Conductor: Intersection-Based Bimanual Pointing and Placement in Augmented and Virtual Reality

ANONYMOUS AUTHOR(S)

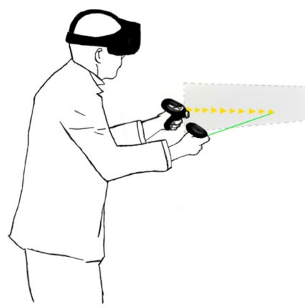


Fig. 1. Conductor uses a ray from the dominant hand and a virtual plane from the non-dominant hand to control a cursor.

Pointing – whether to select or to place an object – is an elementary interaction in virtual and augmented reality environments, and, to effectively support both selection and placement, techniques must deal with the challenges of occlusion and depth specification. In this paper, we propose Conductor, a plane-ray, intersection-based, 3D pointing technique where users leverage bimanual input to control a ray and intersecting plane. We evaluate Conductor against Raycursor, a state-of-the-art VR pointing technique, and show that Conductor outperforms Raycursor for pointing tasks and that, for both pointing and placement tasks, Conductor is a promising alternative to RayCursor.

CCS Concepts: • **Human-centered computing → Pointing**.

Additional Key Words and Phrases: VR selection, intersection, raycasting, VR placement

ACM Reference Format:

Anonymous Author(s). 2018. Conductor: Intersection-Based Bimanual Pointing and Placement in Augmented and Virtual Reality. In *CHI '21: The ACM CHI Conference on Human Factors in Computing Systems, May 08–13, 2021, Yokohama, Japan.* ACM, New York, NY, USA, 16 pages. https://doi.org/10.1145/1122445.1122456

1 INTRODUCTION

As virtual and augmented reality (VR and AR) headsets increase in resolution and responsiveness, increasingly immersive visual experiences are available to a broad set of end-users. However, one challenge with VR and AR is how best to interact within these environments [2]. Motivated by the increasing availability of cost-effective, high-performance

Manuscript submitted to ACM

virtual reality headsets, there exists a large and on-going body of research in interaction techniques to support target acquisition and target repositioning within these virtual environments [3].

Targeting within VR and AR environments is challenging because these environments, rather than being populated by a series of 2D targets as on a computer display, are populated by targets and objects of interest arranged in a 3D world where issues of depth variation and occlusion confound interaction. Recent research on interaction within these environments focuses, specifically, on how to support rapid, fluid targeting given partial [3] or full [10] occlusion within the 3D content contained within these spaces.

In this work, we explore a bimanual targeting technique, *Conductor*, which allows users to control position with one hand and depth with a second within virtual environments. Conductor was designed based upon two observations: 1) State of the art AR and VR systems normally support bi-manual tracking, either through controllers or some other capture mechanism; and 2) When performing unimanual targeting in virtual environments, users' non-targeting hand is typically not engaged in another task.

We acknowledge, at the outset of this paper, that there may be a role for unimanual targeting in virtual environments. In some situations, it may be that the user's non-targeting hand is engaged in another task. However, acknowledging that there is a role for unimanual targeting, we also posit that there *may* be a role for bimanual techniques, but whether there is or not is unclear. More specifically, when exploring related research, it became clear to us that, while many targeting techniques have been developed for VR/AR environments, including both unimanual [3] and bimanual [19] techniques, it is not clear what the relative performance of bimanual techniques, either existing or proposed, are compared to state-of-the-art unimanual targeting. Is there a role for bimanual targeting at all? Does it provide performance advantages over state-of-the-art unimanual techniques?

Posing these research questions at their most specific level, consider the following: A recent CHI paper presented a unimanual targeting technique, RayCursor [3], for virtual reality environments which effectively supports targeting in both sparse and dense virtual environments. Can we design a bimanual technique that has performance advantages over RayCursor? If so, then both RayCursor and a competing bimanual targeting technique can coexist (one for primarily unimanual contexts, another in contexts where bimanual targeting is acceptable). On the other hand, unless a bimanual technique demonstrates performance advantages over RayCursor, unimanual targeting will remain the default technique for virtual environments due to lower effort coupled with equivalent performance.

This paper presents our initial design work to realize a bimanual virtual environment selection technique, Conductor (Figure 1). It also describes two experiments that evaluate the performance of Conductor against the RayCursor technique [3], evaluating both target acquisition and and target placement [17] tasks, respectively, within a virtual reality environment. Overall, we find that Conductor outperforms RayCursor, both in task completion time and in input accuracy, particularly during target-aware selection tasks. This argues for the viability of Conductor as a viable alternative 3D selection technique to RayCursor, focused particularly on contexts where bimanual input via dual controllers is deemed a useful trade-off for enhanced targeting performance.

The remainder of this paper is organized as follows. We first review related work on targeting in virtual and augmented reality environments. Next, we describe the design of Conductor and experiments testing its efficacy for both targeting and 3D placement tasks. We conclude by discussing the implications of this work for interaction in virtual environments.

2 RELATED WORK

Our related work section focuses on 3D pointing techniques. We particularly explore techniques that handle occlusion, techniques that allow placement of objects, and related work on other intersection-based pointing techniques.

2.1 3D Selecting Technique

In their survey of virtual and augmented reality-based pointing techniques Argelaguet and Andujar [2] divided selection metaphors into two broad categories: virtual hand and virtual pointing. The virtual hand metaphor [8] uses a projected hand, typically located near the cursor, to support 3D object selection. In contrast, RayCasting [3] linearly projects a 3D ray oriented near the controller into the virtual environment in order to perform target selection.

The basic virtual hand technique is relatively straight forward to implement [8]. As in real-world interactions, users can target objects that are within arms’ reach; for more distant objects, users must first move near the objects before being able to select them. To address this shortcoming, techniques such as the Go-Go technique [11] combine some form of virtual hand with an extendable arm interaction. Specifically, within a restricted range, users leverage the virtual hand technique. However, when the hand moves further than a cutoff from the body, the hand’s depth, or distance from the user, is magnified via a CD Gain function to allow users to acquire more distant targets.

