ground-based navigation for this study, though it can easily be added depending on the application. Another feature that we chose to exclude for the current implementation is speed control, thus catering

PinchMove for near-field environments. This is because we wish to analyze it as its most basic form (one-to-one translation and rotation) before adding additional features that may impact its accuracy.

To allow the user to move forward by pinching the space, we first find the initial position of the controller the frame the trigger is pressed and final position after the controller moves to a new position. This difference in Euclidean space is added to the user's current position, allowing them to move front, back, left and right depending on the controller's movement vector. Once the user has decided on the final position and released the trigger, movement is halted and their final position becomes the new current position. Therefore, if a user places their hand on a virtual object and pinches themselves towards it, they will be standing directly in front of the object based on the desired distance. Essentially, it is an implementation of rotate, scale, and translate (RST) manipulation without scaling for the user in fully immersive virtual space [13]. However, mirroring RST mechanics for rotation by rotating the user about their upward axis will result in a position that is undesirable if the user wishes to move and face to a specific virtual object. Therefore, the key innovation is in how the rotation is handled, which will be detailed further in the subsections below.

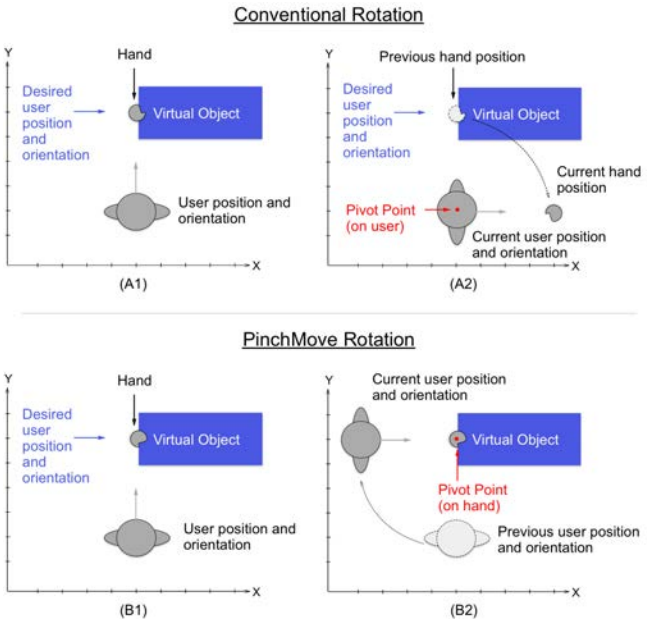

Figure 2. An example of 90°rotation behavior where (A1) shows before and (A2) shows after for conventional rotation with pivot point about the user, whereas (B1) shows before and (B2) shows after PinchMove rotation with pivot point about the hand.

Unimanual Navigation

Unimanual navigation ensures that the user can fully control his or her position in virtual space using a single Touch controller. This leaves the second hand to be free for other tasks. However, care needs to be taken when mapping functions for translation and rotation to avoid unintended navigation. To emulate pinching, the index finger trigger allows the user to perform pinch translation, whereas the middle finger trigger allows the user to rotate about the y-axis. Initially, we experimented with using a single trigger button for both translation and rotation, but this resulted in an inaccurate movement where the user keeps performing minor rotations during translation that was rather nauseous. The use of separate buttons for the different function was to avoid any accidental activation. We argue that users who use unimanual navigation are most likely engaged in other activities at the same time, thus may require navigation controls that are simpler and with minimal memorization. The center of rotation is assigned to the controller held in hand instead of the user, where the user orbits around the controller during rotation. This is the key difference compared to other works like RST [13] with central pivot rotation; it is mathematically impossible to our knowledge to achieve this result using other pivotal points. We illustrate this in Figure 2. In this figure, the user desires to stand directly in front of the table, with the table being within arms reach (Figure 2(A1) and (B1)). The user then rests his/her hand on the table's edge as a reference and performs the rotation gestures. The two main differences between Figure 2(A2) and 2(B2) is the pivot point location and direction of rotation of the hand (not the user).

Figure 2(A2) shows the conventional method adapted directly from the RST method with the pivot point being the center of the user and the rotation gesture being rotating the wrist to the right to turn right. The resulting final orientation would be the same with PinchMove, but the position of the user would be undesirable. This is because the user simply rotates about his/her center point, causing him/her to turn right (the arm positions in space changes since it is always in front of the user) but not reposition correctly at the same time. This method of rotation is adopted in RST, which is perfectly suitable for object manipulation.

