Eye Gaze-based Object Rotation for Head-mounted Displays

Chang Liu
liu.chang@lab.ime.cmc.osaka-u.ac.jp
Osaka University
Suita, Osaka, Japan

Jason Orlosky
Osaka University
Suita, Osaka, Japan

Alexander Plopski
University of Otago
Dunedin, New Zealand

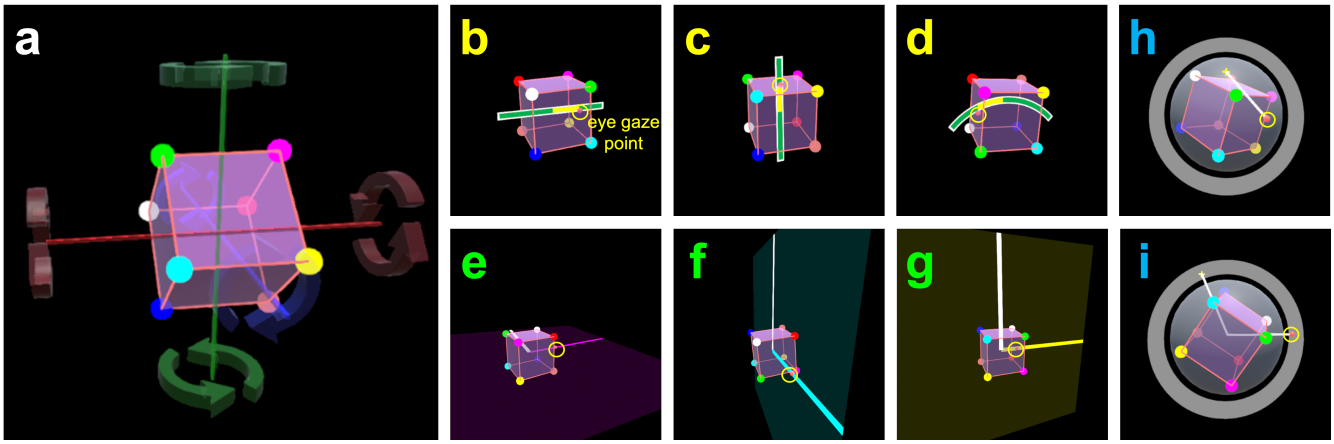

Figure 1: Images showing the three methods we developed to handle eye gaze-based rotations of virtual objects on head-mounted displays. a) The initial interface for selecting an axis by which to rotate. b)–d) Examples of RotBar, which maps per-axis rotations onto one-dimensional eye gaze movements. e)–g) Examples of RotPlane, which makes use of orthogonal planes to achieve per-axis angular rotations. h) i) Examples of RotBall, which combines a traditional arcball with a ring for handling user-perspective roll (z-axis) rotations. The yellow circle highlights the eye gaze point in the figures, but is not rendered during use.

ABSTRACT

Hands-free manipulation of 3D objects has long been a challenge for augmented and virtual reality (AR/VR). While many methods use eye gaze to assist with hand-based manipulations, interfaces cannot yet provide completely gaze-based 6 degree-of-freedom (DoF) manipulations in an efficient manner. To address this problem, we implemented three methods to handle rotations of virtual objects using gaze, including RotBar: a method that maps line-of-sight eye gaze onto per-axis rotations, RotPlane: a method that makes use of orthogonal planes to achieve per-axis angular rotations, and RotBall: a method that combines a traditional arcball with an external ring to handle user-perspective roll manipulations. We validated the efficiency of each method by conducting a user study involving a series of orientation tasks along different axes with each method. Experimental results showed that users could accomplish single-axis orientation tasks with RotBar and RotPlane significantly faster and more accurate than RotBall. On the other hand for multi-axis orientation tasks, RotBall significantly outperformed RotBar and RotPlane in terms of speed and accuracy.