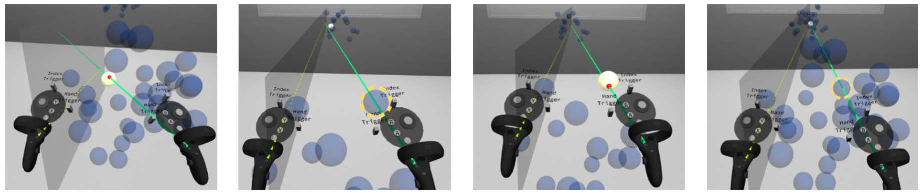
In contrast to the virtual hand technique, raycasting has been the focus of significant research regarding improved performance, primarily due to a desire to select distant objects in virtual environments. One challenge with selection in 3D virtual environments is that objects near the user appear larger (due to perspective) and can occlude objects of interest that are “behind” nearby objects. To address occlusion, techniques such as Outline Pursuits[14] enable users to trace a moving stimulus on the outline of objects with eye gaze to select occluded object. In their work on volumetric displays, Grossman and Balakrishnan [7] proposed four advanced raycasting techniques that allow users to select occluded object on the same ray. One of their techniques, depthray, mimics the Go-Go technique in that users move the controller either toward or away from themselves to select objects along the ray. Finally, RayCursor [3], a recently developed pointing technique presents significant performance benefits when targeting in densely populated virtual environments. It does this by combining the Raycasting metaphor with depth control (as in Grossman and Balakrishan [7]) via an additional touchpad located on some VR controllers.

2.2 3D Placement

In their work on 3D manipulation, Vuibert et al. [17] segment 3D object manipulation into three tasks: placement, where an object is positioned in 3D space; orientation, where an object’s yaw, pitch, and roll are adjusted; and docking, a 6dof task that combines both placement and manipulation. In this work, as our focus is primarily on target acquisition, we consider primarily 3D target acquisition and 3D placement as two components of 3D selection. Placement is, essentially, target agnostic spatial selection.

In 3D placement, Berard et al. [4] evaluated the mouse against 3dof input devices. Somewhat surprisingly, they found that the 2D computer mouse outperformed special purpose 3D controllers for 3D placement tasks. One challenge with generalizing their work to virtual environments is that they assume a user seated at a desk, interacting with 3D content. As virtual environments – augmented realities – become increasingly embedded into real world environments, users often must interact with 3D content when standing or moving.

Surprisingly, despite the significant body of past research exploring targeting in 3D spaces, raycasting techniques have been rarely evaluated for placement tasks. However, many of these techniques can be used in placement as easily



(a) (b) (c) (d)

Fig. 2. Selecting task with Conductor. (a)Conductor with close target. (b)A target on the ray. Manual mode Conductor selects the target closest to the intersection of the ray and the plane. (c)Semi-Auto mode Conductor selects the target on the ray. (d)Multiple targets on the ray. Semi-Auto mode Conductor selects the intersected target closest to the plane.

as in targeting. For example, Raycasting variants [7, 13], including RayCursor [3] typically have some indicator of depth attached to the ray, and this depth indicator can be freely manipulated to control position. Virtual hand techniques, including Go-Go [9, 12] also support placement in a relatively straightforward manner.

Our work in this paper explores bimanual selection and placement. We are not the first to explore this domain: different symmetric and asymmetric bimanual selection techniques have been for proposed for volumetric displays, in particular work by Ulinski et al. [15, 16]. However, Ulinski et al.'s research identified best practice in the design of bimanual selection techniques; the performance of these techniques in virtual environments in comparison to optimal unimanual techniques was not assessed.

2.3 Intersection-Based Pointing

In heavily occluded 3D spaces, some mechanism is needed to support targeting along varying depths. One such technique that has been proposed in the literature, and the technique that we leverage for conductor, is a technique that we label intersection-based pointing. Within this domain, tiltcasting[10] enables intersection-based selection by rotating a virtual plane in a 3d scene and selecting the target intersecting the virtual plane. While tiltcasting effectively supports 3D targeting and placement, its primary evaluation was on traditional 3D displays, rather than in virtual environments and head-mounted displays.

Specifically in the domain of virtual environments, and most similar to *Conductor*, the iSith [19] is an intersection-based pointing technique which uses two rays from emanating from the index fingers of each hand. The point that is the shortest distance between the two rays is considered the current cursor location. Like conductor, iSith shares the advantages of nearly unlimited range of interaction and supports easy selection of occluded objects. However, one challenge with iSith is that, if the pitch axes of two rays differ, the cursor can drift away from the visual intersection of two rays. Furthermore, while iSith has been proposed as an intersection-based, bimanual pointing technique, its performance against competing unimanual techniques has, to the best of our knowledge, not been evaluated [19].

3 CONDUCTOR

We describe the design of Conductor, a plane-ray, intersection-based, bi-manual pointing technique. A ray and a plane are manipulated by two 6 DOF controllers. The ray is controlled by the dominant hand and the plane by the non-dominant hand. Users direct the ray to point at a desired target and control the depth by moving the plane with the non-dominant hand. Conductor exists in three variants: a manual variant; a semi-automatic variant; and a semi-automatic with refinement variant, distinguished as follows:

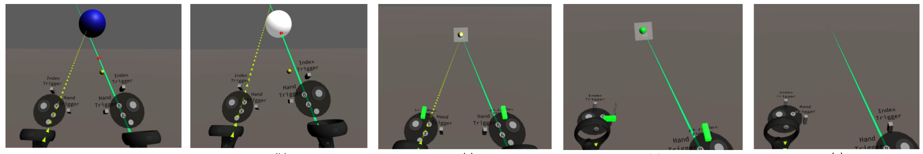


Fig. 3. Placement task with Conductor. (a)Initial status (b)Use Semi-Auto Conductor to select target (blue ball) (c)Quick Access by pressing the non-dominant index trigger and moving or rotating two hands. There are two white semi-transparent rectangles on the ray to indicate the range that they can refine in the next step. (d)Keep pressing the non-dominant hand (palm) trigger and move the non-dominant hand forward or backward to move the target along the ray correspondingly.(e)Release dominant index trigger to confirm