However, in navigation where the user wishes to reach the desired position, he/she needs to orbit around the point of rotation using the opposite gesture direction (inverted direction as previously mentioned, since the viewport is being manipulated instead) that is outside of the user. PinchMove leverages the user's hand position or controller to achieve this as shown in Figure 2(B2), making the hand position in space static but the user's body position changes with reference to the hand's rotation. The resulting orbital movement means that rotation has combined with translation, similar to Hovercam [12] and Shocam [19] which were intended for desktop use. The user can actually choose to only translate, or orbit (translate and rotate), to a desired position. Orbital-like movements in VR has been explored before and has been used as a primary navigation mechanic [20], though this work relies on head rotation and eye tracking. Furthermore, our applied orbital movement mechanic is more subtle since the controller is never too far away from the user. This subtleness provides the illusion that the user is rotating conventionally, i.e. about themselves, but with a higher degree of accuracy of self-placement in the virtual space. A large orbital movement resulting from further pivot points will be disorienting unless made for a specific purpose like AnyOrbit [20]. The angle of rotation is based on the hand controller orientation about the y-axis. When

the controller is rotated, it is as though the viewport rotates along with it, though in actuality the user has successfully rotated about the controller. Therefore, if the user performs this movement when placing the controller on a virtual object as shown in Figure 2 facing another direction, they will be able to reposition themselves in front of it at any desired orientation. The resulting implementation makes translation and rotation feel extremely natural.

Bimanual Navigation

Bimanual navigation allows the user to navigate either with the left controller, right controller, or both at the same time. Either left or right translation would behave no different from unimanual navigation. However, for bimanual, we take the average position of both controllers as a reference for movement. If one controller travels further than the other, then rotation is triggered, where the user rotates about the reference object. Therefore, if the user performs this movement when placing both controllers on a virtual object, they will be able to reposition themselves accurately based on the movements of both of their hands.

The implemented bimanual controls are for users to use both hands as reference positions during navigation. We argue that users of bimanual navigation wish to focus purely on accurate navigation. Therefore we choose to mirror the controls for both hands and use the same button input for both translation and rotation to minimize memorization of functions. Furthermore, assigning rotation to another trigger button is excessive, since the user already has access to a set of index trigger buttons on both hands anyway. Naturally, enabling PinchMove for both hands removes the ability of the user to use a free hand for another task. However, we wish to evaluate the performance comparison of these two methods and determine the users' preference with regards to accuracy.

Tunneling

To ensure that simulator sickness is minimized, we employ the tunneling method for PinchMove. Tunneling is enabled through the vignetting and chromatic aberration visual effect attached to the main camera in the scene. A vignetting intensity of 0 creates a maximum FOV for the user, in this case, 110°which is the FOV of the Oculus CV1 headset. When the vignetting intensity is increased to 1, the screen is completely blacked out, equaling to a FOV of 0°. The two important tunneling parameters are therefore the intensity as well as the rate of which the tunneling effect appears, as discussed by Fernandes et, al[9].

In this work, it was determined that an 80°FOV was the minimum allowable FOV before it starts to detract from the experience. Even though the rate of tunneling was also determined, movement in PinchMove is largely different than regular gamepad input where the rate of movement is controlled, whereas PinchMove is more discrete. Therefore, a pilot study was designed to determine the appropriate maximum speed for tunneling and will be further explained in the following section.

Figure 3. Environment for the pilot study. The left image is the top view with all waypoints position shown, whereas the right image shows the participant's view during the study.

USER STUDY

The user study is divided into a pilot study that focuses on determining the right parameter for tunneling to minimize motion sickness and the main study that focuses on evaluating the accuracy of positioning and orientation. We chose to use the Oculus Rift CV1 with the Oculus Touch controllers due to its more ergonomic and natural controller shape for pinching, though our implementation has been tested with the HTC Vive as well. The reason we employed the pilot study was to reduce the possibility of motion sickness as much as possible before conducting the main study, so that it does not significantly effect the accuracy of PinchMove. Furthermore, we also believe that understanding motion sickness is always important for studies on navigation in virtual environments.

Pilot Study

For this study, the goal is to determine if the proposed navigation method causes a significant amount of simulator sickness depending on the speed of the tunneling effect. Although there exist several methods to mitigate sickness in VR, we chose the conventional tunneling effect of limiting the FOV of the user as it has been proven time and again to be effective [9]. Fernandes et, al. found that in a pilot study, a FOV of 90°is deemed to be the preferred minimum FOV whereas 80°is the largest FOV to detract from the experience.

Therefore, we choose to use a FOV of 80°so that it would not further detract the experience. Furthermore, a subtle and slow decrease in the FOV restrictor even causes some of the participants to not notice the change in FOV. However, a major difference between that study and ours is that the study was designed for the participants to navigate a huge virtual space, whereas PinchMove was designed for a more confined area. Secondly, pinching the viewport results in a more discrete style of navigation as opposed to regular navigation that is continuous and works by controlling the movement rate. The tunneling effect only appears when the user is moving in the environment and slowly dissipates when the user is static. Therefore, the tunneling effect would appear at a slower rate as opposed to the conventional method, since releasing the pinch gesture dissipates the tunneling.

