University of Nevada, Reno

OPTIMIZING NATURAL WALKING USAGE IN VR USING REDIRECTED TELEPORTATION

A Thesis Submitted in Partial Fulfillment

of the Requirements for the Degree of Master of Science in

Computer Science and Engineering

by

Hirav Parekh

Dr. Eelke Folmer / Thesis Advisor

December 2017

UNIVERSITY OF NEVADA RENO **THE GRADUATE SCHOOL**

We recommend that the thesis prepared
under our supervision by

HIRAV PAREKH

entitled

OPTIMIZING NATURAL WALKING USAGE IN VR USING REDIRECTED TELEPORTATION

be accepted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE

Eelke Folmer, Ph.D. – Advisor

Sergiu Dascalu, Ph.D. – Committee Member

Paul Macneilage, Ph.D. – Graduate School Representative

David Zeh, Ph.D. – Dean, Graduate School

December 2017

ABSTRACT

Virtual Reality (VR) has come a long way since its inception and with the recent advancements in technology, high end VR headsets are now commercially available. Although these headsets offer full motion tracking capabilities, locomotion in VR is yet to be fully solved due to space constraints, potential VR sickness and problems with retaining immersion. Teleportation is the most popular locomotion technique in VR as it allows users to safely navigate beyond the confines of the available positional tracking space without inducing VR sickness. It has been argued that the use of teleportation doesn't facilitate the use of natural walking input which is considered to have a higher presence because teleportation is faster, requires little physical effort and uses limited available tracking space. When a user walks to the edge of the tracking space, he/she must switch to teleportation. When navigating in the same direction, available walking space does not increase, which forces users to remain stationary and continue using teleportation. We present redirected teleportation, a novel locomotion method that increases tracking space usage and natural walking input by subtle reorientation and repositioning of the user. We first analyzed the positional tendencies of the users as they played popular games implementing teleportation and found the utilization of the tracking space to be limited. We then compared redirected teleportation with regular teleportation using a navigation task in three different environments. Analysis of our data show that although redirected walking takes more time, users used significantly fewer teleports and more natural walking input while using more of the available tracking space. The increase in time is largely due to users walking more, which takes more time than using teleportation. Our results provide evidence that redirected teleportation may be a viable approach to increase the usage of natural walking input while decreasing the dependency on teleportation.

ACKNOWLEDGEMENTS

I owe a huge debt of gratitude to my advisor, Dr. Eelke Folmer for all his support and guidance. His encouragement and feedback has been very crucial to me. I would like to thank the members of my committee Dr. Sergiu Dascalu and Dr. Paul Macneilage. I want to thank my parents, family members and friends for their love and support.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1
INTRODUCTION

Moving around freely has been a fundamental appeal of 3D games for decades, but facilitating this in virtual reality (VR) has been a major challenge [22]. Using natural walking input using positional tracking on PC VR platforms, generally delivers the most natural and immersive experiences [30, 36], while it reduces cognitive load [38] and minimizes the occurrence of VR sickness [20, 15]. Natural walking is bounded by the size of available positional tracking space [29]. Current consumer VR platforms (Vive/Oculus) support tracking areas up to 4.5 x 4.5 meters [8]. In practice, because these systems are installed in a home environment, the available tracking space is limited by the available space of a living room or office, with many users freeing up only enough space to meet the minimum tracking space requirements [8]. Because of these space constraints, some VR games limit gameplay to the available tracking space or they involve no navigation at all by offering an on-rails or wave-shooter like experiences [22]. It has been argued that such experiences offer limited engagement and a lower presence [33].

Natural walking doesn't scale beyond the confines of limited available tracking space, therefore to navigate large VR environments, users must switch to an alternative locomotion technique (ALT) that is generally activated using a controller. Popular ALTs include teleportation, directional movement (also known as full locomotion) and related vehicle movement. Because ALTs are typically activated using a controller, frequently having to switch from leg to hand input is cumbersome, and for full locomotion a closely related study [16] found this to lead to users largely abandoning natural walking input. Full and vehicle locomotion generate optical flow, which in the absence of vestibular and proprioceptive afferents, may

confuse the senses and lead to vection-induced VR sickness [17].

Teleportation is considered a risk free way of navigating in VR because it discontinuously translates the user's viewpoint, with no optical flow generated that can cause VR sickness [17]. The lack of optical flow limits the user's ability to perform path integration (e.g. estimating distance travelled) which may cause spatial disorientation [4, 2]. It has also been argued that teleportation breaks presence [4] because it lets users do something that doesn't exist in real life. For games, discontinuous displacement of users can alters intended gameplay [23], and is especially a challenge for multiplayer games as it becomes difficult to predict or follow the path of other users when they teleport around.

Available walking space is a crucial factor in the use of teleportation. If a user navigates in a particular direction using natural walking and reaches the edge of tracking space they are forced to switch to use teleport to continue their path. However, as long as they continue in the same direction, the available walking space won't increase which requires them to keep using teleportation. Users may take a few step backwards to create walking space teleport then walk forward but this leads to very unnatural oscillating behavior, which may impede presence. Overall, teleportation doesn't facilitate the use of natural walking input, which raises the question whether positional tracking is even needed. Because it is already faster than natural walking and requires negligible physical effort, over time, users may become lazy and adopt teleport as their primary means of locomotion [23], despite it having a lower presence than natural walking.

The contribution of this paper is as follows: (1) we present results from an empirical study on the usage of teleportation in commercially available games that shows that tracking space usage is limited when using teleport; and (2) we evaluate

an improved version of teleportation called *redirected teleportation*, that increases available walking space by subtle repositioning and reorientation of the user.

CHAPTER 2
BACKGROUND AND RELATED WORK

Being able to navigate beyond the confines of available tracking space while minimizing cybersickness, cost and maintaining a high presence is considered a major challenge for the mass adoption of VR [22]. Though natural walking offers the highest presence it doesn't scale to navigate large environments and alternative locomotion solutions need to be used.

Several attempts have been made to offer real walking in VR that covers virtual environments larger than the tracked physical space. Redirection techniques achieve this generally by inducing imperceptible gains to the virtual viewpoint to keep the user away from the boundaries of the tracking space [32]. Redirected walking techniques largely use curvature gains to give users the illusion of walking straight while they are actually moving on a curve [29, 26]. Walking redirection has been implemented using one of three strategies: Steer-to-Center, Steer-onto-Orbit, and Steer-to-Alternating-Targets [28]. Redirection with distractors techniques [27] scale head rotations while attending to a distractor to reorient users from the boundaries of the physical space. Scaled walking techniques [6, 12] apply translation gains that amplify physical steps to cover greater virtual distances. Aside from applying gains to the virtual viewpoint, imperceptible redirection has also been implemented by manipulating the virtual environment's architecture [34] and using visio-haptic interaction [24]. When these techniques fail to keep the user within the confines of the tracking space, resetting techniques that explicitly ask users to reorient themselves can be used. Examples include the Freeze-Backup, Freeze-Turn and 2:1-Turn resetting techniques proposed by Williams et al. [37].

Other solutions have attempted to offer natural walking in VR through the de-

velopment of walking-based devices that keep users stationary while naturally walking. These include omni-directional treadmills [7, 11], step-based devices [14, 13], and spherical walking platforms [9, 25]. Aside from the need for a dedicated hardware and limited commercial availability [35], these techniques are criticized for their limited safety, usability, and responsiveness [19]. Because they require users to be strapped into or onto them, these solutions are difficult to integrate with existing positional tracking systems, as they impede certain types of interactions, for example, kneeling down and picking up an object.

Another set of implementations augment real walking with magical repositioning techniques that offer users with the ability to travel to a distant virtual target. The most basic implementation is teleportation, where users select a virtual travel destination that is within their field of view (FoV). Teleportation is considered a safe ALT as it doesn't cause VR sickness. Depending on the implementation, the virtual viewpoint either moves instantly or at a certain rate [21] and the travel target can either be selected traditionally using a controller or via a body gesture [3]. Several approaches have aimed to improve teleportation. LaViola [18] presents a modification of teleport that requires users to step into a location on a map that is rendered at their feet in order to teleport to that location. Jumper [3] is a hands-free form of teleportation on PC VR platforms where users physically jump forward to a location specified by their gaze. Point and teleport [5] allows users to specify their post-teleport orientation.

Most closely related to our redirected teleportation is the work by Freitag et al [10] which presents a portal-based teleportation method for a CAVE. Users can select a location to teleport to upon which a portal slides up from the ground in the center of the CAVE space. If the portal is not visible in the user's current view,

a notification appears indicating to the user to turn left or right to see the portal. When stepping into the portal the user teleports and the amount of available walkable space is maximized. Besides a difference in VR platforms used, e.g., we focus on currently more popular VR HMD platforms, and we explore a more subtle way for reorienting and repositioning the user using a series of teleports (rather than a single one). Our approach doesn't require pop-up messages or drastic 180° turns that could break presence. In addition, our paper also presents empirical evidence that confirms the assumed limitations of teleportation usage that motivate the development of redirected teleportation.

CHAPTER 3

UNDERSTANDING POSITIONAL TRACKING USE

It has been suggested that teleportation can lead to users adopting a “lazy" gameplay style where they abandon the usage of natural walking [23]. Unfortunately we did not identify any empirical studies on the usage of teleportation that could support this assumption. Our first study collects positional tracking data of users playing four popular commercially available VR games that use teleport as an ALT. This study provides a better insight in how teleportation affects natural walking usage.

3.1 Instrumentation

For this study, we used the HTC Vive, a popular PC VR platform that allows full outside-in tracking of the HMD and two controllers using infrared “room scale" technology. Many VR games are available for this platform that fully utilize room scale tracking. The HTC Vive uses a pair of wall-mounted base stations that are placed across the room from each other above head height. The Vive offers 1080x1200 per-eye resolution at 90 Hz and a 110° FoV. For our study we configured our tracking space to have a size of 2.4m x 2.2m. This value was chosen based on results of a survey conducted by Vive [8] among 2,008 Vive users that found this was the average tracking space size users had available for VR.