- *Manual Conductor:* Manual conductor selects the target closest to the intersection of the plane and the ray (Figure 3 (a)(b)). In the case of Figure 3 (b), there is a target on the ray close to the right controller. However, since there are targets closer to the intersection, the target closest to the intersection but not on the ray will be selected. Target selection is confirmed via the dominant hand trigger.
- *Semi-Automatic Conductor:* Semi-Automatic Conductor combines the benefits of Conductor and Raycasting. If any single target exists on the ray, that target will be selected (Figure 3 (c)). If multiple targets exist on the ray, the target closest to the intersection of plane and ray will be selected(Figure 3 (d)). Users can do precise Raycasting with their dominant hand controller and manipulate the non-dominant hand plane to approach the target at the same time. This strategy addresses issues of occlusion. Users can also short circuit the semi-automatic refinement and select the target closest to the intersection, as in Manual Conductor, via the non-dominant index trigger if necessary.
- *Semi-Automatic Conductor with Refinement:* Semi-Automatic Conductor with Refinement extends Semi-Automatic Conductor by including a refinement strategy for a more precise control of cursor depth. One problem with Conductor is that the precision of depth specification on distant objects is reduced due to small angular deviations creating large changes in depth. The user can depress the non-dominant hand trigger with their middle finger and move the non-dominant hand forward and backward to refine the cursor position along the ray. The length of the cursor movement equals the length of hand movement times the current cursor distance to the dominant hand (start point of the ray). This makes visually perceived movement similar regardless of depth. Any other cursor movement is blocked when the refine mode is engaged, so users can first quickly access a position that is close to the destination, and then use refinement to precisely target.

To provide visual feedback for the target location, a dashed yellow line (see Figure 3) emanates from the non-dominant hand controller along the intersecting plane to the ray. As well, as with Raycursor, to support smooth ray and plane movement, we used the 1€ Filter[5] with parameters set as mincutoff = 0.1 and β = 50 to smooth the ray for targeting. These parameters were identical to those used in RayCursor[3].

4 EXPERIMENT1: TARGET SELECTION TASK

Our first experiment aims to compare the selection performance of RayCursor’s variants [3] to Conductor’s variants. We evaluate 4 techniques in this task: Manual RayCursor, Semi-Auto RayCursor, Manual Conductor, and Semi-Auto Conductor.

4.1 Participants

8 participants (6 male, 2 female) were recruited from the local university and the community. All were right handed. Seven of them were run in person and one was run remotely. Five of them experienced VR for the first time. Each participant received $10 as remuneration.

4.2 Apparatus

We use a replication of the study conducted in Baloup et al. [3] with minor differences in hardware. The experiment used a PC Intel Core i7-9700K PC with NVIDIA RTX-2080 Ti GPU and an Oculus Rift s VR headset. The application was programmed in the C# language with Unity3D Version 2019.4.7. Participants manipulated a pair of Oculus Touch controllers with a joystick on each. For the RayCursor techniques, users used one controller in their dominant hand. The users used the joystick on the controller to control the depth of the cursor along the ray. For the Conductor techniques, users controlled the position and rotation of two controllers together to manipulate the ray and plane. Target selection was performed by pressing the dominant controller's trigger using the index finger.

In the original Raycursor experiments, the VR controllers were equipped with touchpads. However, Baloup et al. [3] note that Raycursor performs well regardless of whether the controller includes a touchpad or joystick. Because the Oculus Touch controllers are equipped with joysticks instead of touchpads, we set new parameters for Raycursor as per Baloup et al. to fit the joystick: k1 = 0.6, k2 = 3, v1 = 0.3, v2 = 0.9. We preserved the scale of k1/k2 and v1/v2 in original RayCursor paper to maintain the CD gain of the original implementation. We performed an initial pilot evaluation of this Raycursor variant to ensure consistent performance with the original Raycursor implementation prior to our experiments.

4.3 Methodology

We replicated the environment and task in RayCursor [3] with minor modifications to create additional occlusion. The 4 techniques were balanced by Latin square between subjects. Following the RayCursor task protocol, we also used 2 target SIZES and 2 target DENSITIES. The 2 target sizes were 8cm (= S_{Big}) and 4cm (= S_{Small}) in diameter and the 2 target densities were 60 targets (= D_{High}) and 30 targets (= D_{Low}).

Again, identical to the RayCursor protocol, all the targets were distributed inside 2 spheres of 60cm diameter randomly in front of the participant. The closer sphere was 1m away from the participant and the farther was 4m. We raised the height of two centres from 80cm to 140cm so that the closer group of targets could create occlusion when selecting from the farther group of targets, especially for the condition with D_{High} and S_{Big}. We also chose the farthest nine targets in each group for users to select to build additional within-group occlusion. All targets were semi-transparent blue and the target to be selected was solid yellow. Participants were required to select the targets alternating between the closer group and the farther group. When the participant selected the correct target, the selected target would flash green and the controller(s) vibrated for 10ms. If incorrect target is selected, the selected target would flash red and the controllers would vibrate for 200ms to indicate the erroneous target. The trial would be marked as error, and the participant was required to reattempt selection until it was correct. There was a virtual scoreboard in the front of the participant behind the targets indicating the current error rate, and the participants were asked to keep an error rate below 4% to balance accuracy and speed.

Each participant watched a series of short videos introducing the technique before beginning each technique. They could experiment with the technique during the instruction videos, and started one practice run with S_{Big} and D_{High} to familiarize themselves with the experiment.

Then they started the experiment with 3 blocks of the 4 counterbalanced (via a perfect Latin square) techniques combining different sizes and targets (D_{Low} + S_{Big}, D_{Low} + S_{Small}, D_{High} + S_{Big}, D_{High} + S_{Small}). Participants were allowed to rest between runs. Each combination of technique × size × density contained 9 trials. In summary, this yields 4 TECHNIQUES × 2 DENSITIES × 2 SIZES × 9 trials × 3 blocks = 432 data points per participant.

After participants completed the selection task for each technique, the participants were asked to complete the NASA Task Load Index (NASA-TLX) questionnaire [1] and respond to four additional questions on similar 20-point scales. They filled the questionnaire using their headset and controller. The study took about 30 minutes.

4.3.1 Hypotheses. We wish to evaluate the performance of our bimanual Conductor technique against Raycursor. Therefore our primary hypothesis is that Bimanual Conductor outperforms similar Raycursor variants in time and error rate. More specifically:

- H1: Semi-Auto Conductor outperforms semi-auto Raycursor in time and error.
- H2: Manual Conductor outperforms manual Raycursor in time and error.
- H3: Participants rate Conductor higher than Raycursor in NASA TLX and custom questionnaire measures.

4.4 Results

To analyze the data, we used a repeated measures three-way ANOVA with TECHNIQUE, DENSITY and SIZE as independent variables. Holm-Bonferroni corrected post hoc pairwise t-tests are used for further analysis. When sphericity was violated, degrees of freedom were corrected using Greenhouse-Geisser ($\epsilon < 0.75$).