We recruited a total of 8 participants for the pilot study. To determine the appropriate tunneling rate and sickness, we designed an office-like environment where the user is required to navigate using PinchMove for two minutes. We also predefined three levels of tunneling speed, slow (maximum of 10.15 degree per second), medium (maximum 15.18 degree per second) and fast (maximum of 30.08 degrees per second)

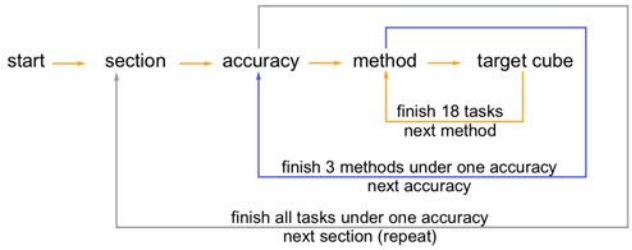

Figure 4. Flow Chart for the designed user study

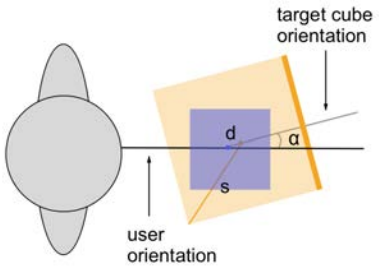

Figure 5. Parameters in defining the accuracy of translation and rotation. α is the angle between the user cube and target cube, d is the distance between the center points of the user cube and target cube, whereas s is the distance between the center point of the target cube with its corner

by altering the tunneling smoothing time from 110°to 80°. To reduce ordering effect, we employed a Latin Square order for counterbalancing (though since we have 8 participants with 3 conditions, the last order was eliminated, but we nevertheless deem it acceptable for a preliminary study).At the beginning and end of each session, each participant is required to answer the Simulator Sickness Questionnaire (SSQ) to estimate current nausea, oculomotor, disorientation, total sickness score, and total rise in sickness that is being felt [11].

During the study, the participant is required to navigate through white capsule waypoints that appear one at a time in a confined office space which is placed in a way that forces them to translate and rotate the viewpoint, similar to the experiment by Fernandes et, al. as shown in Figure 3. At the end of each session, each participant is also required to answer another set of questionnaires designed by Suma et, al. to determine if the change in FOV was noticeable or not [27]. Participants were asked to rate the following questions from a scale of 1 to 7, where 1 means "I did not notice anything" and 7 means "I obviously noticed it."

1. I saw the virtual environment get smaller or larger.
2. I saw the virtual environment flicker.
3. I saw the virtual environment get brighter or dimmer.
4. I saw that something in the virtual environment had changed size.
5. I felt like my field of view was changing in size.
6. I felt like I was getting bigger or smaller.
7. I saw that something in the virtual environment had changed size.

Experiment Design

To determine the accuracy of navigation, we designed a virtual environment for the participant to navigate using PinchMove that is inspired by the RST user study [13]. The flow of the study is illustrated in Figure 4. To determine the accuracy of movement, each participant is required to complete a task where the time required to complete each trial is recorded. The task requires the user to align a user object with a target object. The user object is a cube placed in front of the user that follows the user's translation and rotation (blue cube). The target cube is a cube placed in the virtual environment at a predefined position and orientation (semi-transparent cube with an orange face to depict the front of the cube), illustrated in Figure 6.

To determine the possible predefined locations, we consider the FOV (3 angles) and distance from the user (2 distances) to determine 6 possible positions. We use FOV to ensure that the target cube is always visible to the participant at the beginning of each trial, and we select distances according to the standard arm length for near-field interactions. There is a possibility of 3 orientations per position (-45°, 0°, and 45°). We also fix three accuracy levels for the user for each trial (98%, 96% and 94%) for the position and orientation, which can be seen on Figure 5. For the position accuracy, an accuracy of 100% means that the center point of the user object is exactly at the center point of the target object, or when $d = 0$. 0% accuracy is the furthest possible distance between the two center points when they are in in contact, which is also equivalent to the distance between the center point and the middle point of the edge of the target cube (when $d > s$). The predefined values of 98%, 96% and 94% were determined from initial testing; our test participants found that an accuracy level of 100% was nearly impossible, whereas it becomes relatively easy at 92%. Therefore, to keep even spacing between the accuracy levels, we decided to chose the aforementioned values.

For the orientation accuracy, an accuracy of 100% means that the angle between the user object and target object, α, is 0°. 0% accuracy is the largest possible angle deviation between the two objects. Since we measure the acute angle as the difference in orientation, 90°is deemed 0% for orientation accuracy (when $\alpha \geqslant 90$). Finally, each trial is repeated three times. The three input methods are described below:

- GamePad: A conventional navigation mechanic utilizing the thumbsticks on the gamepad to move and rotate. This input method serves as the standard baseline of comparison.
- Unimanual: In addition to navigation with the index trigger, the middle finger trigger is used to enable rotation. The participant will be seated on a leg chair for this scenario.
- Bimanual: Both controllers are required to rotate the participant by the relative angle between them, without any additional button. The participant will be seated on a leg chair for this scenario.

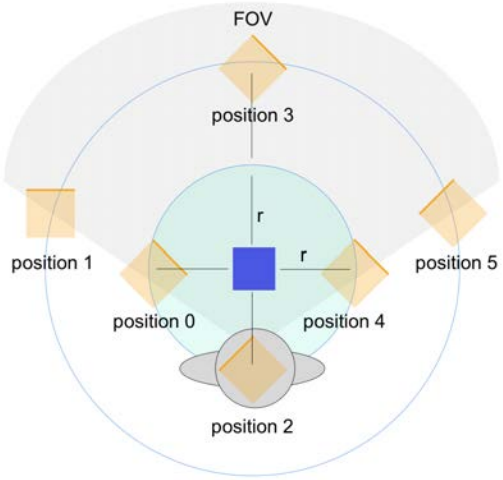

Figure 6. The positions of the target cube (only 1 can be seen at a given time) for the user study with 3 possible orientations.