SUI '20, October 31-November 1, 2020, Virtual Event, Canada

ACM ISBN 978-1-4503-7943-4/20/10…$15.00
https://doi.org/10.1145/3385959.3418444

CCS CONCEPTS

• **Human-centered computing** → **Graphical user interfaces**; *Interaction techniques.*

KEYWORDS

eye gaze, object rotation, head-mounted display, user interface

ACM Reference Format:
Chang Liu, Jason Orlosky, and Alexander Plopski. 2020. Eye Gaze-based Object Rotation for Head-mounted Displays. In *Symposium on Spatial User Interaction (SUI '20), October 31-November 1, 2020, Virtual Event, Canada.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3385959.3418444

1 INTRODUCTION

In recent years, head-mounted displays (HMDs) have made an increasing appearance as a consumer product. While various interaction methods are available such as controllers, gestures, and voice control, users still lack a private, hands-free method for manipulating virtual objects in 3D space. Tasks such as annotation placement,

3D modeling, and CAD simulation often require translations, rotations, and scaling operations. While hand or controller gestures are the gold standard for these manipulations, users may have their hands occupied by other tasks, or feel embarrassed when performing these actions in public that prevent the use of controllers or certain gestures. As such, many users need access to completely hands-free methods for performing these tasks in AR, VR, and other virtual object manipulation tasks.

To assist with certain actions, eye and head gaze have been seen as an unobtrusive alternative means of selecting objects in 2D and 3D user interfaces [14, 15, 26]. As eye gaze can serve as a quick on-screen pointing cursor [28], it has often been coupled with other means of manipulation to improve accuracy and efficacy. For example, users can use the eye gaze to quickly point at target objects, and then perform further manipulation through hand gestures [14, 24, 25]. However, manipulations using gaze have traditionally been very difficult due to the Midas touch problem [5] that refers to unintentional triggering of interactions due to misclassification of natural eye gaze movements. Controllers can help solve this problem since the hands and subsequent pointing are decoupled from gaze during interaction, and consequently almost all gaze-based research uses gaze as a support mechanism rather than primary interaction. Moreover, methods that can exclusively use gaze for manipulation are scarce. In our previous work [9], we made some progress in this direction by exploring the use of eye and head gaze for object translations. The method focused on position adjustment, but could not handle more complex tasks like rotating or scaling. Chen et al. compared a sphere-mapping rotation method with bar-based rotation methods using a mouse on a 2D monitor [3]. While their study gives thorough insights on the design of rotation interfaces with 2D input modalities, it is difficult to directly apply their results to eye-based interaction in a 3D scenario due to the difference in the control methods.

In this work, we take a step towards completely hands-free manipulation and explore three different techniques for eye gaze-based object rotations. The first two techniques, RotBar and RotPlane, allow users to select rotations around one world axis at a time, and they include two unique visualizations. We have also designed a third technique, RotBall, that draws inspiration from the traditional arcball manipulation scheme [20], though our design is significantly adapted to work with eye gaze. All three methods for rotation, shown in Figure 1, went through several test and redesign iterations as the nature of gaze interactions are much more confined than hand-eye coupled interactions like controllers or gestures.

We conducted a user experiment to compare the three methods in terms of speed, accuracy, and usability. The experiment included orientation tasks requiring both single-axis and multi-axis alignments. Quantitative results showed that users could perform single-axis orientation with RotBar and RotPlane significantly faster and more accurate than RotBall. On the other hand for multi-axis orientation tasks, RotBall significantly outperformed RotBar and RotPlane in speed and accuracy. Results also revealed that all three methods allowed users to effectively align the object with a mismatched angle less than 6 degrees for both single-axis and multi-axis rotations, and there was no significant difference in terms of the alignment accuracy between the three methods. Results of subjective evaluation suggested that there was no specific tendency in the preference among the three methods.

In summary, the contributions of this work include:

- We implemented three different methods that improve upon existing work and redesigned them to specifically address the needs of those who need eye-only control to rotate virtual objects.
- We conducted a user study to compare these methods for docking tasks to determine the suitability of these techniques both for simple and complex object manipulations. Our results show that RotBar and RotPlane are more suitable for simple rotations and RotBall is more suitable for complicated manipulations.
- From the results of our study, we provide several guidelines to inform the design of future interfaces for gaze-based manipulation.

2 RELATED WORK

The research most closely related to our work includes user interfaces that have been designed for manipulating the rotation of virtual objects.