The Vive headset is tethered to a PC using a cable, but this can impede moving freely within available tracking space, especially when the cables get twisted from the user turning. Therefore, we used a TPCAST [1] wireless adapter to allow users to play games untethered. The TPCast adapter consist of an HMD receiver that

mounts directly to the Vive headset, which then wirelessly communicates with an included PC transmitter. The TPCAST supports a tracking area of 5x5m. To power the HMD receiver, users wear a 354 gram 20,000mAH battery pack on their belt or pocket. According to their specifications [1], the TPCAST adds less than 2 milliseconds of latency, but we didn't observe a noticeable delay.

Data collection. A standalone script using OpenVR was used to access the HTC Viveâs hardware for positional tracking data through SteamVR. The script saved separate CSV files for positional data, tracking space corner positions, and time elapsed. Positional data in the form of an (x,y) pair was collected once every $100ms$.

3.2 Virtual Environment

After extensively playing various commercially available VR games that use teleportation, we selected four different VR games based on their popularity, e.g., they received mostly positive reviews ($n > 100$) on Valve's Steam digital distribution platform.

- **Arizona Sunshine:** is a zombie survival first-person shooter game that takes place in a large open desert environment.
- **Budget Cuts:** is a stealth VR game that requires solving puzzles and disabling robots in constrained indoor environment that is connected using an elevator.
- **Raw Data:** is a science fiction first person shooter that involves shooting robots in a large indoor environments.

- **Vanishing Realms:** is a first-person role playing game where users fight monsters using swords, bows and magic in an arena like environment.

3.3 Participants

We recruited 7 participants (7 males, average age 25.0, SD=4.6) for our user study. All participants had some VR experience, with two having lots of VR experience. None of the subjects self-reported any non-correctable impairments in perception or limitations in mobility.

3.4 Procedure

Participants played each of the four games in a randomized order, with approximately 15 minutes rest between each game. All games use teleportation with users selecting a location to teleport to using their controller and teleportation is then activated by pressing the trigger or the touchpad. Budget Cuts uses a slightly different teleportation mechanism as it show the user a preview of the area to be teleported to (see Figure 3.2). For each game, we selected a specific level or part to play and performed data collection as follows:

- **Arizona Sunshine.** Each participant played the first level of the campaign in Arizona Sunshine (see Figure 3.1). We start data recording after they finish the tutorial and go down the ladder. We stop data collection once the participant reaches the sand road before the museum. If the user dies before reaching this point they are asked to continue the game from the progression point at which

Figure 3.1: Arizona sunshine is the most popular zombie shooter game that offers a standard teleportation technique in which the user presses down on the trackpad of the controller and points the parabolic pointer at the location they want to teleport. Arizona sunshine offers users the ability to face any desired direction by rotating the controller.

they died.

- **Budget Cuts.** Each participant played Budget Cuts from the first level (see Figure 3.2). We start data recording as soon as the participant successfully finishes the first tutorial room and takes the elevator down to the next level. We stop data collection once the user reaches the area where they have to go into the ceiling. If the user dies before this point, they are asked to continue the game from the progression point at which they died.
- **Raw Data.** Each participant plays the campaign using the sword wielding character (see Figure 3.3) using casual difficulty. Participants first play a tutorial where they familiarize themselves with the game control and mechanics. The user is told not to use turrets or special abilities besides the default sword usage (regular hitting and throwing). The participant then plays the second level

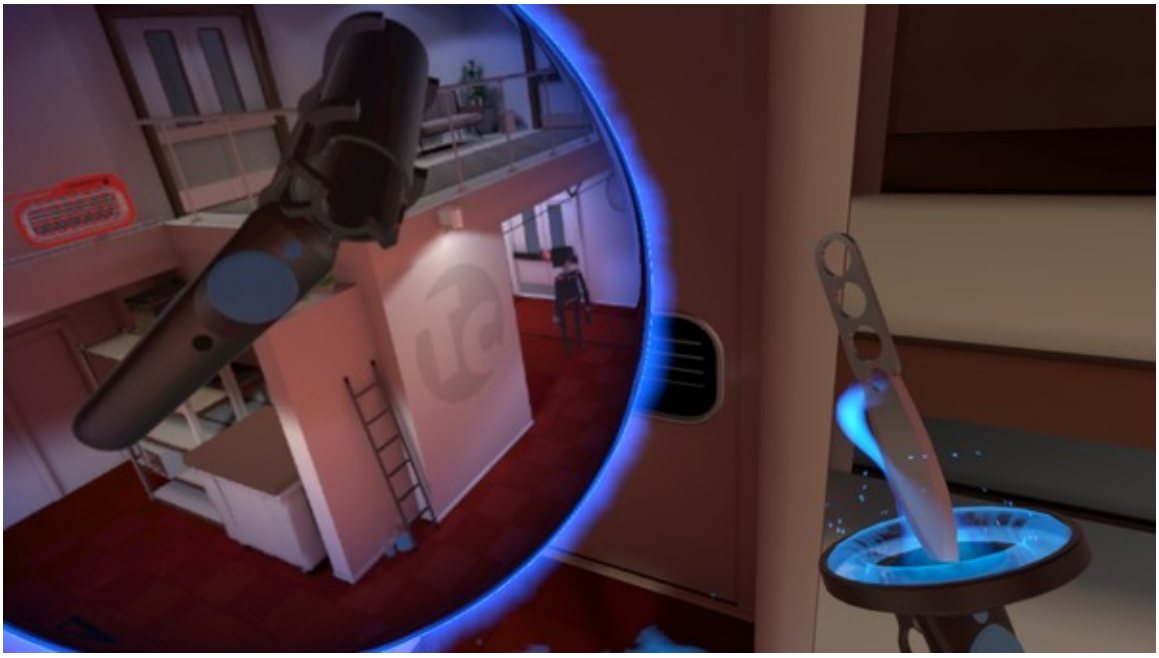

Figure 3.2: Budget Cuts is a popular stealth VR game that offers a unique teleportation mechanism that shows a preview of the area to be teleported to that is attached to the controller used for teleportation. This preview allows for a more seamless transition of viewpoints but also inspired the mechanism used in redirected teleportation.

of the campaign where we start the data collection and this level consists of five rounds of fighting enemies in an open space. Data collection stops when the user finishes the level.

- **Vanishing Realms.** Participants play the campaign with the auto pause feature turned off. We instructed users only to use the sword and the bow and arrow (when using magic there is a high chance that users kill themselves). Participants first play the tutorial (see Figure 3.4) to familiarize themselves with the weapons and combat. Each participant then plays the game from the first campaign checkpoint for Chapter 2, which is also when data collection starts. We stop data collection when the participant finishes all nine rounds of Chapter 2. If a participant dies, they are instructed to play from the current progression point.

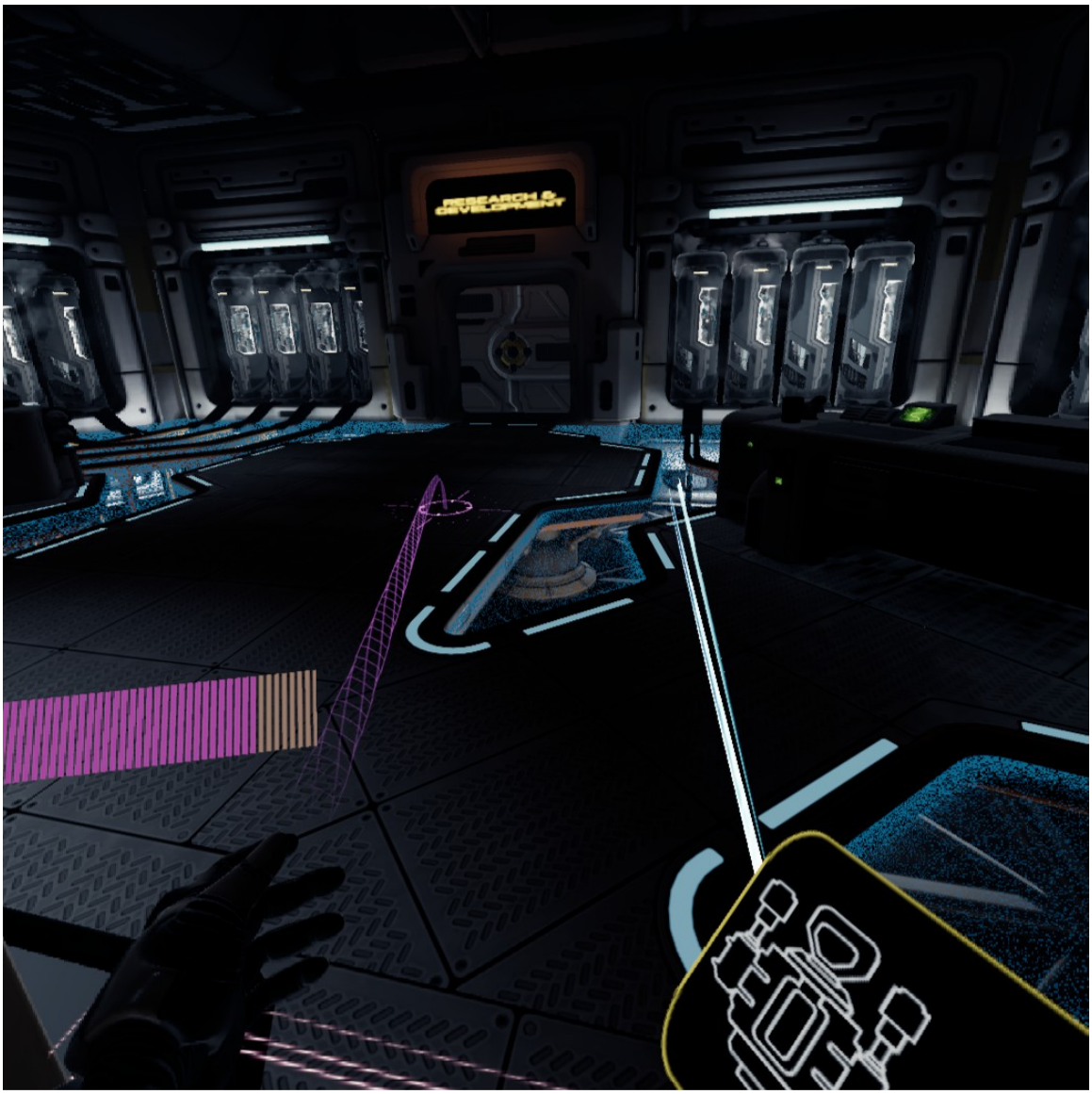

Figure 3.3: Raw Data is a popular action combat VR game that offers dash teleportation method in which the user is almost instantly teleported to destination with a continuous viewpoint displacement achieved by blurring the background.