Only one target cube will appear at a given time, and the next cube will only appear once the participant successfully completes that trial. Correct alignment of the cubes are detected automatically by the system. Once the participant completes the trial, there will be a one second delay before the next cube appears for them to know what the next difficult level and input method is for the upcoming trial. The total amount of trials per participant is 6(positions) x 3(orientation) x 3(inputs) x 3(accuracy) = 162 trials per participant. In this study, the independent variables are the input methods, accuracy, position and orientation of the cube. The dependent variable is the time required to complete the trial. Figure 7 shows the view of the participant during the user study.

Prior to the study, the participants are allowed to familiarize themselves with the navigation mechanic for 5 minutes. During this period, we also perform a quick calibration for each user to collect the speed of movement during PinchMove. We then modify the gamepad input speed to be equal to the average speed of PinchMove for each participant. This is because the speed of navigation for PinchMove purely depends on the speed of arm movement for each participant, whereas the speed of movement for the gamepad needs to be predetermined. Therefore, to ensure fair comparison, the gamepad movement speed is calibrated to be on average, equal to the pinchMove speed for each participant. We recruited a total of 19 participants (10 male, 9 female) aged between 22 to 34 years (mean: 25.63, SD: 2.81). At the end of the study, each participant finally answers a Likert-scale questionnaire from a rating of 1 to 5 for perceived accuracy, efficiency, reliability, learnability and likability for each navigation method [29].

RESULTS

For the pilot study, Figure 8 show that a medium tunneling speed overall caused the least rise in motion sickness with a score of 10.285 compared to a higher tunneling speed with a rising score of 17.3 and a slower tunneling speed with a rising score of 24.78. Looking at the average score for nausea, oculomotor, disorientation and overall total score, the medium

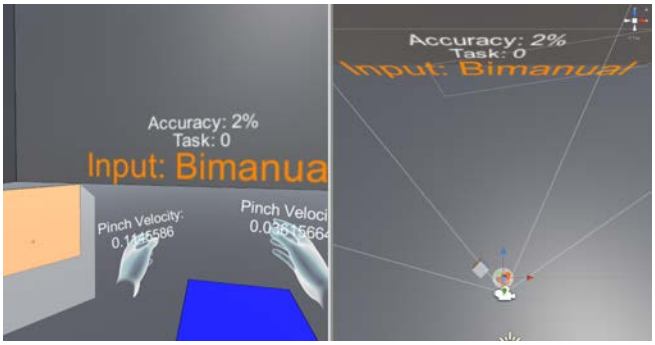

Figure 7. The participant's view of the study and a top-down view of the environment. The target cube can be seen at Position 0

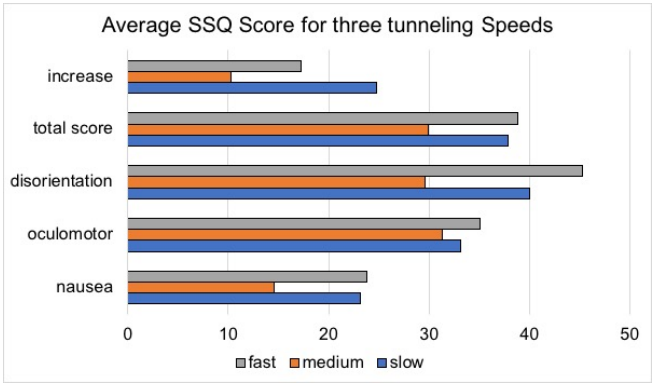

Figure 8. SSQ results by comparing between three tunneling speeds

tunneling speed scores overall causes less sickness as well. The data were analyzed using repeated-measures ANOVA. The repeated measures ANOVA reveals that there is no difference found on increase score between the treatments ($F_{2,14}$ = 1.57; p = 0.243). Interestingly, a faster tunneling speed scored lower than slow tunneling speed in all these categories except for the average increase of total sickness. For the noticibility questionnaire, the only relevant questions were the third and fifth questions which are related to the change in FOV. According to Figure 9, participants were much quicker to realize that the environment seemed darker at the fastest tunneling speed whereas the slow and medium speed were about equally slow to realize. However, most participants realize the change in FOV at the medium speed, followed by fast, and finally slow.