2.1 Object Orientation

The function of rotating a virtual object is a fundamental element in virtual environment design. Over several decades, various devices and methods have been explored that target interactive 3-DoF orientation of virtual objects. For example, in 2D interfaces, different viewpoints of the object and control bars where users can control 2 DoF at a time have been widely used [8]. In another example, Chen et al. [3] compared bar based rotation interfaces with a *Virtual Sphere* visualization that encapsulates the target object into a sphere and maps mouse strokes to "rolling" of the sphere. They also considered a second sphere rotation technique called *XY+Z* that allowed users to continuously control the rotation around the X and Y axes as the user swipes across the sphere, and the rotation around the Z axis by pressing and moving the mouse around the sphere. They found that the slider controls performed faster for simple rotations around a single axis but were slower for more complex manipulations and that most participants preferred using the Virtual Sphere metaphor over sliders. The Arcball is a method similar to the Virtual Sphere, except that the implementation of rotation is based on quaternion curves [19, 20]. Katzakis et al. [6] evaluated the effects of drawing a sphere around the manipulated object, a method called Arcball-3D, and selection of the rotation point directly through raycasting, called Meshgrab, for ray-based object manipulation. They found that users could rotate objects faster with Arcball-3D and generally preferred it over Meshgrab.

Groß [4] designed a manipulation tool where rotations happen along prescribed axes through continuous rotation or the selection of two points on rings surrounding the object to define the rotation angle. She also defined a free-rotation mode that gradually aligns a selected point with the forward facing direction. Pathmanathan et al. [13] compared head and eye-gaze for manipulating object transformations using these manipulation techniques. They found that users preferred head-gaze, while there was no difference in the preference between the constrained and free-rotation methods.

Although their work is similar to ours, their rotation method is still selection-based, and users only have control limited to the rotation velocity in one direction.

While our exploration bears some similarity to the work of Chen et al., their work presented the information to users at a single location, and the rotation controls were aligned with the user's view. While this may be an option on a monitor on HMDs users can move their voluntarily or involuntarily thus shifting the controls. We thus align use the axes of the world coordinate system to allow users to consistently control the rotation of the object.

Utilizing gaze also presents several additional challenges over a desktop mouse. For example, gaze results in more noisy movements due to jitter and tracking errors. Fixations over a given time period have been widely used to avoid the Midas Touch problem of unintended activations [5]. This could significantly increase the time required to rotate the object when using *Virtual Sphere*. Finally, visual confirmation of the intended rotation with the object's current rotation may cause unintentional rotations. We thus introduce an improvement over bar controls used by Chen et al. that allows users more freedom to position their gaze while performing the rotation.

2.2 Eye Tracking and Control

One primary advantage of eye-based interaction is that using eye movements requires less muscle compared to performing body movements, which contributes to quick pointing and reduction of physical fatigue For example, Raiha et al. showed that gaze can be used to more quickly select between cursors distributed over multiple screens to increase performance [17]. Additionally, the angular speed of travel of conscious eye movements is much faster compared to the hand or body. This is beneficial in terms of performance, but can also be a challenge when trying to achieve eye-based interaction that allows for high degrees of freedom, e.g. moving and rotating virtual objects in a three-dimensional space.

Though less related to object manipulation, Puimsonboon et al. designed several methods to disambiguate between 3D object selection [15]. These methods, Duo-reticles, Nod-and-Roll, and Radial Pursuit, provide additional methods for selection outside of the traditional gaze-dwell mechanism. Moreover, these methods have a closer relationship with objects in the environment that might be selected for the purposes of augmented or mixed reality. A "Pursuit," the concept originally developed by Vidal et al. [27] also makes use of a similar mechanism for making hands-free selections by tracking the eye's smooth pursuit of an object and detecting its corresponding trajectory. This allows for a more expressive mode of interaction and location independence since pursuits can have different shapes. Pursuits are also suitable for hands-free object selections in virtual environments [7], or even selecting occluded objects [21]. Recent literature also gives insights on utilizing eye gaze compensated with head movements for gaze-based selections for improving targeting precision [10], enhancing usability [11, 23], and freeing eye gaze exploration [22].

More recently, we investigated the usability of gaze-only positioning for translations, and found that plane-based interfaces could assist with three-dimensional positioning via eye gaze [9]. One of the methods for performing rotations in this work draws from ideas presented in that plane-based interaction.

2.3 Further Motivation

It is not hard to imagine that there exist practical use cases of HMDs where controller-based devices or extensive body movements are not available, e.g. in a crowded public space or with users' hands occupied by other tasks. Thus options are necessary that allow for completely hands-free interaction with HMDs.