4.4.1 Task Completion Time. We removed outlier data points with selection times more than 3 standard deviations from the task mean. This removed 58 data points (1.7%) in total. Figure 4 shows task completion time for target selection. There was a significant main effect for TECHNIQUE on task completion time ($F_{3,21} = 23.0$, $p < .001$). Overall, the Conductor techniques (SA_Condoctor and manual Conductor) were faster than their corresponding RayCursor variants. Focusing, specifically, on the semi-automatic techniques, semi-automatic Conductor(mean = 1.51 sec) was 22% faster than the Semi-Auto RayCursor technique (SA_RayCursor) (mean = 1.93 sec, p<0.001). Manual Conductor (mean = 1.97 sec) had a similar advantage over manual RayCursor (mean = 2.79 sec, p<0.001), and performed approximately on-par with semi-automatic RayCursor (mean = 1.93 sec vs 1.97 sec, ns).

Although there was no significant main effects for DENSITY nor SIZE, significant interaction effects of TECHNIQUE × DENSITY ($F_{3,21} = 12.1$, $p < .01$) and TECHNIQUE × SIZE ($F_{3,21} = 13.6$, $p < .001$) were found. Figure 5 shows average task completion time for different sizes of targets. A post hoc test revealed there were significant difference on task completion time between D_{Low} and D_{High} with the Semi-Auto RayCursor technique(p<.001) and between S_{Big} and S_{Small} with the Semi-Auto RayCursor technique (p<.001) and Semi-Auto Conductor technique (p<.001). As shown, Semi-Auto Conductor performed better with larger targets than smaller targets. However, with Semi-Auto RayCursor, the larger targets slowed selection because of occlusion. For the same reason, with a higher density of targets, the selection speed with Semi-Auto RayCursor slows. This result indicates that the Semi-Auto RayCursor technique is more prone to occlusion whereas Semi-Auto Conductor is more resilient. However, note that, overall, for selection, semi-auto Conductor outperforms semi-auto Raycursor by a statistically significant margin.

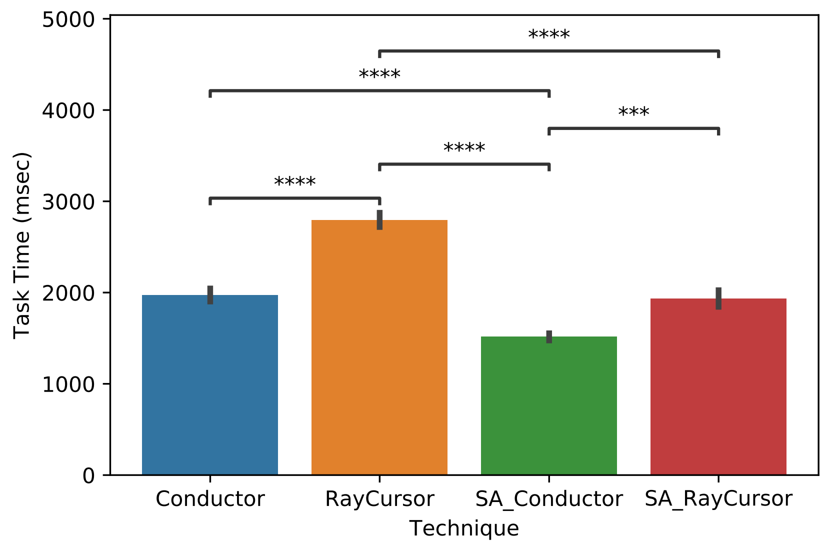


Fig. 4. Task completion time for target selection task. Error bars are 95% CI.

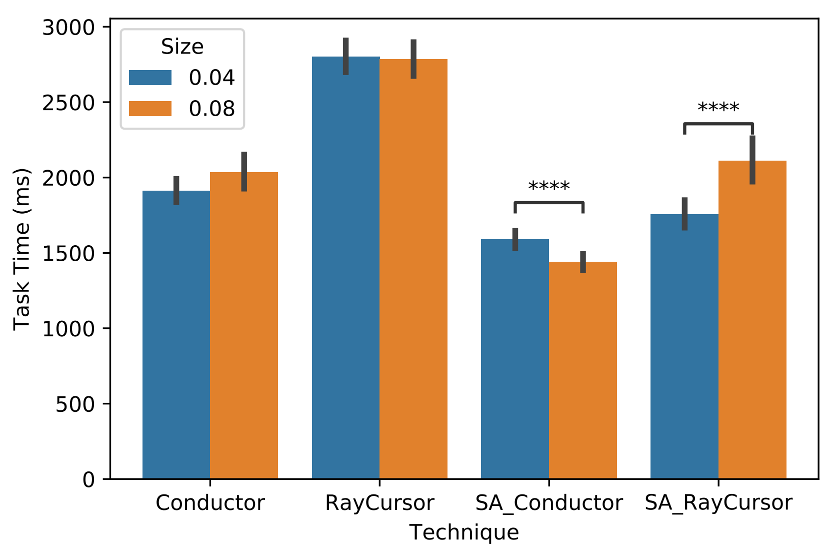


Fig. 5. Task completion time for different target sizes and techniques. Error bars are 95% CI.

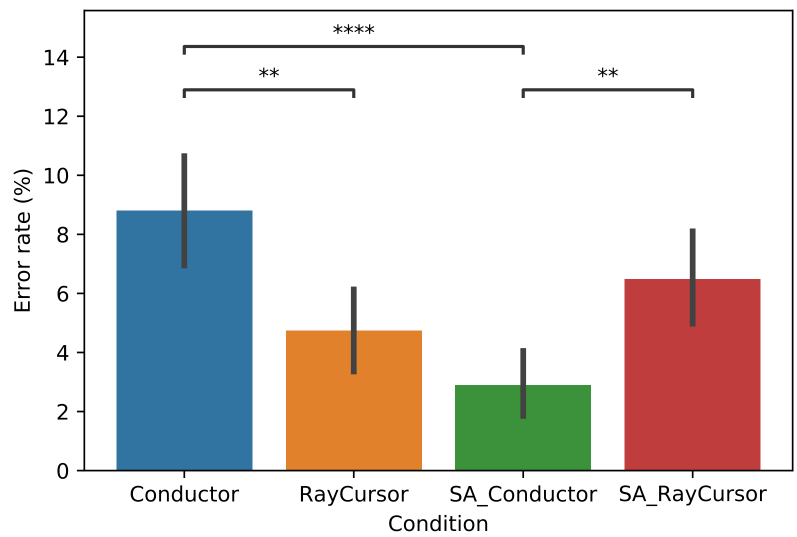


Fig. 6. Error rate for target selection task. Error bars are 95% CI.