For the main user study, the results were analyzed using ANOVA repeated measures where we evaluated the accuracy of each input method based on completion time of the predefined accuracy. Figure 10 shows the estimated marginal means for the gamepad, unimanual, and bimanual interfaces for the accuracy of 94%, 96% and 98%. All post-hoc comparisons used Bonferroni corrected confidence intervals. There was a significant main effect of accuracy ($F_{2,36}$ = 7.250; p = 0.002 < 0.005) and pairwise tests show that users were significantly faster with an accuracy of 94% compared to 98% (6.326 vs. 8.329, p = 0.002 < 0.005). There was also a significant main effect of interface ($F_{2,36}$ = 3.41; p = 0.002 < 0.005). Pairwise test found that bimanual input was faster than unimanual input (6.849 vs. 8.183, p = 0.004 < 0.005). However, there was no

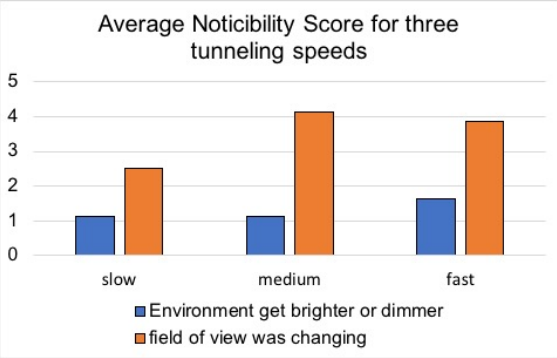

Figure 9. Noticibility Score for the three tunneling speeds

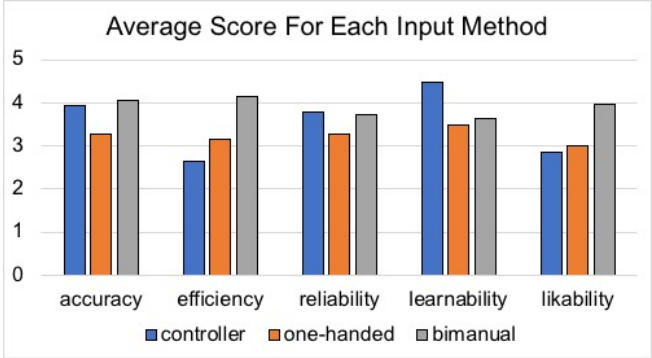

Figure 11. Qualitative results of three input methods

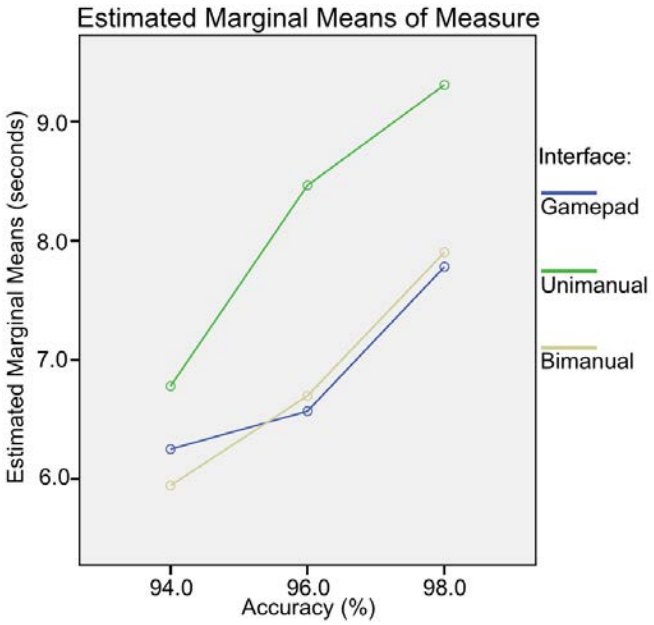

Figure 10. Estimated Marginal Mean of three input methods according to accuracy.

significant difference between gamepad and bimanual, as well as gamepad with unimanual.

For the final qualitative questionnaire, Figure 11 shows that bimanual was at average, generally perceived to be the most accurate, efficient, and likable. Gamepad input was slightly, but not significantly, higher than bimanual in terms of reliability, but is noticeably higher in terms of learnability. For perceived accuracy, bimanual was slightly, though not significantly higher than gamepad, with the unimanual method being the least accurate. Significant differences were found in accuracy, efficiency, likability, and learnablity, but not in reliability. For accuracy ($F_{2,36} = 5.16$; $p < 0.05$), significance exist between gamepad and unimanual ($p < 0.05$) as well as unimanual and bimanual ($p < 0.05$). For efficiency ($F_{2,36} = 9.38$; $p < 0.001$), significance can be seen between gamepad and bimanual ($p < 0.05$), and between unimanual and bimanual ($p < 0.05$). For likability ($F_{2,36} = 4.0$; $p < 0.05$), there is a significant difference between unimanual and bimanual ($p < 0.05$). Finally, for learnability ($F_{2,36} = 7.317$; $p < 0.005$), there is a significant difference between gamepad and bimanual ($p < 0.05$), and between unimanual and bimanual ($p < 0.05$). There is no significant difference for reliability ($F_{2,36} = 2.520$; $p = 0.095$).

DISCUSSION

Based on the quantitative results from the main study, it was found that at 98% accuracy, gamepad still performed slightly, though not significantly better, than both unimanual and bimanual, with unimanual being overall the least accurate method of navigation. as the accuracy requirement drops slowly to 96% and finally 94%, bimanual scores the fastest time for the completion of the task. In average for all three levels of accuracy, bimanual ends up as the best performer, followed by gamepad and finally unimanual. Furthermore, for our qualitative questionnaire, it can be seen that bimanual was significantly preferred in terms of efficiency and likability, and scoring almost the same for accuracy and reliability with gamepad. It only score significantly lesser for learnability due to it being a newly developed method.