As a step towards this goal, our work is dedicated to handling hands-free object orientation. While large number of methods exist that effectively supports orientation of virtual objects, there is few that works in a hands-free manner. To tackle this issue, this work implemented several prototypes of gaze-based user interfaces intended for manipulating the rotation of virtual objects on HMDs, and investigates the usability of such interfaces through thorough user studies.

3 INTERACTION METHODS

In this work, we present and test three methods suitable for gaze-based rotation, denoted as RotBar, RotPlane, and RotBall. Each takes advantage of a different mechanism and visualization for performing rotations, and we also describe the adaptations that were necessary to ensure these were usable with eye gaze.

3.1 Initial Axis Selection

The first two rotation methods are preceded by an axis selection phase, as shown on the leftmost image in Fig. 2. The visualization for this selection includes a set of typical coordinate system axes included in programs like Unity[1] or Blender[2], but the ends of these axes are capped with a circular arrow icon. This 3D visualization helps the user understand the axis of rotation more effectively since he or she has a perspective view of these arrows. Though we initially only included the axes lines, several initial testers mentioned that this was unclear since the lines were essentially 2D.

These axes can be selected by any eye based method such as dwell or blink based selections. Upon selecting one of the three axes, the user can then utilize RotBar or RotPlane to complete the rotation. Note that these axes disappear once the user has engaged any of the three rotation methods.

3.2 RotBar

The first method we tested was RotBar, which is a method that enables per-axis rotation. By mapping each per-axis rotation to a 1-DoF eye gaze movements, RotBar is designed to simplify the eye movements users must do to manipulate the object. It makes use of three "bars" that travel along and correspond to the selected axis of rotation. These essentially function as a visual gauge to determine the degree of rotation, as shown in Fig. 2. Rotations are mapped so that the one side of the bar corresponds to a 180-degree rotation in one direction and the other side corresponds to 180 degrees in the opposite direction. The centers of the three bars shown in the top left row of Fig. 2 represent the initial orientation of the object during each rotation action. As the user's gaze proceeds along each of these bars, the gauge turns from green to yellow, and the object rotates synchronously with the gaze change.