Alongside our pilot study to test performance, our results provide an additional opportunity to contrast our Raycursor implementation to the original implementation of Baloup et al. [3]. The average selection time of our implementation of RayCursor(mean = 2.79 sec) and Semi-Auto RayCursor (mean = 1.93 sec) in our study were virtually identical (within 50ms) of the original Raycursor implementation in Baloup et al.'s studies (mean = 2.72 sec and 1.88 sec, respectively).

4.4.2 Error rate. A three-way ANOVA revealed significant main effect for TECHNIQUE($F_{3,21} = 8.4$, $p < .001$) and SIZE($F_{1,7} = 9.7$, $p < .05$) on error rate. There was no interaction effect. Figure 6 shows error rate for each technique. With all four techniques, the error rate was less than 10%. The error rate of Semi-Auto Conductor was the lowest (2.9%) and it was significantly lower than Semi-Auto RayCursor (6.5%) (p=0.002) and both manual techniques. Manual Conductor had the highest error rate, but, interestingly, while manual Raycursor outperformed manual Conductor by a statistically significant margin, semi-automatic RayCursor did not outperform manual conductor.

The error rate of RayCursor(4.7%) and Semi-Auto RayCursor(6.5%) in our study were slightly higher than the original study results (3.8% and 2.4%). We assume this was the result of higher occlusion of the targets due to a slight increase in target density.

4.4.3 Questionnaire. The NASA-TLX result did not indicate a significant difference between Semi-Auto RayCursor and Semi-Auto Conductor techniques nor between the manual variants of the techniques. However, our additional questions indicate participants had a more positive impression of the Semi-Auto Conductor technique compare to other techniques. We identified no statistically significant differences between the manual variants. Figure 7 shows the additional questions and their result. As with the NASA-TLX, the additional questions were on a 20-point Likert scale where left is "Strongly Agree" and right is "Strongly Disagree".

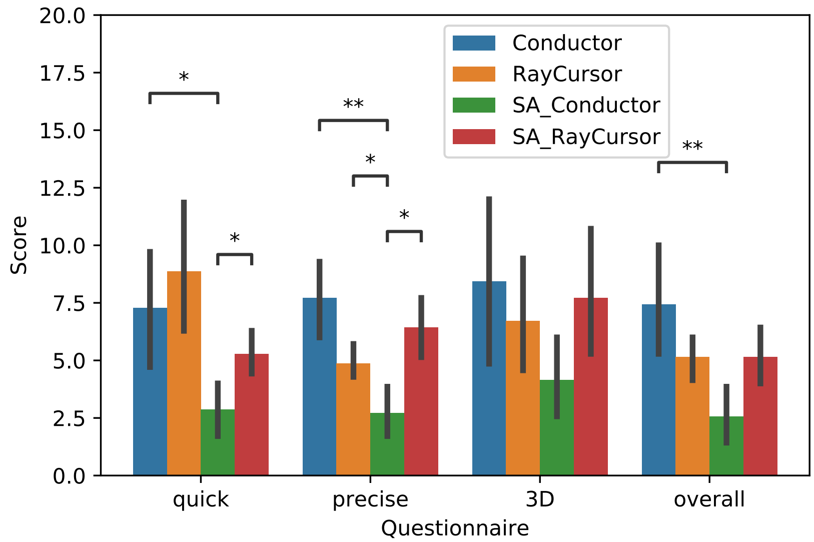


Fig. 7. Additional questionnaire results for each technique. The questions were as follows; "quick": I can control it quickly., "precise": I can control it precisely., "3D": I can feel that I'm controlling the cursor in all three dimensions., "overall": Overall, I like this condition. Error bars are 95% CI.

4.4.4 Synthesis. Considering our initial hypotheses for this experiment, we find that H1 is supported, the semi-automatic variant of Conductor, has performance advantages over the semi-automatic variant of Raycursor. However, results for H2 and H3 are more mixed. Speed advantages for the manual variant of Conductor are offset by a higher error rate. As well, questionnaire results are mixed, though responses do provide evidence that Conductor is no worse than Raycursor in terms of user preference and cognitive load.

Given these mixed results, we turn to a second experiment to focus more directly on object placement tasks with Raycursor and Conductor.

5 EXPERIMENT2: PLACEMENT TASK

Examining the results of our first study, while the semi-automatic variant of Conductor appears to outperform the semi-automatic version of Raycursor, a careful analysis of manual variants tells a more mixed story. While selection is significantly faster for manual conductor than manual Raycursor, the error rate of manual conductor is statistically significantly higher than manual Raycursor, and, more generally, than all other techniques except semi-automatic Raycursor. Semi-automatic techniques speed targeting by taking a target-aware approach to selection along a cast ray. This, then, leads to the question of whether or not Raycursor techniques might outperform Conductor techniques in a target agnostic selection task. For this reason, we perform a second experiment to contrast Raycuror variants and Conductor variants in a target agnostic way.

Therefore, this second experiment aims to assess the controllability of both the RayCursor [3] and Conductor techniques in a 3D target placement task [18]. 3D target placement requires participants to move an object to another location within the 3D environment. The desired location to which to move the target is specified in the task; however,

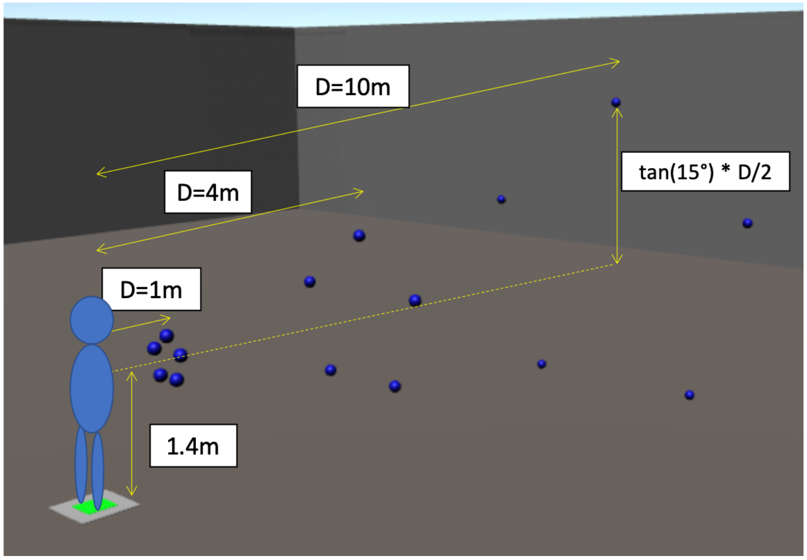


Fig. 8. The target/destination coordination for the placement task.

unlike in the selection task, the target location cannot be selected automatically; the participant must precisely control object depth and position to place the indicated object within the target location.