3.5 Results

The CSV files that were collected for each user and game were imported into a custom R script for data analysis. We used a custom R script using bin2 from the ash library to generate heatmaps with contour lines from each participant's

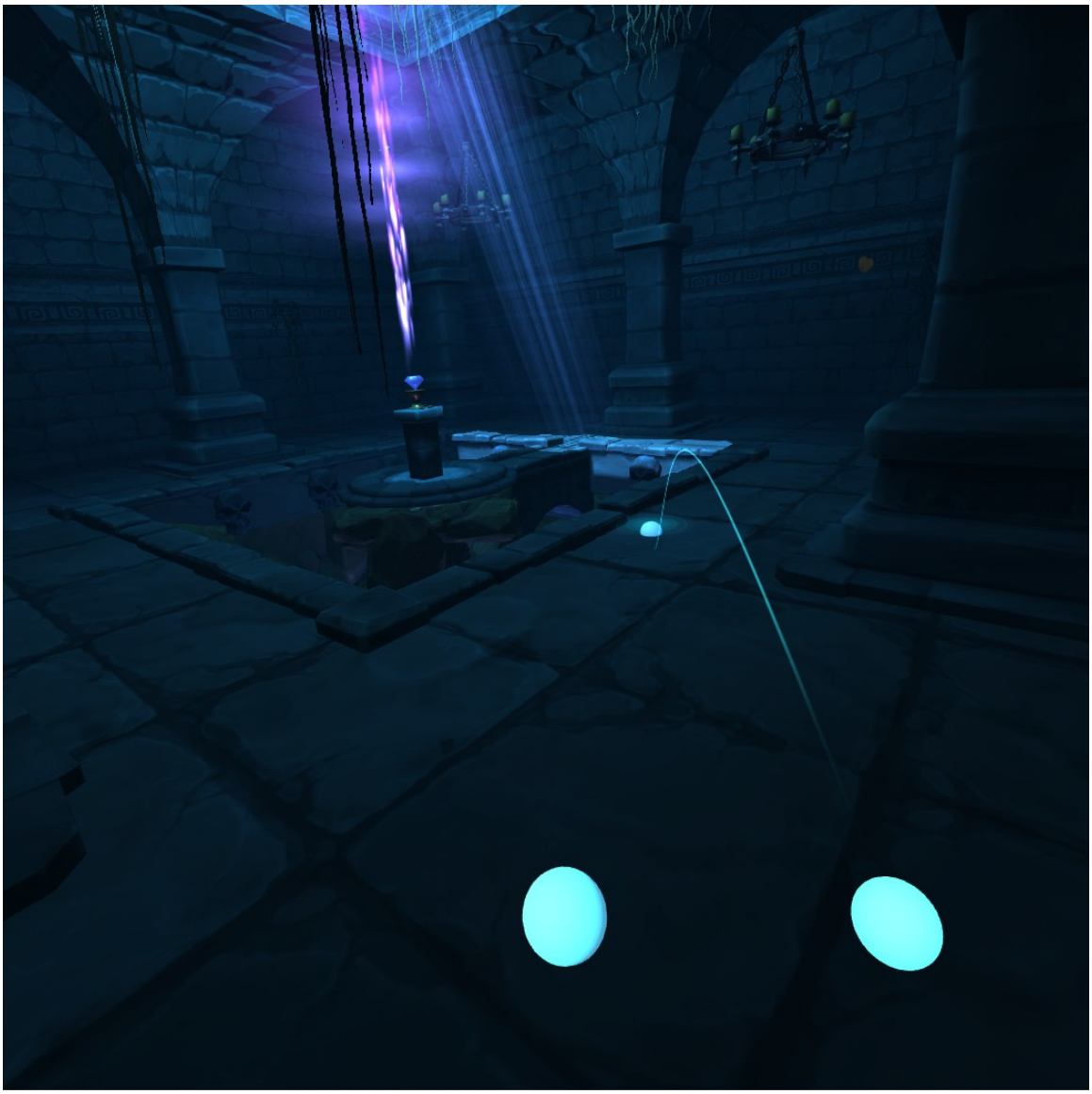

Figure 3.4: Vanishing Realms is an immersive Role Playing Game (RPG) in VR (an early access game). It offers a regular teleportation method with slight modifications to make it easier to teleport for example when the parabolic pointer hits an obstacle near the ground, it is bounced off the obstacle to a nearby location where the user can teleport.

positional data set. The corners of the rectangular tracking space are shown as black dots on the heatmaps. The same variable range was used in the binning of all of the participants' positional data and was determined from all of the participants' data. The results are listed in Figure 3.5.

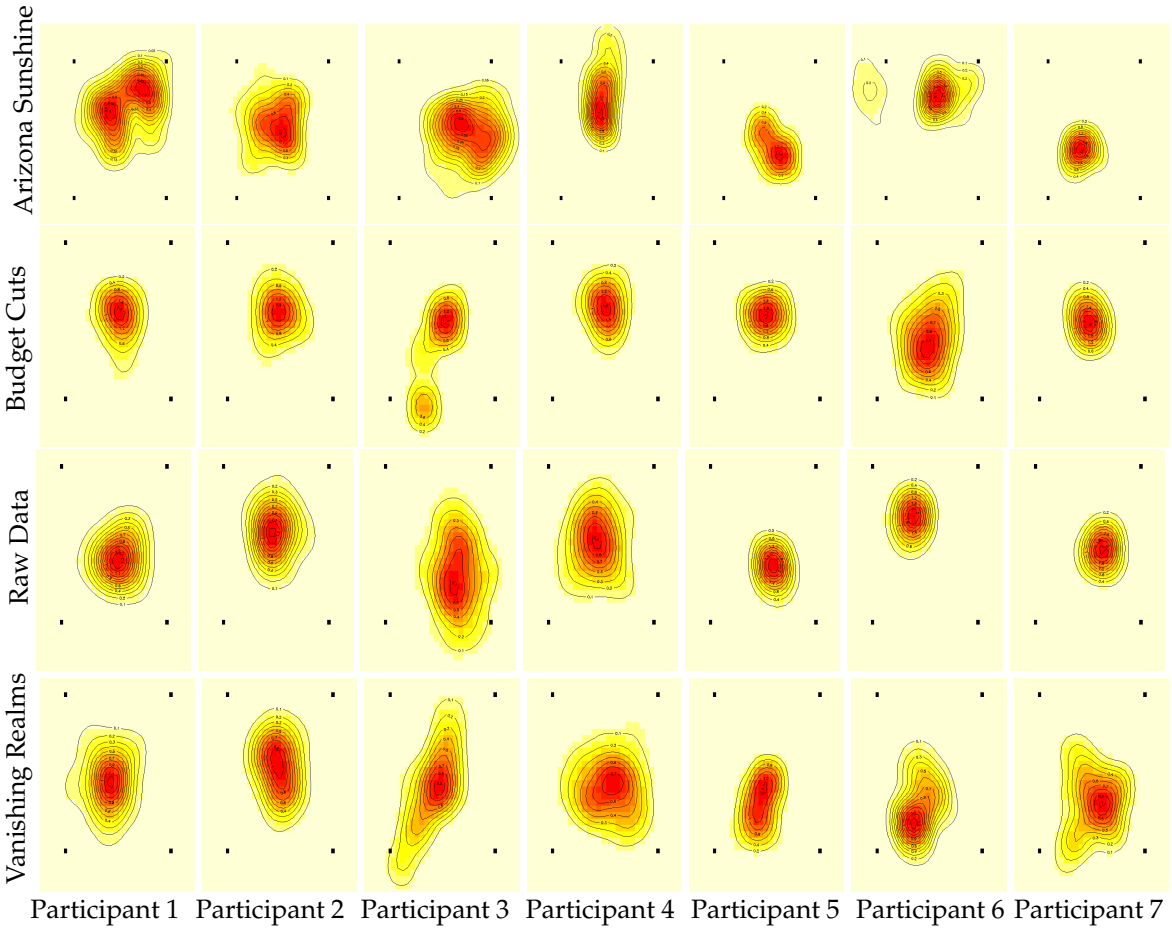

Figure 3.5: Heatmaps showing positional tracking data for 7 users playing four different VR games. The columns indicate the 7 participants and the rows are the games that were played which were: (1) Arizona Sunshine, (2) Budget Cuts, (3) Raw Data, and (4) Vanishing Realms. The black dots indicate the four corners of the available tracking space. heatmaps have been scaled and clipped such to optimize their visibility, but the available tracking space is the same for every heatmap. Darker red colors indicate a higher frequency of the user being in that location.

3.6 Discussion

The heat-maps vary significantly between participants and between games. For example, participant P3 and P4 move much more than P5 as is evident by the difference in diameter of the heatmap. Both P5 and P7 were experienced VR users, who apparently already used teleportation as their primarily form of locomotion

and who were observed to walk very little. Users also at times ran out of tracking space (especially participant 3). This happened mainly because users used walking to move around in VR and soon ended up near the edge of tracking space. At this point most users switched to teleport until they were attacked by the enemy and had to move around to dodge or had to reach out for picking up an item and ended up going out of tracking space. The heatmaps for Vanishing Realms, Raw Data and Arizona Sunshine in general are more spread out than for Budget cuts (except for P3 and P6). This could be explained by that by that users have to fight lots of enemies in these games, which requires moving around more to dodge attacks. Budget cuts has few enemies that have to be eliminated using stealth and are not actively attacking the user. Because it is a stealth game, and the teleport preview can be used to spy on areas that the user cannot see directly, it encourages the use of teleport and not necessarily natural walking, which is different from the other games we used. The levels in Budget cuts are also fairly small with users moving between floors using an elevator which doesn't require a lot of walking, unlike Arizona Sunshine where the level the participants played was elongated with users mostly navigating in a single direction. As a result for Arizona Sunshine, the heatmaps are often centered close to the border (P3, P4) or in a corner (P1, P5, P6, P7).