[1] https://unity.com/
[2] https://www.blender.org/

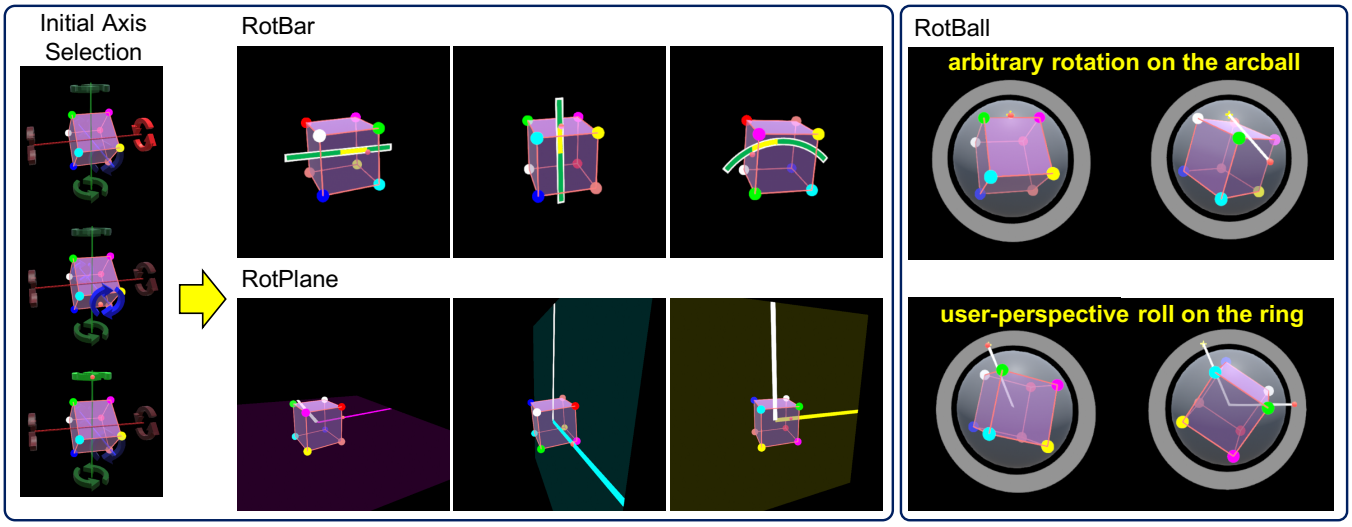

Figure 2: Figures showing the three interaction methods we tested. On the left are two per-axis rotation methods: RotBar and RotPlane. A commonality between the per-axis methods is that users first select one world axis (very left of the figure) to enter a rotation mode. RotBar, shown in the upper left row, maps a 360-degree rotation to a bar for each rotation axis where, the center of the bar represents the starting orientation. RotPlane, shown in the bottom left row, calculates a rotation based on the angular offset of where the eye gaze intersects a plane relative to the white bar on the same plane. On the right is RotBall, which is a method composed of a traditional arcball with a surrounding ring that enables user-perspective rolling rotations. In RotBall, users can set an initial pin and then conduct a rotation based on the new gaze point and that pin. The resulting rotation is calculated as the angular offset from the first pin based on the central point of the ball.

Thus, a rotation r [rad] on each axis is calculated based on a 1-dimensional gaze offset d from the central white bar as:

$$r = 2\pi d, \tag{1}$$

where d is given corresponding to the rotation axis as:

$$d = \begin{cases} \dfrac{gaze_x - bar_x}{\mathrm{L_h}}, & \text{yaw and roll} \\ \dfrac{gaze_y - bar_y}{\mathrm{L_v}}, & \text{pitch} \end{cases} \tag{2}$$

where $gaze_{x,y,z}$ denotes the gaze position, $bar_{x,y,z}$ denotes the position of the bar center, and $\mathrm{L_h}/\mathrm{L_v}$ denote the horizontal/vertical length of the gauge, respectively. In short, the gauge linearly maps a full 360-degree rotation for any axis to the offset from the corresponding central bar to the gaze point.

When one of the axes appears almost parallel to the user's viewpoint, e.g., <10° offset, it would be very difficult for the user to control the rotation accurately. To prevent this, we switch the visualization from a bar to an arc around the axis. In Fig. 2 this applies to the cube's roll.

3.3 RotPlane

In our previous work [9], we found that planes are effective for guiding arbitrary eye gaze movements and enabling eye gaze-based per-axis positioning in virtual environments. Following the insights, RotPlane makes use of three orthogonal planes to handle the per-axis rotation, as shown in Fig. 2. Here the user's gaze functions like a handle that is rotated from its original location (white bar) to the target location (the user's gaze) on a plane around the corresponding axis. The bars on the plane are visualized to match the eye gaze movements with the acquired mental rotation. Assuming rotation r [rad] is clockwise at each axis, it can be acquired using the same equation as Eq. (1), but in this case the offset d is given by a planar angular offset between from the white bar to the gaze point as:

$$d = \frac{\angle(\hat{bar}, \hat{gaze})}{2\pi} \tag{3}$$

where $\angle(\hat{a}, \hat{b})$ denotes a directional angle from vector $\hat{a}$ to $\hat{b}$ resulting in $[0, 2\pi]$, $\hat{gaze}$ denotes the vector from the center of the plane to the gaze point, and $\hat{bar}$ denotes the vector from the center to the tip of the bar. While this method is similar to the ring control approach of Groß [4], the larger area where users can position their gaze and the alignment of the rotation indicator with the user's gaze could help users keep their focus on the object they manipulate instead of the control elements.

3.4 RotBall

Previous work showed that sphere based control methods can be efficient in controlling an object's orientation. To investigate the applicability of these findings to gaze based manipulation we designed RotBall that combines a traditional arcball mechanism with a surrounding ring to support rolling rotations that would otherwise be difficult due to the user's perspective. We visualize a sphere around the object and users can rotate the object by interacting with the sphere. As such, it is similar to Chen et al.'s *Virtual Sphere* [3] and Katzakis et al.'s *Arcball-3D* [6]. Since the arcball mechanism does not require a single-axis selection, there is no axis selection phase.

Instead, the user performs a 2-step command to apply a rotation: setting an anchor point and then acquiring a rotation based on the spherical gaze trajectory from the anchor point. This is similar to the placement and release of a finger on a ball mouse. The initial placement of the user's finger sets the initial point to drag, then dragging his or her finger will rotate the ball in-place, and finally the removal of the finger will disengage the rotation of the ball.

In general, a rotation r [rad] on the RotBall is calculated as

$$r = \angle(\hat{anchor}, \hat{gaze}) \tag{4}$$

where $\hat{gaze}$ denotes the vector from the center of the RotBall to the second gaze point, and $\hat{anchor}$ denotes the vector from the center to the first anchor.