5.1 Participants

9 participants (8 male, 1 female) were recruited from our local university and internet. All were right handed. One were experiencing VR for the first time. Each participant received $20 as remuneration. Three of the participants did the study in person and six completed the study remotely.

5.2 Apparatus

Due to research ethics rule changes during this research[1], equipment transfer between locations and in-person experiments were halted and remain prohibited, requiring six participants to be run remotely. As a result the experiment used a PC powered by Intel Core i7-9700K CPU and NVIDIA RTX-2080 Ti GPU for those who did the study in-person and participants' own personal equipment for those who did the study remotely. The study was performed using either an Oculus Rift s VR headset or Oculus Quest with Oculus Link connected to the corresponding PC. The application was programmed in the C# language with Unity3D Version 2019.4.7.

5.3 Methodology

Techniques (2) were counter-balanced across participants. Three distances in the z-axis were used including Z_{Near}(1m), Z_{Middle}(4m), Z_{Far}(10m), and the target and destination at any of the three distances such that all combinations of distances, i.e. 3×3 = 9 distance combinations were tested. The target was a blue ball of 16cm in diameter and the destination was a semi-transparent yellow ball with a diameter of 32cm.

[1]This experiment was conducted early during the Covid-19 pandemic.

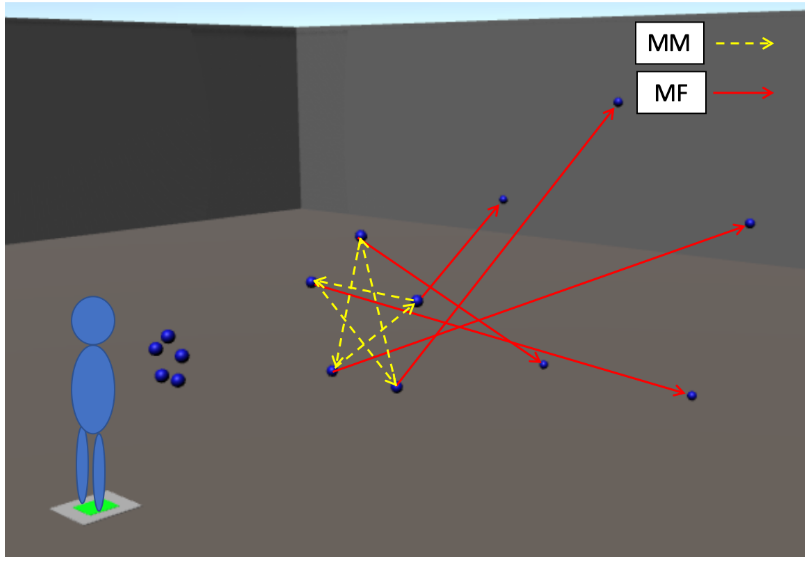


Fig. 9. Example directions of the placement task. Yellow dotted lines represent the MM trials and solid red lines show the MF trials.

We designed a more controlled task for the target placement task. For our task, the x and y axes for the target and destination were assigned using a modified ISO 9241-411 multi-directional (2D) Fitts's Law task with a diameter of $\tan(15^{\circ}) \times$ distance in z-axis of the target. Figure 10 depicts the target and destination location of our placement task. Figure 9 shows examples of the placement directions. Five trials at each distance were required to balance the 2D directions based on our modified ISO task.

During the experiment, participants began by selecting a white ball located 1m in front of them. Once they selected this start target, a target and destination was shown. The participant moved to select the target, and then dragged to reposition the target to be fully inside the destination, at which point the destination would turn green. A new target to acquire was then displayed along with a destination location, and the experiment would continue. Destination locations varied in position (oscillating around the circular arrangement of targets as in the ISO 9241-411 Fitts's Law task) and in depth, and participants had to control both position and depth to complete the placement task. The duration was counted from the time that participant selected the target to be moved to the time they released the target fully inside the destination. To keep the study within a reasonable time, we provided an option to skip a trial. 15 seconds after the participant selected the target for the trial, a prompt "Now you can press B to skip if you want" was displayed for one second.

To speed study time, the actual techniques used were a hybrid of semi-automatic and manual techniques. To aid in selecting the target to be repositioned, we use semi-automatic enhancements to speed selecting the initial target. However, once dragging the target to be repositioned, the techniques were fully manual (i.e. no depth-based nor target assistance was available because is it typically impossible to infer, a priori, the desired target location during a free target placement task). RayCursor and Conductor with Refinement were the two specific techniques used to compare performance. Our choice of incorporating refinement in Conductor was motivated by the complexities of the target placement task, where minute refinements were required to “nudge” targets into target locations.