For the main study, we asked the participants their general experience in VR, informally rank their preferred input methods from best to worst, overall comment on the user study, as well as suggestions for scenarios that can benefit PinchMove. Out of the 19 participants, 14 preferred PinchMove over conventional gamepad, and 12 from them preferred bimanual input over unimanual. The 5 participants who preferred gamepad mainly said that it was simply due to it being more conventional and common, leading to lesser time to master since the operation does not need to be understood. PinchMove also lead to fatigue of the arm since each user was required to keep up with the pinch and pull motion. Finally, another common feedback is that since PinchMove relies on pulling as opposed to gamepad which is more akin to pushing, the inverted controls presented some difficulties for these participants, similar to scrolling on a touchscreen where scrolling upwards with the finger causes the screen to scroll downwards instead.

The users who prefer PinchMove navigation presented some interesting related scenarios. One participant mentioned that, a gamepad was similar to sitting in a car and moving forwards, whereas PinchMove is more like grabbing the road and pulling it to move forward. Furthermore, PinchMove allows fine control of movement speed, where the user moves as fast as

their hand, as opposed to gamepad with a predefined maximum speed. Compared to the conventional gamepad, PinchMove was overall more enjoyable to use.

Bimanual was overall preferred because it provided a better spatial sense during movement, and using both hands for positioning overall feels easier and more intuitive. A common negative feedback for bimanual navigation is that since only the index trigger was used, participants who moved too fast caused accidental rotation, since they press the second controller trigger before releasing the first one for a split second. However, some participants also mentioned that since only a single trigger button was used for both controllers, no memorization was needed and it was easier to master. One of the participants who was experienced in VR, even used strategies for bimanual navigation for maximum speed, such as using a circular, pedaling motion between hands for quick movement, and orbiting one controller around another for quick rotation.

The 2 participants who preferred unimanual navigation agreed that it was overall more intuitive after the learning phase. However, it heavily depends on the scenario as well, since the one-hand option was provided so that the user is free to use another hand for other use anyway. They also mentioned that since translation and rotation was mapped to different buttons, no unintended movement can occur. For PinchMove, participants overall prefer bimanual over unimanual because they claim that unimanual rotation felt inverted, even though rotation for both methods actually rotate at the same direction.

In terms of motion sickness, more participants claim to have felt more motion sick using gamepad, followed by one-hand, and finally bimanual. A participant claimed that pulling navigation overall presented less motion sickness compared to pushing navigation. It is a common design rule in VR to avoid mapping rotation externally as this can cause heavy sickness, more so than translation [7, 18, 23]. Even though all input methods implement this, sickness was less for PinchMove since the rotation provided the participants with a sensation that they were rotating the world around them instead. Because of this, a participant even claimed that mastering PinchMove in return caused him or her to be more motion sick when using conventional gamepad.

Overall, we showed the feasibility of PinchMove as an accurate navigation mechanic for virtual environments, with its performance being overall slightly better than a regular gamepad. We find these results to be acceptable for our first implementation of PinchMove, since gamepad is an input method that has been available for a long time for use of navigation in virtual environments. However, we would argue for its superiority over a gamepad for several reasons. Firstly, PinchMove is a gesture-based input which is arguably more natural for interactions in AR/VR space as opposed to gamepad (based on the participants feedback on it feeling easier and intuitive with a better spatial sense). Secondly, we believe that with further fine tuning with feedback gathered from the participants in this study, it is highly possible for PinchMove's performance to perform much better than a gamepad (such as increasing learning time). It is worth mentioning though, that with the use of a gamepad versus a gesture based input like PinchMove, it is overall less tiring to use as mentioned by the participants, which is why for this point of time, we specifically state that PinchMove is limited to near-field navigation so users do not need to navigate too far or for a long period of time.

SUGGESTED SCENARIOS

As previously mentioned, PinchMove was designed for specific scenarios that are near-field and prioritizes accuracy. This section will discuss some of the possible application scenarios where PinchMove may prove beneficial, given the context, as well as some of the scenarios suggested by the participants.

For a more general use, we look at PinchMove as a possible direction for a definitive form of navigation for AR/VR. Seeing as motion controllers have become the definitive input mechanic for AR/VR. Teleportation has also become a method for far navigation that is an easy and motion-sickness free method of navigation. However, for fine tuning position, we need to physically walk towards where we desire. If the user lacks the necessary space or is seated, they would need to stand, or reposition themselves with the teleportation mechanic. PinchMove can possibly be the solution for this fine-tuning of position.

For accuracy-based applications, we looked into professional scenarios where AR/VR is used for simulation, training, or a tool. A possible application is in architecture and product design. In these fields, careful dimensioning of buildings and objects is vital. Since PinchMove navigates based on movement distance of the arms, this allows users to take advantage of proprioceptive sensing for precise near-field locomotion. Therefore, we believe PinchMove provides a accurate measurement-based locomotion that can be coupled with the work in architecture, computer-aided design modeling, and sculpting.