4 EXPERIMENT

This section introduces the details of a user experiment that we conducted to validate the usability and user experience of each rotation paradigm. In the experiment, participants were asked to use RotBar, RotPlane and RotBall to complete a set of orientation tasks (docking tasks) in an HMD. For all three methods, participants used eye gaze to rotate cubes as closely as possible to a target rotation, represented by a semi-transparent target cube as shown in Figure 3. Both the target cube and cube to rotate contained 8 colored points, one for each corner, to help the user understand the orientation and confirm that the task was completed correctly. We selected simple alignment targets in line with previous research on object manipulation [6]. While the simple alignment task does not perfectly reflect realistic scenarios, e.g. a user placing a virtual furniture next to a wall, it allows an unbiased comparison of the manipulation techniques.

Though gaze dwell, blink, or other eye-based selection methods are compatible with these rotation paradigms, we asked participants to make and confirm selections using the trigger and touch pad buttons on an HTC Vive controller. Our previous work [9] showed that for a manipulation method that utilizes only eye gaze, familiarity with gaze-based selection and gaze dwell could have a significant effect on the performance and subjective evaluation of manipulation methods. To ensure that this would not bias the results of our own evaluation, we opted to confirm selections with a controller, which presents a familiar interface to all users while avoiding the Midas touch problem [5].

4.1 Hypotheses

The main purpose of the experiment is to evaluate how quickly and accurately users would perform with each method for eye gaze-based orientation tasks. Based on the previous findings of Chen et al. [3], we expected that for single-axis orientation tasks, axis-based methods could outperform arcball-based methods in terms of speed, and the opposite for multi-axis tasks. While RotBar and RotPlane map a full 360-degree rotation in a specified field of view, RotBall has the least sensitivity and was expected to be most suitable for small alignments. In addition, compared to Rotbar, RotPlane occupies a larger field of view and could potentially be easier to perform slight alignments. Thus we expected that RotPlane will overall outperform RotBar. To summarize, we formulated the following hypotheses:

- **H1a** Participants will complete single-axis orientation tasks most quickly with RotPlane, followed by RotBar, and then RotBall.
- **H1b** Participants will complete the multi-axis orientation tasks most quickly with RotBall, followed by RotPlane, and then RotBar.
- **H2** Participants will complete the orientation tasks most accurately with RotBall, second with RotPlane, and third with RotBar.

4.2 Setup and Participants

In total, we recruited 11 students and researchers from multiple universities, six male and five female, ranging in age from 22 to 33 (avg. 27.2, stdev. 3.7) through participant calls on mailing lists and social networks. Of these participants, six were wearing prescription glasses with the HMD during the experiment. Five of them had no experience in eye tracking and eye gaze-based human-computer interaction, two had less than five times in total, while the remaining four had more experience (more than five times in total) before this experiment. Four of them ran this experiment remotely in their home or office to mitigate the transmission of infectious diseases during in-person experiments. The in-person experiment ran on a desktop computer with an Intel Xeon E5–2690 CPU and an Nvidia GeForce GTX 970 GPU at an average frame rate of 40 frames per second, synchronized for both eye tracking and graphical rendering. For the HMD, all participants used an HTC Vive Pro Eye that has integrated eye tracking cameras and provides relatively stable eye gaze data. For participants running the experiment remotely, we ensured via a remote connection that all participants could run the eye tracking, rotation interfaces, and experiments without any issues. The frame rate of the application was set to 40 frames per second, the same as the in-person test bed. When the experiment was conducted remotely, an experiment conductor supervised the remote participant throughout the experiment to ensure that the external conditions matched that of in-person participants as closely as possible. Participants who took part in the experiment in-person received a gift card worth approximately 5 USD as remuneration.

The virtual environment was created using Unity 2018.3.2f1 and contained all of the rotation paradigms and experiment tasks. The eye tracking data was run and recorded in real-time through the Vive Sranipal runtime version 1.1.2.0. We applied a smoothing filter that aggregated and averaged the eye gaze data for the last 5 frames. The eye gaze point was visualized as a red dot only when the eye gaze intersected with interactive objects and interfaces. The statistical evaluation of the experimental data was processed using R 3.6.2 [3] and coin 1.3-1 [4].