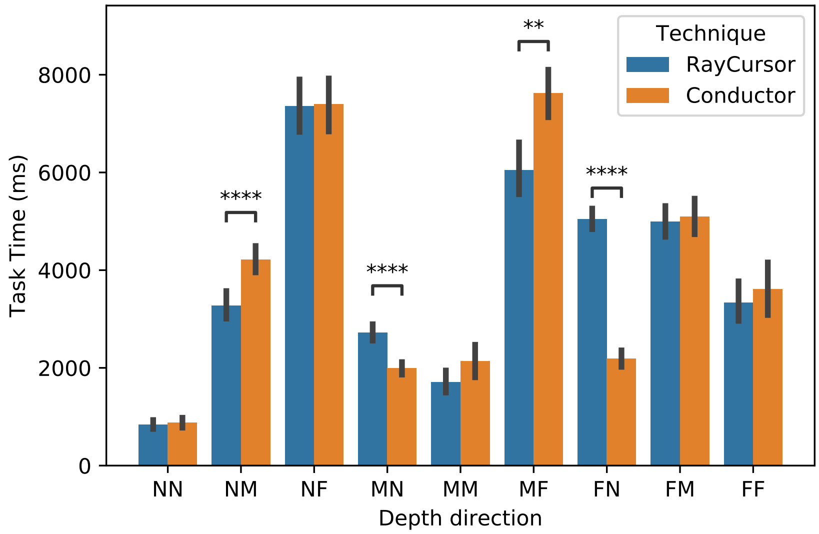


Fig. 10. Task completion time for each depth direction. N, M, L represents Z_{Near}(1m), Z_{Middle}(4m), Z_{Far} respectively. The first letter represents the start depth and the next letter represents the end depth. For example, "NM" means the target locates at distance Z_{Near} in z axis and the destination is Z_{Middle} in z axis. Error bars are 95% CI.

As in the previous study, each participant watched videos introducing them to the general task and each technique, and then began the experiment. There was also a demo destination and a demo target used for training during the instructional video. Each time the demo target was successfully docked, the demo target would return to the original position and the demo destination would change in a near-middle-far sequence. Users could practice until confident with the technique.

Once comfortable, participants started with two practice blocks with all depth changes (9 trials), and then completed 2 experimental blocks with the 9 depth change conditions (3 start depth $\times$ 3 end depth). Each condition contained 5 trials as noted earlier. In summary, there were total 2 TECHNIQUES $\times$ 3 START_DEPTH conditions $\times$ 3 END_DEPTH conditions $\times$ 5 trials $\times$ 2 blocks = 180 data points per participant.

The study took about 60 minutes.

5.4 Results

One participant's data was excluded from the analysis due to high skip rate (18%). The average skip rate of other participants was less than 1%. This resulted in eight participants' data included in analysis, with counterbalancing of techniques preserved. To analyze the data, we used a repeated measures three-way ANOVA with TECHNIQUE, START_DEPTH and END_DEPTH as independent variables. Holm-Bonferroni corrected post hoc pairwise t-tests were used for subsequent analysis. When sphericity was violated, degrees of freedom were corrected using Greenhouse-Geisser ($\epsilon < 0.75$).

5.4.1 Task Completion Time. We removed outlier data points with selection times of more than 3 standard deviations from the task mean. This removed 23 data points (1.6%) in total.

The average task completion time was 4.00 second for Conductor and 4.05 second for RayCursor. There was a significant main effect for END_DEPTH on task completion time ($F_{2,14} = 332.8$, $p < .001$) but no main effect for TECHNIQUE. We also found significant interaction effects of TECHNIQUE × START_DEPTH ($F_{2,14} = 25.7$, $p < .001$), TECHNIQUE × END_DEPTH ($F_{2,14} = 24.0$, $p < .001$), START_DEPTH × END_DEPTH ($F_{4,28} = 123.3$, $p < .001$), and TECHNIQUE × START_DEPTH × END_DEPTH ($F_{4,28} = 6.8$, $p < .05$).

Figure 10 shows task completion time for each depth direction. Post hoc tests found significant difference between RayCursor and Conductor with depth direction NM ($p < .001$), MN($p < .001$), MF($p < .01$) and FN($p < .001$). As we can see in the Figure 10, Conductor performed well when the target destination is Z_{Near}; particularly for FN, the mean task time of Conductor (2.19 sec) was less than half of RayCursor(5.08 sec). However, Raycusor performed better when moving NM and MF, but not NF.

5.4.2 Questionnaire. No significant main effect on any of TECHNIQUE, START_DEPTH and END_DEPTH for questionnaire results was found.

6 DISCUSSION

Combining results from our experiments, we note that, for target selection tasks, our bimanual technique, Conductor, outperforms Raycursor by a statistically significant margin. However, for the target placement task, where depth must be manipulated, statistical advantages were not found and overall performance was similar (averages differed by approximately 50ms). The overall implications of this are as follows:

- For target aware selection tasks, if VR application constraints indicate that bimanual selection is permissible, then evidence exists that Conductor provides a throughput advantage (faster selection with lower error) over Raycursor for semi-automatic variants. In manual cases, throughput advantages may be more mixed as higher speed may be offset by an increase in error rate.
- For the target placement task, advantages of Conductor over Raycursor were not immediately apparent. However, if target placement involves selecting a target and then moving it to a new location (as in our implementation), Conductor provides overall benefit due to its faster initial target selection. However, if, instead, the goal is to place the cursor at an arbitrary location in space (e.g. for sketching in VR [6], for example, where no initial target acquisition is needed), advantages are not apparent.

Combining the two results above, if rapid targeting is desired in virtual environments (which is the premise of our work, Raycursor, and many other proposed depth and occlusion resistant interaction techniques [3, 7, 11, 15]), then Conductor can be a viable selection technique for incorporation into virtual environments. This is not to say that Conductor should replace Raycursor in implementations. In some instances, for example due to the presence of only one controller or encumbrance during interaction of the non-dominant hand controller, unimanual selection techniques such as Raycursor are needed. However, in many instances during target selection and placement, the non-dominant controller is at rest with no assigned task. In these contexts, leveraging a bimanual technique to enhance throughput (via higher speed and lower error) may be desirable.

6.1 Future Work and Limitations

We invested some effort in diagnosing challenges with Conductor in our second target placement experiment. One challenge seemed to be the sensitivity of the plane when positioned at far locations. Small variations in angle resulted in large depth displacements due to paralax. One area of future work that we are exploring draws inspiration from the Go-Go technique [11] to control planar angle during input. Specifically, for "near" interactions, the CD Gain of angular rotation to control the plane can be constant. However, as the plane moves "farther" from the user, beyond a specified range, we can reduce the z-speed of the plane and allow the user to rotate their wrist more to achieve depth changes. In initial pilot studies, this "angular remapping" of controller rotation to planar displacement for distant plane locations appears to provide advantages for precision in distant input.

Beyond tuning parameters, we currently use VR-based controllers in our input. However, other controllers including smartphone-in-hand/smartwatch-on-wrist interactions can be leveraged using on-board device inertial measurement units (IMUs) to perform bimanual targeting. Both Conductor and Raycursor can be easily implemented for use with commodity personal devices. In our current experiment, we did not explore this option due to our desire to replicate the Raycursor study for appropriate contrast with Raycursor. However, future work contrasting the performance of less precise, personal-device-based, IMU tracking is planned.