An analysis of tracking space utilization was done by binning the positional data of each participant that lies within the rectangular tracking space into 1600 bins (40 by 40). The section of tracking space that a bin represents is considered utilized if it is not 0. The fraction of bins that were utilized is the user's tracking space utilization rate. An analysis of tracking space utilization showed that at most 52% of the available tracking space was used (P3, Vanishing Realms) and at a minimum 7% (P7, Arizona Sunshine) and on average 26% of available tracking

space was used. A visual analysis of the 28 heatmaps shows that 11 were centered in a corner of a tracking space, 4 were centered at the edge and 13 were in the center of the tracking space. If we exclude the maps for Budget Cuts, which doesn't require users to walk a lot, 21 maps remain with 11 being centered in the corner, 7 in the center and 3 on the border.

CHAPTER 4
REDIRECTED TELEPORTATION

4.1 Design of Redirected Teleportation

Our first study confirms that teleportation currently implemented in VR games doesn't encourage the use of natural walking since users often stay at the edge or corner of the tracking space where available walking space is limited. Therefore, users primarily use teleportation to navigate virtual environments. A closer analysis of the actual paths taken by users reveals that many participants were found to "oscillate", e.g. they would run into the edge of tracking space, walk backward a bit, teleport, move forward to pick up an item, run out of space, walk backwards, teleport again (Figure 4.2: Left shows an example trace). This is especially the case when participants had to travel longer distances in a particular direction, leading to forced unnatural movement, which might be detrimental to presence.

Illusionary techniques such as redirected walking [29] circumvent tracking limitations by unobtrusively rotating the virtual world to redirect the user's position and orientation to let them walk on a curve. Though redirection techniques maintain a high presence, they require a large tracking space (radius > 22 meters) [31] to be imperceptible to the user. This tracking requirement typically exceed the tracking capabilities of existing consumer PC VR systems as well as the available space in a typical home. Nevertheless the various repositioning and reorientation techniques used in redirected walking techniques could be used to augment teleportation to more effectively use limited walking space.

If a user approaches the tracking space border, we need to reorient and repo-

sition them, such that their available walking space increases. To reposition the user we adopt the widely praised [23] preview mechanism used in Budget Cuts, e.g., this game features an oval preview that is rendered at the user's controller. This preview not only serves as a spyglass to look into an area before teleporting there but also allows for a more seamless transition of viewpoints, which may limit spatial disorientation; an unwanted byproduct of teleportation [4, 2].

Redirected teleportation aims to reposition and reorient the user towards the center, where available walking space is maximum, through the usage of teleports. For redirected teleportation, we show a similar preview that is generated to either the left or the right of the user's view at angle (θ) and distance (D) when they select a location to teleport to using their controller. The scene shown in the preview itself is further rotated along the Y-axis by a small amount (α). Our design assumes the right-handed coordinate system where the X-axis points to the right of the user's viewpoint and is parallel to the ground, and the Z-axis points forward and is parallel to the ground. The preview is spawned in the direction of least rotation which is determined by the Algorithm 1 where $\vec{v}_{UP}$ is the unit vector along the positive Y-axis $< 0, 1, 0 >$, $\vec{v}_{PD}$ is the unit vector representing the direction from the user to the teleport destination, and $\vec{v}_{PC}$ is the unit vector representing the direction from the user to the center.

Both directional vectors were projected on the XZ plane (ground) and transformed into the unit vectors $\vec{v}_{PD}$ and $\vec{v}_{PC}$ before the cross product. In the case where the cross product results in a zero vector, $\vec{v}_{PD}$ and $\vec{v}_{PC}$ are either parallel or anti-parallel. If they are parallel, meaning that the user is looking directly at the center of the tracking space, the preview is generated directly in front of the user instead of to the left or the right of the user. If they are anti-parallel, meaning that

```
if (v⃗_PD × v⃗_PC) = 0 then
    if (v⃗_PD − v⃗_PC) = 0 then
        no rotation;
    else
        rotate clockwise;
    end
else
    if (v⃗_PD × v⃗_PC) · v⃗_UP > 0 then
        rotate counter clockwise;
    else
        rotate clockwise;
    end
end
```

Algorithm 1: Conditions for determining direction of least rotation (or none)

the user is looking directly away from the center of the tracking space, the preview can be generated to either the left or the right of the user; we chose to generate it to the right. If the cross product results in a non-zero vector, the sign of the result from the dot product of the resulting vector and a unit vector along the positive Y axis determines the direction of least rotation.

In the case where the user does not need a maximum reorientation to reach the desired forward direction, the user should not be redirected more than necessary. The total amount the user is turned, e.g., the total rotation angle (γ), should not exceed the angle between $\vec{v}_{PD}$ and $\vec{v}_{PC}$ that is needed to rotate the user to the center (ϕ). As shown in Algorithm 2, when $\phi < \theta + \alpha$, the amount of preview rotation is first reduced (θ_R) and then, if necessary, the amount of viewpoint rotation is reduced (α_R) until $\gamma = \phi$.

The difference between Budget Cuts' teleportation and redirected teleportation is that the users must activate the teleport by stepping into the preview, so it functions as a portal. This leads to a repositioning of the user with a distance D and a rotation of their viewpoint with angle θ. We then take it one step further to reorient the user's viewpoint even further by applying a small amount of horizontal

if $\phi \geqslant \theta + \alpha$ **then**
  $\gamma = \theta + \alpha;$
else
  if $\phi < \theta$ **then**
    $\gamma = \theta_R = \phi, \alpha = 0;$
  end
  if $\phi > \theta$ **then**
    $\gamma = \theta, \alpha_R = \phi - \theta;$
  end
  if $\phi = \theta$ **then**
    $\gamma = \phi, \alpha = 0;$
  end
end

Algorithm 2: Conditions for reducing θ and/or α to prevent over-redirection

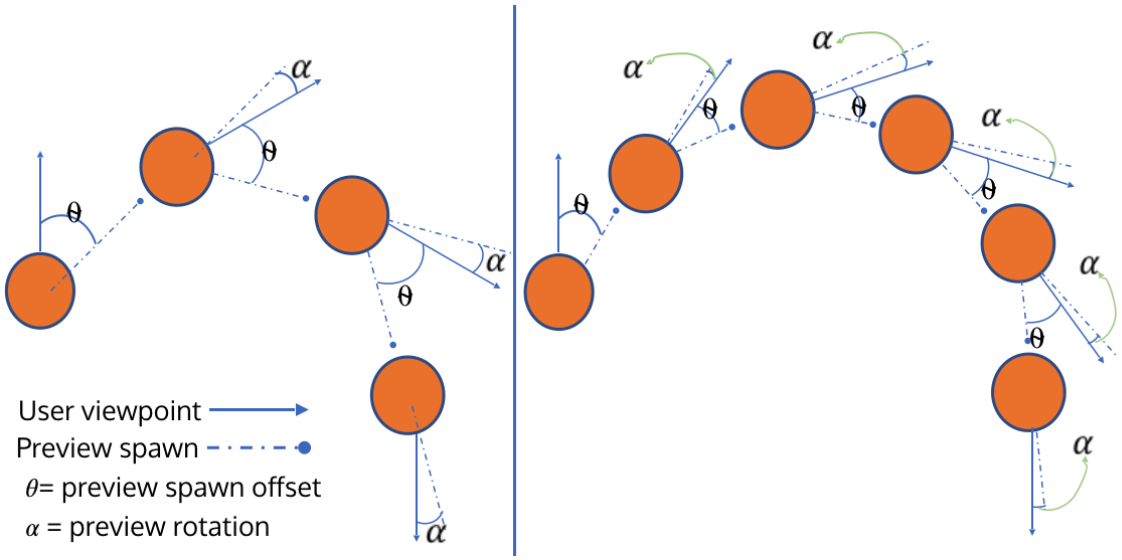

Figure 4.1: Consecutive teleports reposition and reorient the user such the amount of available walkable space increases. Left: $\alpha = 15, \theta = 45$, $D = 0.8m$ **Right:** $\alpha = 6, \theta = 30$, $D = 0.4m$

rotation (e.g. α) to the preview. The idea here is that when the user is navigating in a particular direction then steps through the preview, they find their current heading to be slightly misaligned with the direction they were going by α degrees. As a result they will quickly correct their orientation to face the direction they planned to go leading to a total maximum reorientation of $\theta + \alpha$ degrees at every teleport issued (see Figure 4.1).

Using preliminary experiments with the HTC Vive, we explored using different parameters (α, θ, D) to understand how it affects redirecting the user. To allow for the preview to be visible within the peripheral vision of the user, we found that the maximum value for α should be at most 45°. Regarding θ, larger values allow for more reorientation but this shift in orientation post-teleportation does become quite noticeable and using values up to 15° seemed largely imperceptible. The value D affects the “turning radius" of the user and must be chosen very carefully. Smaller values of D require less walking input for activating the teleport where larger values reposition the user more quickly and which will require fewer teleports. If the user is close to the edge of the tracking space, a large value of D may require them to cross the tracking space boundary, where collision free navigation cannot be guaranteed. In addition, using a too large value of D will have the user overshoot their repositioning from the center to the edge of the opposing tracking space.

Redirected teleportation can be configured using a single number (n) that states the number of teleports required to achieve a 180° change in orientation, and which allows the user to go from minimal to maximum available walking space. This (n) then defines a 3-tuple (α, θ, D). Figure 4.1 shows two different teleport sequence sequences of teleports for $(n = 3)$, with (15,45,0.8m) and $(n = 5)$ with (6,30,0.4m). In general to make redirected teleportation less obtrusive smaller values of α, θ and D should be used but more teleports are required to redirect the user. Figure 4.2: left shows the positional tracking data taken of a user navigating in a single direction using regular teleport where Figure 4.2: right shows it using redirected teleport. As can be seen from the trajectories, the user gets redirected around the tracking space boundaries and walks significantly more.