4.3 Procedure

When participants entered the experiment environment, they first received an introduction to the experiment tasks and an explanation of each rotation method. Second, they read and signed a consent form that explained the details and risks of the experiment. During the experiment participants were asked to remain seated on a swivel chair and not to stand up or move around. Local body movements were not physically restricted, so participants could rotate their

[3] https://www.r-project.org/
[4] http://coin.r-forge.r-project.org/

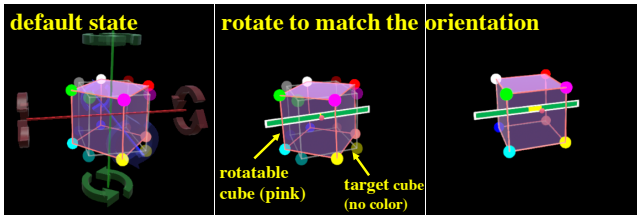

Figure 3: Sample images showing the orientation task in our user study. (a) The default state where the two cubes are of different orientations. The user selects an axis in this phase. (b) After choosing an axis, the rotation mode (in this case RotBar) is engaged. (c) Image showing that the two cubes have been aligned, after which the user can confirm the rotation and end the trial.

chair if necessary. We also did not restrict the participant's neck, head, or seated body movements to ensure that the participants could view the tasks in a natural manner and perform natural eye movements supported by head movements.

We ran the Vive Pro Eye integrated eye tracking calibration at the very beginning of the experiment. We also showed the tracked gaze point to participants during the experiment to ensure that the tracking accuracy was adequate. Participants were able to pause the trial and rerun the eye tracking calibration if they felt the eye tracking had become inaccurate (e.g. due to drift) during the experiment.

Before entering the formal trials, participants had a training session where they could practice each method until they felt comfortable, and could take a break between each trial if needed. Participants had to finish all trials with one method, and then proceeded to the next. After finishing all trials, participants completed a System Usability Scale (SUS) survey [2] for each method. Overall, the experiment took from 60 to 120 minutes, including the consent process and surveys. This experiment was approved by the Osaka University Institutional Review Board.

4.4 Tasks and Conditions

As shown in Fig. 3, participants first see two cubes positioned in front of them, a pink cube that is rotatable, and a white-framed cube that has a different (target) orientation. The task is to rotate the pink cube to match the target orientation using the 8 colored dots affixed to each corner. The starting view point was positioned at (0, 1, 0) Unity units (meters) and faced the front direction: (0, 1, 1) meters. The orientation tasks for each method included two sets of 3 single-axis and 1 multi-axis matching tasks, $(1/4\pi, 0, 0)$, $(0, 1/4\pi, 0)$, $(0, 0, 1/4\pi)$, and $(1/4\pi, 1/4\pi, 1/4\pi)$ [rad] in the world coordinate system, ordered from single-axis to multi-axis. The two sets appeared at two different locations: (0.75, 0.3, 2) meters and (-0.75, 0.3, 2) meters. This resulted in 8 trials per method, with 6 single-axis and 2 multi-axis, for a total of 24 trials for the experiment. The order of the methods was counterbalanced using a Latin square to alleviate ordering effects.

As mentioned previously, participants used 2 buttons of an HTC Vive controller: the trigger button for making selections (e.g. selecting a rotation axis and fixating a rotation), and the touchpad button for making confirmations (e.g. confirming that they had finished a trial and proceeding to the next trial). Additionally, the controller had no aiming or laser function, as participants had to exclusively use their eye gaze to utilize each method to rotate the object.

As the priority goal of each task, we asked participants to match the orientation as accurately as possible. Although we were recording the time taken for each trial, we did not set a time limit. The timer of each trial would not start until the participant made the first selection (trigger button), giving participants sufficient time to observe the target orientation at the very beginning of each trial without needing to pay attention to a timer.

After participants rotated the object and decided that it was correctly aligned with the target, or that they could not align it more accurately, they pressed the touchpad button to conclude the trial and proceed to the next one. Note that participants could not perform the final confirmation while in the rotation mode, which means participants had to affix the in-progress rotation prior to concluding the trial. Between each trial, participants could take breaks without any time restrictions and proceed to the next trial by pressing the touchpad button when ready. We gave participants full control of the trial start time so that they had ample time to understand the target orientation. Otherwise this mental effort might have biased the rotation times.