Finally, considering limitations, we note two. First, unlike in the initial Raycursor implementation, where touchpad equipped controllers were used, our controllers incorporated joysticks. To control for this, we piloted to ensure equivalent performance and we also contrast our performance during experiment one to performance values reported by Baloup et al. [3] in their published work. Given that our performance is near equivalent (within 50ms), we would argue that our results are representative of touchpad implementation performance. Furthermore, understanding performance on both touchpad and joystick controllers is valuable due to the fact that both are commonplace alternatives for VR control.

Second, due to the global health context of experiment two, some of our participants had to perform the experiment remotely using their own equipment. While this is less ideal than a lab-based environment for careful experimental control, one advantage to VR-based experimentation is that virtual environments can be leveraged to control what is seen by participants, and controllers have similar performance across a broad range of contexts. While this is a potential limitation on the experimental validity of our results in study two, the increase in ecological validity undoubtedly trades off as an advantage against this potential limitation.

7 CONCLUSION

In this paper, we present Conductor, a bimanual technique for target selection in virtual environments. Through an initial carefully replicated experiment, we show that Conductor provides statistically significant performance enhancements during target-aware selection tasks compared to Raycursor, a state-of-the-art unimanual target selection technique for virtual environments. We also contrast Conductor and Raycursor via a second, 3D placement experiment, and argue that, in contexts where targets must be selected and repositioned, Conductor provides performance benefits due to its initial targeting speed. Together, these results argue for the potential value of Conductor as a competing targeting technique in those environments where fast targeting is desired and bimanual targeting is a feasible option during interaction.

REFERENCES

[1] [n.d.]. NASA-TLX Questionnaire. https://humansystems.arc.nasa.gov/groups/TLX/.

[2] Ferran Argelaguet and Carlos Andujar. 2013. A survey of 3D object selection techniques for virtual environments. *Computers & Graphics* 37, 3 (2013), 121–136.

[3] Marc Baloup, Thomas Pietrzak, and Géry Casiez. 2019. Raycursor: A 3d pointing facilitation technique based on raycasting. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*. 1–12.

[4] François Bérard, Jessica Ip, Mitchel Benovoy, Dalia El-Shimy, Jeffrey R Blum, and Jeremy R Cooperstock. 2009. Did "Minority Report" get it wrong? Superiority of the mouse over 3D input devices in a 3D placement task. In *IFIP Conference on Human-Computer Interaction*. Springer, 400–414.

[5] Géry Casiez, Nicolas Roussel, and Daniel Vogel. 2012. 1€ filter: a simple speed-based low-pass filter for noisy input in interactive systems. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. 2527–2530.

[6] Michael F. Deering. 1995. HoloSketch: A Virtual Reality Sketching/Animation Tool. *ACM Trans. Comput.-Hum. Interact.* 2, 3 (Sept. 1995), 220–238. https://doi.org/10.1145/210079.210087

[7] Tovi Grossman and Ravin Balakrishnan. 2006. The design and evaluation of selection techniques for 3D volumetric displays. In *Proceedings of the 19th annual ACM symposium on User interface software and technology*. 3–12.

[8] Mark R. Mine, Frederick P. Brooks, and Carlo H. Sequin. 1997. Moving Objects in Space: Exploiting Proprioception in Virtual-Environment Interaction. In *Proceedings of the 24th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '97)*. ACM Press/Addison-Wesley Publishing Co., USA, 19–26. https://doi.org/10.1145/258734.258747

[9] Mark R Mine, Frederick P Brooks Jr, and Carlo H Sequin. 1997. Moving objects in space: exploiting proprioception in virtual-environment interaction. In *Proceedings of the 24th annual conference on Computer graphics and interactive techniques*. 19–26.

[10] Krzysztof Pietroszek, James R Wallace, and Edward Lank. 2015. Tiltcasting: 3D interaction on large displays using a mobile device. In *Proceedings of the 28th Annual ACM Symposium on User Interface Software & Technology*. 57–62.

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

[12] Ivan Poupyrev, Mark Billinghurst, Suzanne Weghorst, and Tadao Ichikawa. 1996. The go-go interaction technique: non-linear mapping for direct manipulation in VR. In *Proceedings of the 9th annual ACM symposium on User interface software and technology*. 79–80.

[13] Hyocheol Ro, Seungho Chae, Inhwan Kim, Junghyun Byun, Yoonsik Yang, Yoonjung Park, and Tackdon Han. 2017. A dynamic depth-variable ray-casting interface for object manipulation in ar environments. In *2017 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*. IEEE, 2873–2878.

[14] Ludwig Sidenmark, Christopher Clarke, XueSong Zhang, Jenny Phu, and Hans-W. Gellersen. 2020. Outline Pursuits: Gaze-assisted Selection of Occluded Objects in Virtual Reality. *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems* (2020).

[15] Amy Ulinski, Catherine Zanbaka, Zachary Wartell, Paula Goolkasian, and Larry F Hodges. 2007. Two handed selection techniques for volumetric data. In *2007 IEEE Symposium on 3D User Interfaces*. IEEE.

[16] Amy C Ulinski, Zachary Wartell, Paula Goolkasian, Evan A Suma, and Larry F Hodges. 2009. Selection performance based on classes of bimanual actions. In *2009 IEEE Symposium on 3D User Interfaces*. IEEE, 51–58.

[17] Vanessa Vuibert, Wolfgang Stuerzlinger, and Jeremy R. Cooperstock. 2015. Evaluation of Docking Task Performance Using Mid-Air Interaction Techniques. In *Proceedings of the 3rd ACM Symposium on Spatial User Interaction* (Los Angeles, California, USA) *(SUI '15)*. Association for Computing Machinery, New York, NY, USA, 44–52. https://doi.org/10.1145/2788940.2788950

[18] Vanessa Vuibert, Wolfgang Stuerzlinger, and Jeremy R Cooperstock. 2015. Evaluation of docking task performance using mid-air interaction techniques. In *Proceedings of the 3rd ACM Symposium on Spatial User Interaction*. 44–52.

[19] Hans Peter Wyss, R. Blach, and M. Bues. 2006. iSith - Intersection-based Spatial Interaction for Two Hands. *3D User Interfaces (3DUI'06)* (2006), 59–61.