The previously suggested applications are catered towards 3D modeling, though we also see PinchMove as a useful tool for 2D interface design in VR as well. For example, user interface (UI) designers for mobile application or web pages can now possibly utilize VR to produce accurate design and navigate between several screens, as well as understand the spaces between each of their designs through navigation.

Another very relatable scenario is using PinchMove as a accurate navigating for simulation of space astronauts. Although several space exploration VR games utilize grab-and-pull mechanics to simulate zero gravity navigation [22], they don't allow the player to pivot around a grabbed object like PinchMove, nor provide the option for unimanual or bimanual navigation. Therefore, we believe PinchMove is a more accurate use for this scenario, not just in games, but for real simulations of astronaut navigation.

CONCLUSION AND FUTURE WORKS

We presented a novel navigation mechanic for virtual environments that allows manipulation of the viewport for near-field, accurate movement. Even though this paper presented a study on PinchMove using a VR environment, it can also easily be adopted into an AR environment for navigation, as long as proper 6DOF tracking is provided. This means that this

work is also supported by recent mixed reality hardware with inside-out tracking like Windows Mixed Reality and Hololens. We will also investigate different methods of interaction to compare with pinching, such as grasping and grabbing to determine the difference in perceived comfort and intuitiveness. Furthermore, we would like to investigate why some participants claim unimanual navigation to be more intuitive or feels inverted.

REFERENCES

1. Tom Bailey. 2017. Virtual Reality and the Body: Best Practices for Locomotion System Design. (2017).
2. Doug Bowman, Ernst Kruijff, Joseph J LaViola Jr, and Ivan P Poupyrev. 2004. *3D User Interfaces: Theory and Practice, CourseSmart eTextbook*. Addison-Wesley.
3. Doug Bowman, Chadwick Wingrave, Joshua Campbell, and Vinh Ly. 2001. Using pinch gloves (tm) for both natural and abstract interaction techniques in virtual environments. (2001).
4. Volkert Buchmann, Stephen Violich, Mark Billinghurst, and Andy Cockburn. 2004. FingARtips: gesture based direct manipulation in Augmented Reality. In *Proceedings of the 2nd international conference on Computer graphics and interactive techniques in Australasia and South East Asia*. ACM, 212–221. DOI: http://dx.doi.org/10.1145/988834.988871
5. Crytek. 2016. The Climb. (2016). http://www.theclimbgame.com/
6. Paul DiZio and James R Lackner. 1997. Circumventing side effects of immersive virtual environments. In *HCI (2)*. 893–896.
7. Mark H Draper, Erik S Viirre, Thomas A Furness, and Valerie J Gawron. 2001. Effects of image scale and system time delay on simulator sickness within head-coupled virtual environments. *Human factors* 43, 1 (2001), 129–146. DOI: http://dx.doi.org/10.1518/001872001775992552
8. Henry Been-Lirn Duh, Donald E Parker, and Thomas A Furness. 2004. An independent visual background reduced simulator sickness in a driving simulator. *Presence: Teleoperators and Virtual Environments* 13, 5 (2004), 578–588. DOI: http://dx.doi.org/10.1162/1054746042545283
9. Ajoy S Fernandes and Steven K Feiner. 2016. Combating VR sickness through subtle dynamic field-of-view modification. In *3D User Interfaces (3DUI), 2016 IEEE Symposium on*. IEEE, 201–210. DOI: http://dx.doi.org/10.1109/3DUI.2016.7460053
10. Michael J Griffin. 2012. *Handbook of human vibration*. Academic press.
11. Robert S. Kennedy, Norman E. Lane, Kevin S. Berbaum, and Michael G. Lilienthal. 1993. Simulator Sickness Questionnaire: An Enhanced Method for Quantifying Simulator Sickness. (1993). DOI: http://dx.doi.org/10.1207/s15327108ijap0303_3
12. Azam Khan, Ben Komalo, Jos Stam, George Fitzmaurice, and Gordon Kurtenbach. 2005. Hovercam: interactive 3d navigation for proximal object inspection. In *Proceedings of the 2005 symposium on Interactive 3D graphics and games*. ACM, 73–80. DOI: http://dx.doi.org/10.1145/1053427.1053439
13. S Knoedel and M Hachet. 2011. Multi-touch RST in 2D and 3D spaces: Studying the impact of directness on user performance. In *2011 IEEE Symposium on 3D User Interfaces (3DUI)*. 75–78. DOI: http://dx.doi.org/10.1109/3DUI.2011.5759220
14. Joseph J LaViola Jr. 2000. A discussion of cybersickness in virtual environments. *ACM SIGCHI Bulletin* 32, 1 (2000), 47–56. DOI: http://dx.doi.org/10.1145/333329.333344
15. Brian Lindenhof. 2016. Climbey. (2016). http://store.steampowered.com/app/520010/Climbey/
16. Daniel Medeiros, Eduardo Cordeiro, Daniel Mendes, Mauricio Sousa, Alberto Raposo, Alfredo Ferreira, and Joaquim Jorge. 2016. Effects of speed and transitions on target-based travel techniques. In *Proceedings of the 22nd ACM Conference on Virtual Reality Software and Technology - VRST '16*. ACM, 327–328. DOI: http://dx.doi.org/10.1145/2993369.2996348
17. Oculus. 2017. Introduction to Best Practices. (2017). https://developer3.oculus.com/documentation/intro-vr/latest/concepts/bp_intro/
18. Charles M Oman. 1990. Motion sickness: a synthesis and evaluation of the sensory conflict theory. *Canadian journal of physiology and pharmacology* 68, 2 (1990), 294–303. DOI:http://dx.doi.org/10.1139/y90-044
19. Michael Ortega, Wolfgang Stuerzlinger, and Doug Scheurich. 2015. SHOCam: A 3D Orbiting Algorithm. In *Proceedings of the 28th Annual ACM Symposium on User Interface Software & Technology*. ACM, 119–128. DOI: http://dx.doi.org/10.1145/2807442.2807496
20. Benjamin I Outram, Yun Suen Pai, Kevin Fan, Kouta Minamizawa, and Kai Kunze. 2016. AnyOrbit: Fluid 6DOF Spatial Navigation of Virtual Environments using Orbital Motion. In *Proceedings of the 2016 Symposium on Spatial User Interaction*. ACM, 199. DOI: http://dx.doi.org/10.1145/2983310.2989195
21. Yun Suen Pai and Kai Kunze. 2017. Armswing: Using Arm Swings for Accessible and Immersive Navigation in AR/VR Spaces. In *Proceedings of the 16th International Conference on Mobile and Ubiquitous Multimedia (MUM '17)*. ACM, New York, NY, USA, 189–198. DOI: http://dx.doi.org/10.1145/3152832.3152864
22. Ready at Dawn. 2017. Lone Echo. (2017). https://www.oculus.com/experiences/rift/1368187813209608/