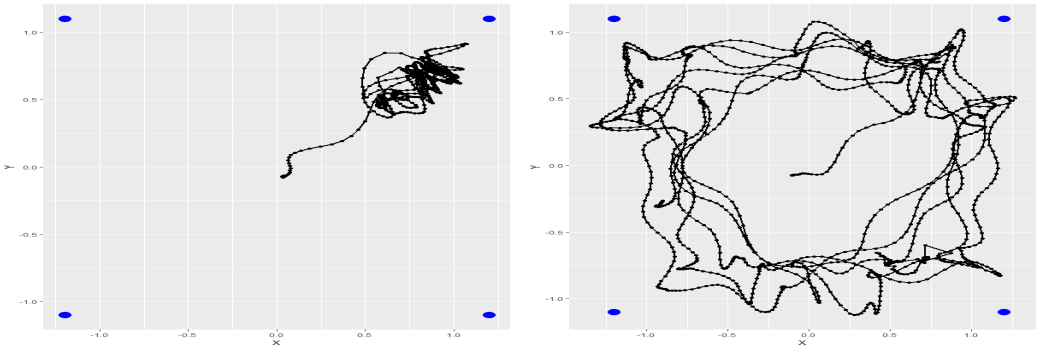

Figure 4.2: Sample trajectories. Left: user using regular teleport to navigate in the same direction; Right: user using redirected teleportation showing user walking through the tracking space.

4.2 User Study: Evaluation of Redirected Teleport

The goal of the user study was to compare redirected teleport to regular teleport to see whether it facilitates using more natural walking input.

4.2.1 Instrumentation

For this study, we used the same HTC Vive and TPCAST wireless adapter that we used in our first experiment. We configured our tracking space to have a size of 2.4m x 2.2m.

We built our navigation environments in Unity 5.6.1. A δ of 1.0 was used so a 1.0 meter displacement in the real world corresponded to a 1.0 meter viewpoint translation in the virtual environment. To implement both teleports we used the Vive-Teleporter asset. In regular teleport, participants can aim at a location to teleport to by holding down the controller touchpad and moving the orange circular cursor that is connected to a visual arch (see Figure 4.3:left) onto the desired teleport destination. By releasing the controller touchpad, the user is immediately

teleported to the desired teleport destination which the user previously aimed at. In redirected teleport, the method to aim at the desired teleport destination is the same. However, when the controller touchpad is released, a preview, rendered in a similar way as budget cuts with a blue glowing outline (see Figure 4.3:right), spawns near the user. The user must then walk into the previews to teleport to the desired teleport destination which the user previously aimed at.

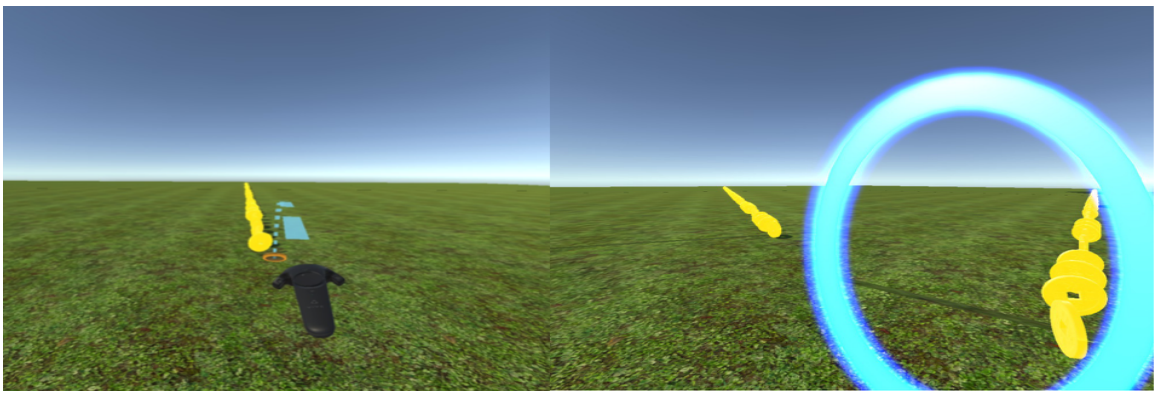

Figure 4.3: Regular vs Redirected Teleportation in action. Left: With regular teleportation, users select a teleport location with controller touchpad and are instantly teleported to the destination when they release it. Right: With redirected teleportation, when user release the controller touchpad, a preview of the teleport location appears either to the left, right or center and the user need to step into the preview to teleport to destination.

We implemented redirected teleport using the Vive Stereo Rendering Toolkit. We decided to redirect a user in three teleports and to achieve this we used the following values: $D = 0.5m$, $\alpha = 15$, and $\theta = 45$. These values were determined experimentally based on the available tracking space; and the HTC Vive's FoV. Redirected teleportation does not take into account whether the preview appears outside the tracking space, we did this because we wanted to offer our mechanism in a predictable way. Looking at the heatmaps in Figure 3.5, users also often veer outside the available tracking space. When setting up roomscale, Vive recommends to have at least $0.3m$ of collision free space available.

4.2.2 Virtual Environment

For this study, we created three different navigation environments and a navigation task that has users collect coins. A coin collection task allows for a controlled comparable navigation task between participants and assures participants cover the entire space. Participants needed to collect all coins while navigating through each environment using natural walking and teleport. All of the navigation environments had the same number of coins (57) and the participants collect the coins by walking over them upon which a sound is being played. We created the following three navigation environments:

- **Straight hallway.** Coins were placed equidistant (.72m) within a straight hallway that was 3m wide and 46m long. Participants start at least 3m away from the first coin and teleport to reach the first coin.
- **Square hallway**. Coins were placed equidistant (.72m) in a square hallway 6m wide with edges being 16m long. Participants start in a corner and can go clockwise or counterclockwise around the square hallway to collect the coins. Participants start 3m away from the first coin and teleport to reach the first coin. Square and straight were designed such that participants travelled the same distance, e.g., 40m.
- **Open space.** A 10 x 10m open area was created and the coins were randomly distributed inside. Participants can freely navigate in any direction to collect the coins and every participant uses the same coin distribution.

These three environments can be found in many VR games and allow for us to test the robustness and generality of our approach. For all of our VE's the Vive's chaperone system will render a translucent grid indicating the tracking

space boundaries when the user gets close to it.

4.3 Procedure and data collection

Data collection starts as soon as participant issue the first teleport and ends when picking up the last coin. We collect positional tracking data, number of teleports issued and total time. Each participant tests both teleportation methods (regular, redirected) which were counterbalanced to control for order effects. Because the navigation tasks increase in complexity we had participants first do the long hallway, then the straight and finally the open space. As a tutorial (with no data collection), participants first tested the straight hallway to familiarize themselves with each teleportation mechanism. When they felt comfortable they they started the sequence of navigation tasks (e.g. straight, square, open). Between teleportation trials, participants took off their headset and rested for 15 minutes. After both trials, users filled in a questionnaire to collect demographic data and qualitative results.

4.3.1 Participants

We recruited 14 participants (4 females, average age=25.71, SD=6.3) for our user study. All participants had experience with navigating 3D desktop environments. Three participants had no experience with VR, four had some VR experience and seven had lots of VR experience. None of the subjects self-reported any non-correctable impairments in perception or limitations in mobility. The user study was approved by an institutional review board.

4.4 Results

The following subsections discuss quantitative and qualitative results.

4.4.1 Quantitative Results

For our comparative quantitative analysis we analyzed: (1) total number of teleports used; (2) total time to collect all coins; (3) total distance travelled using positional tracking; and (4) tracking space usage for the three different navigation trials. Figure 4.1 lists the totaled results.

Table 4.1: Quantitative results listed for the three navigation environments and each teleportation method. Standard deviation listed between parentheses.

	Regular (SD)	**Redirected** (SD)
Straight		
#Teleports	35.50 (21.5)	22.71 (12.4)
Total time	149.06 (54.7)	204.57 (60.2)
Total distance	56.21(27.4)	80.89 (23.44)
Tracking space usage	21.04% (12.5%)	40.27% (10.9%)
Square		
#Teleports	32.36 (17.8)	23.21 (12.9)
Total time	155.80 (68.7)	201.37 (56.6
Total distance	56.21(27.4)	80.89 (23.44)
Tracking space usage	31.59% (15.0%)	43.01% (8.7%)
Open		
#Teleports	30.93 (16.11)	21.50 (4.9)
Total time	167.50 (46.38)	212.64 (55.4)
Total distance	56.21(27.4)	80.89 (23.44)
Tracking space usage	36.93% (15.1%)	49.07% (6.0%)

We used a two-way repeated measures ANOVA to determine the effect of teleportation method (regular, redirected) and navigation trial (straight, square, open) on each one of the three variables. We used a significance level of (p = .05).

For total number of teleports, Mauchly's test of sphericity indicated that the assumption of sphericity had been violated for the two-way interaction ($\chi^2_{\ 2}$ = 7.011, p = 0.03) A two way repeated-measures ANOVA with a Greenhouse-Geisser correction found no statistically significant two-way interaction between teleportation method and navigation trial ($F_{2,26}$ = .425, p = .587). Post hoc tests using a Bonferroni correction detected a statistically significant difference between teleportation methods (p = .07) but not between navigation trials (p > .05).

For total time, the same ANOVA found no statistically significant two-way interaction between teleportation method and navigation trial ($F_{2,26}$ = .155, p = .858) (Mauchlyâs tests assumed sphericity). Post hoc tests using a Bonferroni correction detected a statistically significant difference between teleportation methods (p < .001) but not between any navigation trials (p > .05).

For total distance, using redirected teleportation the preview was rendered 0.5 meters from the user, and for a fair comparison, we subtracted this distance for every teleport made by the user. A two-way repeated measures ANOVA found no statistically significant two-way interaction between teleportation method and navigation trial ($F_{2,26}$ = .061, p = .941) (Mauchlyâs tests assumed sphericity). Post hoc tests using a Bonferroni correction detected a statistically significant difference between teleportation methods (p < .001) and between the straight and open navigation trial (p = .015). This last result was expected as the distance and direction in which users had to travel varied significantly between the open space and the straight and square spaces. For tracking space usage, the same ANOVA found a statistically significant two-way interaction between teleportation method and navigation trial ($F_{2,26}$ = 3.279, p < .05). Post hoc tests using a Bonferroni correction detected a statistically significant difference between teleportation methods

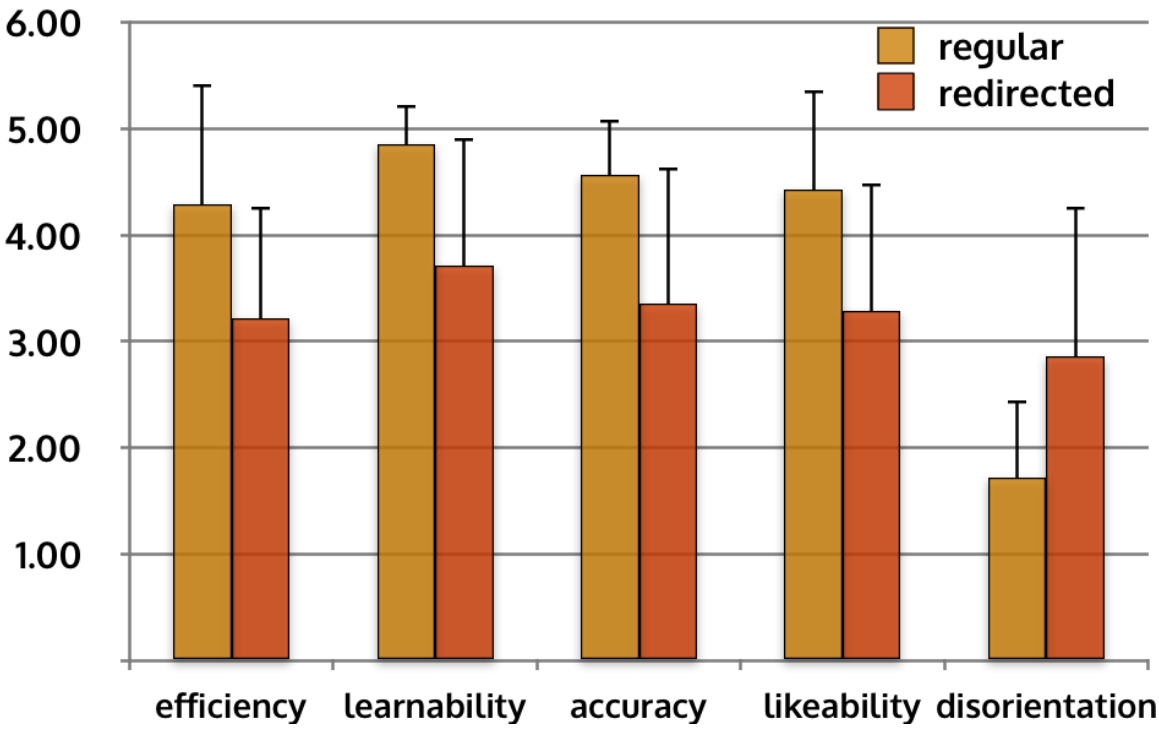

Figure 4.4: Columns show Likert scores (scale 1-5) for each teleportation method based on criteria: efficiency, learnability, accuracy, likeability and disorientation.

($p < .001$) and between the all navigation tasks ($p < .001$).

4.4.2 Qualitative results

We asked users to rate both teleportation methods they tested in terms of efficiency, learnability, accuracy, likability, and disorientation using a 5 point Likert scale. The results are summarized in Figure 4.4. A Wilcoxon Signed-Rank test found a significant difference in Likert scores for learnabilty ($p < .01$), likeability ($p < .01$) and disorientation ($p < .01$).

We also asked participants to rank each teleportation method on four usability criteria where participants could select "both" as a third option. Figure 4.5

lists the results. Ten participants found regular teleportation to be most efficient and three found redirected teleportation to be most efficient with one participant stating there was no difference. Ten participants found regular teleportation to be easiest to learn and four said there was no difference between them. Eight participants found regular teleportation to be most accurate, with two stating redirected teleportation was most accurate. Four participants said there was no difference. Regarding preference, eight participants liked regular teleport and five liked redirected teleport with one participant liking both methods. A χ^2 test found the rankings for efficiency and learnability to be statistically significantly different ($p < .05$).

Participants were also allowed to provide specific feedback on redirected teleportation. One participant felt a bit dizzy because redirected teleportation for the straight hallway made them walk in circles. Only one participant noticed that their orientation was slightly off post teleportation, and stated this was a bug. Two participants said that one time when a preview spawned, it was right at the edge of the tracking space and stepping inside it felt a bit uncomfortable. One participant suggested we create the preview behind the user when you are close to the tracking space border. One participant suggested it would be nice to have a sound played when the preview appears. Another participant stated that redirected teleportation was better for travelling longer distances while regular teleportation worked best for traveling short distances.

4.5 Discussion and Limitations

Quantitative results show that although redirected walking takes more time, users ended up using significantly fewer teleports; and used more natural walking input

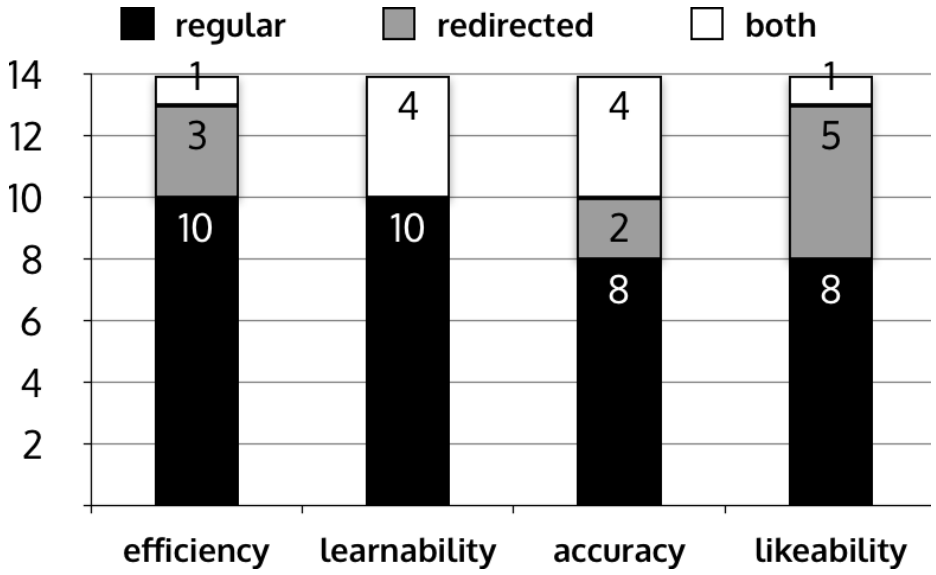

Figure 4.5: Subjective ranking of teleportation methods. Each number states how many participants preferred that teleportation for each usability criterion.

while also using a larger area of the available tracking space. Redirected teleportation is able to replace the oscillating behavior, we observed in our first study when users run out of space (see Figure 4.2: Left), with a more natural trajectory that navigates users from one edge of the tracking space to the other (see Figure 4.2: Right).

Only a single participant noticed their heading was slightly off post-teleportation, which indicates that maybe larger values of α could be used. However the largest gain in rotation is acquired using the offset of the preview. The FoV of the Vive is only 110° but future HMD may have larger field of views and θ may be set to larger values, since the human FoV can be as large as 180◦.

For the three environments, the benefits of redirected teleportation was most beneficial for the straight environment with a 36% reduction in number of teleports, a 44% increase in distance walked and a 91% increase in tracking space usage, but it also increased time with 37%. For the square hallway there was a

31% reduction in teleports, 44% increase in walked distance and 36% space usage while time increased by 29%. Finally the open environment had 31% decrease in teleports used, 36% increase in walked distance, and 33% increase in tracking space used, while time increased by 27%. These findings seem to suggest that redirected teleportation is especially beneficial for VR applications where navigation is largely constrained in the direction the user has to travel.

Though the benefits of redirected walking aren't very large for the total distance walked (e.g. 31% increase) the results for regular teleportation may been been a bit inflated by users who kept moving backwards and forwards to the border of the tracking space. This behavior helped create a bit more walking space to pick up an extra coin and was likely unique to the nature of our repetitive coin collection task. For other VR navigation tasks, users may be more likely to remain stationary, and the benefits in terms of distance walked will be significantly larger. We do believe our coin collection task does represent realistic navigation behavior in VR, as participants in many of the games we used for study 1 also teleported short distances, to avoid or fight enemies or pick up items.

Quantitative results show that redirected teleportation takes more time, which was in agreement with the qualitative Likert scores and ranking. However, the main reason why redirected teleportation takes more time is because users end up using fewer teleports and walk more which takes significantly more time than an instantaneous translation. The Likert scores for learnability were also significantly different though both high (4.86 for regular and 3.71 for redirected). Though redirected teleportation takes more effort to learn to use, given that all participants were able to complete all navigation tasks, this does not seem a reason for concern.

Regarding accuracy, both the ranking and the Likert scores found regular tele-

portation to be more accurate. Some participants were sometimes observed to skip a coin (when the teleport cursors is placed directly on or beyond the next coin) and would have to teleport back to complete the task. This seemed to occur for both both regular and redirected teleportation with the same frequency, so there was not really a difference in accuracy. Though redirected teleportation takes a bit longer this was because natural walking takes more time than teleportation.

Teleportation is known to cause spatial disorientation [4, 2], but this is often attributed to the lack of optical flow during the viewpoint transition that prevents path integration. The Likert results for whether the teleportation method used caused spatial disorientation were 1.71 for regular and 2.69 for redirected. Though these values were low (disagree to neutral) the difference was statistically significant. However, this difference was entirely due to one individual who stated getting dizzy from having to walk in circles during the straight navigation trial and who therefore strongly agreed with the statement that redirected teleportation caused disorientation. Though disorientation can be a result of dizziness, in general they are considered different things. When data for this participant is left out, there is no significant difference between teleportation methods ($\boldsymbol{U} = 52, z = -1.87, p = .062$). One way to reduce disorientation would be to set α to zero. We did not quantitatively assess spatial orientation, for example, using an object search task [4], but we aim to do this in future work to see whether redirected teleportation has some of the benefits associated with the use of natural walking, like increased spatial awareness.