23. Bernhard E. Riecke, Bobby Bodenheimer, Timothy P. McNamara, Betsy Williams, Peng Peng, and Daniel Feuereissen. 2010. Do we need to walk for effective virtual reality navigation? Physical rotations alone may suffice. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* 6222 LNAI (2010), 234–247. DOI: http://dx.doi.org/10.1007/978-3-642-14749-4_21

24. Martijn J Schuemie, Peter Van Der Straaten, Merel Krijn, and Charles A P G Van Der Mast. 2001. Research on presence in virtual reality: A survey. *CyberPsychology & Behavior* 4, 2 (2001), 183–201. DOI: http://dx.doi.org/10.1089/109493101300117884

25. Junbong Song, Sungmin Cho, Seung-Yeob Baek, Kunwoo Lee, and Hyunwoo Bang. 2014. GaFinC: Gaze and Finger Control interface for 3D model manipulation in CAD application. *Computer-Aided Design* 46 (2014), 239–245. DOI: http://dx.doi.org/10.1016/j.cad.2013.08.039

26. David J Sturman, David Zeltzer, and Steve Pieper. 1989. Hands-on interaction with virtual environments. In *Proceedings of the 2nd annual ACM SIGGRAPH symposium on User interface software and technology*. ACM, 19–24. DOI: http://dx.doi.org/10.1145/73660.73663

27. Evan A Suma, Seth Clark, David Krum, Samantha Finkelstein, Mark Bolas, and Zachary Warte. 2011. Leveraging change blindness for redirection in virtual environments. In *Virtual Reality Conference (VR), 2011 IEEE*. IEEE, 159–166. DOI: http://dx.doi.org/10.1109/VR.2011.5759455

28. Martin Sundin and Morten Fjeld. 2009. Softly Elastic 6 DOF Input. *International Journal of HumanâĂŞComputer Interaction* 25, 7 (2009), 647–691. DOI:http://dx.doi.org/10.1080/10447310902964124

29. Sam Tregillus, Majed Al Zayer, and Eelke Folmer. 2017. Handsfree Omnidirectional VR Navigation Using Head Tilt. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems (CHI '17)*. ACM, New York, NY, USA, 4063–4068. DOI: http://dx.doi.org/10.1145/3025453.3025521

30. Sam Tregillus and Eelke Folmer. 2016. VR-STEP: Walking-in-Place using Inertial Sensing for Hands Free Navigation in Mobile VR Environments. *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems - CHI '16* (2016), 1250–1255. DOI: http://dx.doi.org/10.1145/2858036.2858084

31. Michael Venturino. 1990. Performance and head movements using a helmet-mounted display with different sized fields-of-view. *Optical Engineering* 29, 8 (1990), 870–877. DOI: http://dx.doi.org/10.1117/12.55672

32. Colin Ware and Steven Osborne. 1990. Exploration and Virtual Camera Control in Virtual Three Dimensional Environments. In *Proceedings of the 1990 Symposium on Interactive 3D Graphics (I3D '90)*. ACM, New York, NY, USA, 175–183. DOI: http://dx.doi.org/10.1145/91385.91442

33. Thomas G Zimmerman, Jaron Lanier, Chuck Blanchard, Steve Bryson, and Young Harvill. 1987. A hand gesture interface device. In *ACM SIGCHI Bulletin*, Vol. 18. ACM, 189–192. DOI:http://dx.doi.org/10.1145/29933.275628