Though there was an overall preference for regular teleportation, redirected teleportation took fewer teleports and because users walked more it also took longer. Natural walking is generally considered to offer a higher presence [30, 36]

than using teleportation. Redirected teleportation increases the use of natural walking, which suggests it increases presence. However, we did not attempt to directly assess presence for redirected teleportation as teleportation is generally considered to have a low presence [4] and because we only compare two different teleportation methods with each other and not against a locomotion method that offers a higher presence.

CHAPTER 5
CONCLUSION

Teleportation is a widely used, safe locomotion method that doesn't cause VR sickness and that lets users navigate beyond the confines of available tracking space This paper provides empirical evidence that teleportation doesn't optimize the usage of natural walking input which is considered to have a higher presence. Users often remain stationary with a high risk of them adopting teleportation as their primary means for travel. This paper presents redirected teleportation which is an improved version of teleport that uses iterative reorientation and repositioning to redirect the user back to the center of the tracking space, where available walking space is largest. A comparative user study finds that using redirected teleport users used fewer teleports and walked more while utilizing a larger portion of available tracking space. Redirected teleport was found to be slower than regular teleport due to the fact that walking takes more time.

5.1 Future Work

Future work will focus on evaluating how the benefits generated by redirected teleportation vary depending on the available tracking space size. Our study used an "average" tracking space size, e.g., 2.2m x 2.4m, based on results from a survey [8] but we did not explore varying this. With a larger tracking space users are less likely to run out of walking space, so the benefits may be smaller though there is also more space available to redirect the user towards the center. For smaller spaces, users are anticipated run out of walking space more quickly thus the larger the potential benefits of using redirected teleportation. However, for very small spaces there may not be enough space to redirect the user, with users having to

continuously step over of the tracking space boundary.

To avoid a preview being generated outside the available tracking space –which may cause users to run into obstacles– we will explore dynamically adjusting the value of D, which have not explored. However, care must be taken to offer redirected teleport in a consistent and predictable way to the user. Another issue that should be addressed is when redirection requires the preview to appear at a certain side, but the user's movement is obstructed because of an obstacle such as a wall. Feasible strategies will be explored to deal with such situations which may require redirected teleportation to first guide the user away from an obstacle and then redirect their position and orientation.

BIBLIOGRAPHY

[1] Tpcast wireless adapter for vive https://www.tpcast.cn/h_en/index.html.

[2] Niels H Bakker, Peter O Passenier, and Peter J Werkhoven. Effects of head-slaved navigation and the use of teleports on spatial orientation in virtual environments. *Human Factors: The Journal of the Human Factors and Ergonomics Society*, 45(1):160–169, 2003.

[3] Benjamin Bolte, Frank Steinicke, and Gerd Bruder. The jumper metaphor: an effective navigation technique for immersive display setups. In *Proceedings of Virtual Reality International Conference*, 2011.

[4] Doug Bowman, David Koller, Larry F Hodges. Travel in immersive virtual environments: An evaluation of viewpoint motion control techniques. In *Virtual Reality Annual International Symposium, 1997., IEEE 1997*, pages 45–52. IEEE, 1997.

[5] Evren Bozgeyikli, Andrew Raij, Srinivas Katkoori, and Rajiv Dubey. Point and teleport locomotion technique for virtual reality. In *Proceedings of the 2016 Annual Symposium on Computer-Human Interaction in Play*, CHI PLAY '16, pages 205–216, New York, NY, USA, 2016. ACM.

[6] Gabriel Cirio, Maud Marchal, Tony Regia-Corte, and Anatole Lécuyer. The magic barrier tape: a novel metaphor for infinite navigation in virtual worlds with a restricted walking workspace. In *Proceedings of the 16th ACM Symposium on Virtual Reality Software and Technology*, pages 155–162. ACM, 2009.

[7] Rudolph P Darken, William R Cockayne, and David Carmein. The omni-directional treadmill: a locomotion device for virtual worlds. In *Proceedings of UIST'97*, pages 213–221. ACM, 1997.

[8] Jamie Feltham. Vrfocus: Htc: People will use 'a smaller space' for vive's room scale tracking, https://www.vrfocus.com/2015/11/htc-people-will-use-a-smaller-space-for-vives-room-scale-tracking/, Last accessed on Nov 16, 2017.

[9] Kiran J Fernandes, Vinesh Raja, and Julian Eyre. Cybersphere: the fully immersive spherical projection system. *Communications of the ACM*, 46(9):141–146, 2003.

[10] Sebastian Freitag, Dominik Rausch, and Torsten Kuhlen. Reorientation in virtual environments using interactive portals. In *3D User Interfaces (3DUI), 2014 IEEE Symposium on*, pages 119–122. IEEE, 2014.

[11] John M Hollerbach, Yangming Xu, Robert R Christensen, and Stephen C Jacobsen. Design specifications for the second generation sarcos treadport locomotion interface. In *Haptics Symposium, Proc. ASME Dynamic Systems and Control Division*, volume 69, pages 1293–1298, 2000.

[12] Victoria Interrante, Brian Ries, and Lee Anderson. Seven league boots: A new metaphor for augmented locomotion through moderately large scale immersive virtual environments. In *3D User Interfaces, 2007. 3DUI'07. IEEE Symposium on*. IEEE, 2007.

[13] Hiroo Iwata, Hiroaki Yano, Hiroyuki Fukushima, and Haruo Noma. Circulafloor [locomotion interface]. *IEEE Computer Graphics and Applications*, 25(1):64–67, 2005.

[14] Hiroo Iwata, Hiroaki Yano, and Fumitaka Nakaizumi. Gait master: A versatile locomotion interface for uneven virtual terrain. In *Virtual Reality, 2001. Proceedings. IEEE*, pages 131–137. IEEE, 2001.

[15] Beverly K Jaeger and Ronald R Mourant. Comparison of simulator sickness using static and dynamic walking simulators. In *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, volume 45, pages 1896–1900. SAGE Publications, 2001.

[16] Eelke Folmer Jiwan Bhandari, Sam Tregillus. Legomotion: Scalable walking-based virtual locomotion. In *ACM Symposium for Virtual Reality Software and Technology (VRST'17)*, 2017.

[17] Behrang Keshavarz, Bernhard E Riecke, Lawrence J Hettinger, and Jennifer L Campos. Vection and visually induced motion sickness: how are they related? *Frontiers in psychology*, 6:472, 2015.

[18] Joseph J LaViola Jr, Daniel Acevedo Feliz, Daniel F Keefe, and Robert C Zeleznik. Hands-free multi-scale navigation in virtual environments. In *Proceedings of the 2001 Symposium on Interactive 3D Graphics*, pages 9–15. ACM, 2001.

[19] Joseph J LaViola Jr, Ernst Kruijff, Rayan P McMahan, Doug A Bowman, and

Ivan P Poupyrev. *3D User Interfaces: Theory and Practice*. Addison-Wesley, 2017.

[20] Gerard Llorach, Alun Evans, and Josep Blat. Simulator sickness and presence using hmds: comparing use of a game controller and a position estimation system. In *Proceedings of the 20th ACM Symposium on Virtual Reality Software and Technology*, pages 137–140. ACM, 2014.

[21] Jock D Mackinlay, Stuart K Card, and George G Robertson. Rapid controlled movement through a virtual 3d workspace. In *ACM SIGGRAPH Computer Graphics*, volume 24, pages 171–176. ACM, 1990.

[22] Emanuel Maiberg. Virtual reality's locomotion problem, `http://motherboard.vice.com/read/virtual-realitys-locomotion-problem`, Last accessed on Nov 16, 2017.

[23] John Martindale. Digital trends: How should we move around in VR? nobody has figured it out yet. `http://www.digitaltrends.com/virtual-reality/vr-locomotion-movement-omni-hover-junkers`, Last accessed on Nov 16, 2017.

[24] Keigo Matsumoto, Yuki Ban, Takuji Narumi, Yohei Yanase, Tomohiro Tanikawa, and Michitaka Hirose. Unlimited corridor: redirected walking techniques using visuo haptic interaction. In *ACM SIGGRAPH 2016 Emerging Technologies*, page 20. ACM, 2016.

[25] Eliana Medina, Ruth Fruland, and Suzanne Weghorst. Virtusphere: Walking in a human size vr âhamster ballâ. In *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, volume 52, pages 2102–2106. SAGE Publications Sage CA: Los Angeles, CA, 2008.

[26] Norbert Nitzsche, Uwe D Hanebeck, and Günther Schmidt. Motion compression for telepresent walking in large target environments. *Presence: Teleoperators and Virtual Environments*, 13(1):44–60, 2004.

[27] Tabitha C Peck, Henry Fuchs, and Mary C Whitton. The design and evaluation of a large-scale real-walking locomotion interface. *IEEE Transactions on Visualization and Computer Graphics*, 18(7):1053–1067, 2012.

[28] Sharif Razzaque. *Redirected walking*. PhD thesis, University of North Carolina at Chapel Hill, NC, USA, 2005. AAI3190299.

[29] Sharif Razzaque, Zachariah Kohn, and Mary C Whitton. Redirected walking. In *Proceedings of EUROGRAPHICS*, volume 9, pages 105–106. Citeseer, 2001.

[30] Mel Slater, Martin Usoh, and Anthony Steed. Taking steps: the influence of a walking technique on presence in virtual reality. *ACM Transactions on Computer-Human Interaction (TOCHI)*, 2(3):201–219, 1995.

[31] Frank Steinicke, Gerd Bruder, Jason Jerald, Harald Frenz, and Markus Lappe. Estimation of detection thresholds for redirected walking techniques. *IEEE Transactions on Visualization and Computer Graphics*, 16(1):17–27, 2010.

[32] Frank Steinicke, Gerd Bruder, Luv Kohli, Jason Jerald, and Klaus Hinrichs. Taxonomy and implementation of redirection techniques for ubiquitous passive haptic feedback. In *Cyberworlds, 2008 International Conference on*, pages 217–223. IEEE, 2008.

[33] Chris Suellentrop. Kotaku: Virtual reality's movement problem: It isn't nausea, `http://kotaku.com/vrs-real-motion-sickness-how-are-we-going-to-get-aroun-1713863822`, Last accessed on Nov 16, 2017.

[34] Evan A Suma, Zachary Lipps, Samantha Finkelstein, David M Krum, and Mark Bolas. Impossible spaces: Maximizing natural walking in virtual environments with self-overlapping architecture. *IEEE Transactions on Visualization and Computer Graphics*, 18(4):555–564, 2012.

[35] Raymong Wong. Mashable: VR startups: Stop trying to make virtual reality treadmills a thing, `http://mashable.com/2015/06/20/virtual-reality-treadmills//#THGLNhiWviqm`, Last accessed on Nov 16, 2017.

[36] Martin Usoh, Kevin Arthur, Mary C Whitton, Rui Bastos, Anthony Steed, Mel Slater, and Frederick P Brooks Jr. Walking> walking-in-place> flying, in virtual environments. In *Proceedings of the 26th Conference on Computer Graphics and Interactive Techniques*, pages 359–364, 1999.

[37] Betsy Williams, Gayathri Narasimham, Bjoern Rump, Timothy P McNamara, Thomas H Carr, John Rieser, and Bobby Bodenheimer. Exploring large virtual environments with an hmd when physical space is limited. In *Proceedings of the 4th Symposium on Applied Perception in Graphics and Visualization*, pages 41–48. ACM, 2007.

[38] Catherine A Zanbaka, Benjamin C Lok, Sabarish V Babu, Amy Catherine

Ulinski, and Larry F Hodges. Comparison of path visualizations and cognitive measures relative to travel technique in a virtual environment. *IEEE Transactions on Visualization and Computer Graphics*, 11(6):694–705, 2005.

APPENDIX A

SOURCE CODE FOR REDIRECTED TELEPORTATION

Listing A.1: Code snippet from source code

```
1  //if pickup up all the coins, test is over.
2  if ((coinPickUpScript.coinCount == totalCoinCount) && (test_ongoing))
3  {
4      test_ongoing = false;
5      testCompleteCanvas.SetActive(true);
6  }
7  // If we are currently teleporting
8  if (CurrentTeleportState == TeleportState.Teleporting)
9  {
10     // Wait until half of the teleport time has passed before the
           next event
11     if (Time.time - TeleportTimeMarker >= TeleportFadeDuration / 2)
12     {
13         if (FadingIn)
14         {
15             // We have finished fading in
16             CurrentTeleportState = TeleportState.None;
17         }
18         else
19         {
20             // We have finished fading out - time to teleport!
21             Vector3 offset = OriginTransform.position - HeadTransform
                   .position;
22             offset.y = 0;
23             OriginTransform.position = Pointer.SelectedPoint + offset
                   ;
24         }
```

```
25
26          TeleportTimeMarker = Time.time;
27          FadingIn = !FadingIn;
28      }
29  }
30  // At this point, we are NOT actively teleporting
31  else if (CurrentTeleportState == TeleportState.Selecting)
32  {
33      // Here, there is an active controller - user is holding down on
            the trackpad.
34      // Poll controller for pertinent button data
35      int index = (int)ActiveController.index;
36      var device = SteamVR_Controller.Input(index);
37      bool shouldTeleport = device.GetPressUp(SteamVR_Controller.
            ButtonMask.Touchpad);
38      bool shouldCancel = device.GetPressUp(SteamVR_Controller.
            ButtonMask.Grip);
39      if (shouldTeleport || shouldCancel)
40      {
41          //start timer
42          if (first_teleport)
43          {
44              first_teleport = false;
45              test_start_time = Time.time;
46              test_ongoing = true;
47          }
48
49          if (shouldTeleport && Pointer.PointOnNavMesh)
50          {
51              // Begin teleport sequence
52              PortalObject.SetActive(true);
```

```
53              CurrentTeleportState = TeleportState.None;
54
55              //position of player head
56              Vector3 headPosition = HeadTransform.position;
57              Vector3 playerToCenter = OriginTransform.position -
                    headPosition;
58              playerToCenter.y = 0;
59
60              Vector3 hmdDirection = HeadTransform.forward;
61              hmdDirection.y = 0;
62
63              Vector3 portalSpawnDirection = Vector3.RotateTowards(
                    hmdDirection, playerToCenter, PortalMaxAngleOffset
                    /180*Mathf.PI, 0);
64              portalSpawnDirection.Normalize();
65
66              //move portal
67              PortalTransform.position = HeadTransform.position + (
                    portalSpawnDirection * PortalDistanceFromPlayer);
68              PortalTransform.position = new Vector3(PortalTransform.
                    position.x, PortalHeight, PortalTransform.position.z);
69              //direction from portal to player as unit vector
70              Vector3 portalDirectionToPlayer = HeadTransform.position
                    - PortalTransform.position;
71              portalDirectionToPlayer.y = 0;
72              portalDirectionToPlayer.Normalize();
73
74              //orient portal towards player
75              Quaternion rotation = Quaternion.LookRotation(-
                    portalDirectionToPlayer);
76              PortalTransform.rotation = rotation * Quaternion.Euler(0,
```

```
            0, 0);
77
78          PortalDestination.transform.position = Pointer.
                SelectedPoint;
79
80          Vector3 portalDestinationLookRotation = -new Vector3(
                HeadTransform.position.x, 0, HeadTransform.position.z)
                 + new Vector3(Pointer.SelectedPoint.x, 0, Pointer.
                SelectedPoint.z);
81
82          float sign = Mathf.Sign(Vector3.Dot(Vector3.up, Vector3.
                Cross(portalDestinationLookRotation, playerToCenter)))
                ;
83
84          Quaternion portalDestinationLookRotationQuat = Quaternion
                .LookRotation(-new Vector3(HeadTransform.position.x,
                0, HeadTransform.position.z) + new Vector3(Pointer.
                SelectedPoint.x, 0, Pointer.SelectedPoint.z));
85          Quaternion playerToCenterQuat = Quaternion.LookRotation(
                playerToCenter);
86
87          float angleBetweenPortalDestinationLookAndPlayerToCenter
                = Quaternion.Angle(portalDestinationLookRotationQuat,
                playerToCenterQuat);
88          if (angleBetweenPortalDestinationLookAndPlayerToCenter >=
                 PortalCameraAngleOffset + PortalMaxAngleOffset)
89          {
90              if (sign > 0)
91              {
92                  Quaternion portalDestinationRotate = Quaternion.
                        AngleAxis(-PortalCameraAngleOffset, Vector3.up
```

```
                    );
93                  PortalDestination.transform.rotation =
                        portalDestinationLookRotationQuat *
                        portalDestinationRotate;
94              }
95              else
96              {
97                  Quaternion portalDestinationRotate = Quaternion.
                        AngleAxis(PortalCameraAngleOffset, Vector3.up)
                        ;
98                  PortalDestination.transform.rotation =
                        portalDestinationLookRotationQuat *
                        portalDestinationRotate;
99              }
100         }
101         else if (
                angleBetweenPortalDestinationLookAndPlayerToCenter >
                PortalMaxAngleOffset)
102         {
103             //do portal camera rotation up to
                    angleBetweenPortalDestinationLookAndPlayerToCenter
104             Debug.Log("in else");
105             if (sign == 0)
106                 Debug.Log(sign);
107             else if (sign > 0)
108             {
109                 Quaternion portalDestinationRotate = Quaternion.
                        AngleAxis(-(
                        angleBetweenPortalDestinationLookAndPlayerToCenter
                         - PortalMaxAngleOffset), Vector3.up);
110                 PortalDestination.transform.rotation =
```

```
                        portalDestinationLookRotationQuat *
                        portalDestinationRotate;
111                 }
112                 else
113                 {
114                     Quaternion portalDestinationRotate = Quaternion.
                        AngleAxis(
                        angleBetweenPortalDestinationLookAndPlayerToCenter
                         - PortalMaxAngleOffset, Vector3.up);
115                     PortalDestination.transform.rotation =
                        portalDestinationLookRotationQuat *
                        portalDestinationRotate;
116                 }
117             }
118             else if (
                angleBetweenPortalDestinationLookAndPlayerToCenter <=
                PortalMaxAngleOffset)
119             {
120                 //do not turn camera
121                 Debug.Log("Do not turn camera");
122             }
123
124             stereoRenderer.m_canvasOriginWorldPosition = new Vector3(
                PortalTransform.position.x,PortalTransform.position.y,
                PortalTransform.position.z);
125             stereoRenderer.m_canvasOriginWorldRotation =
                PortalTransform.rotation.eulerAngles;
126         }
127         else
128             CurrentTeleportState = TeleportState.None;
129
```

```
130         // Reset active controller, disable pointer, disable visual
                indicators
131         ActiveController = null;
132         Pointer.enabled = false;
133         RoomBorder.enabled = false;
134         //RoomBorder.Transpose = Matrix4x4.TRS(OriginTransform.
                position, Quaternion.identity, Vector3.one);
135         if (NavmeshAnimator != null)
136             NavmeshAnimator.SetBool(EnabledAnimatorID, false);
137
138         Pointer.transform.parent = null;
139         Pointer.transform.position = Vector3.zero;
140         Pointer.transform.rotation = Quaternion.identity;
141         Pointer.transform.localScale = Vector3.one;
142     }
143     else
144     {
145         // The user is still deciding where to teleport and has the
                touchpad held down.
146         // Note: rendering of the parabolic pointer / marker is done
                in ParabolicPointer
147         Vector3 offset = HeadTransform.position - OriginTransform.
                position;
148         offset.y = 0;
149
150         // Haptic feedback click every [HaptickClickAngleStep]
                degrees
151         float angleClickDiff = Pointer.CurrentParabolaAngleY -
                LastClickAngle;
152         if (Mathf.Abs(angleClickDiff) > HapticClickAngleStep)
153         {
```

```
154                 LastClickAngle = Pointer.CurrentParabolaAngleY;
155                 device.TriggerHapticPulse();
156             }
157         }
158 }
```