For evaluating the experimental results quantitatively, we recorded and calculated the following metrics:

- **Completion time**: Completion time is the time between the first selection and the confirmation of the alignment by the participant. For a more objective comparison, we show the completion time of single-axis and multi-axis tasks separately since the multi-axis tasks were more complicated. Note that we did not set a time limit for each trial, which means the completion time could become either very short or extremely long depending on how the participant defined a ''good match". Although we considered limiting the time to one minute to allow for more trials, we opted for unlimited time as this would allow us to analyze both time and accuracy without any trials that were cut-off mid-rotation.
- **Misalignment**: Misalignment is defined as the mismatched angle in [rad] by calculating the angular offset of the rotated object from the target orientation for each trial. This essentially gives us an idea of the accuracy with which participants were able to match rotations. We also show these results separately for single-axis and multi-axis tasks.
- **Number of selections**: Number of selections is the total number of trigger presses from the start of the first rotation to the end of the trial. This gives us an idea of how many interactions (dwells in the case of complete eye-control) were necessary for each method.

4.5 Results

Here we describe both the quantitative (speed, misalignment, and number of selection) and qualitative (subjective scoring from the SUS questionnaire) results of the experiment. We verified if the data met with the requirement of ANOVA by performing Anderson-Darling normality tests and Mauchly's W sphericity tests. We used a threshold of $p = 0.05$ to determine statistical significance. We

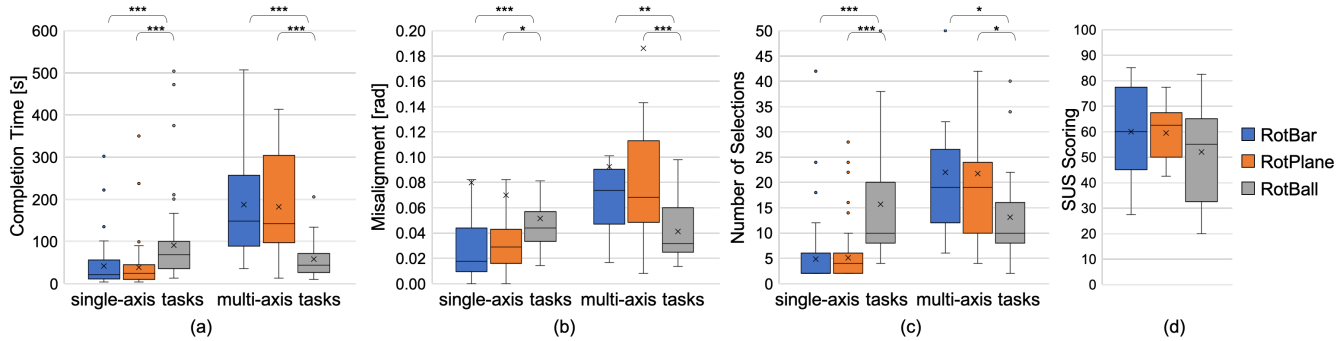

Figure 4: Boxplots showing (a) completion time for both single-axis and multi-axis tasks, (b) misalignments for both single-axis and multi-axis tasks with outliers excluded for clarity, (c) number of selections (trigger presses) required to complete a trial, and (d) average system usability (SU) score of each method from all participants, as calculated by Brooke's standard scoring method [2]. (*: $p < 0.001$, **: $p < 0.01$, *: $p < 0.05$)**

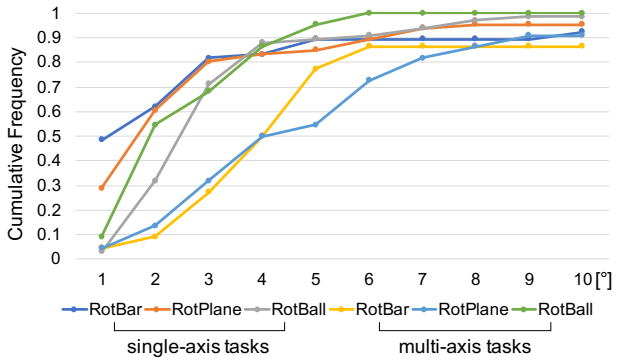

Figure 5: A graph showing the cumulative frequency curve of the matching results of each method for box single-axis and multi-axis tasks, within a range of 10-degree misalignments devided by every 1 degree.

report $r = Z/\sqrt{N}$ as the effect size for post-hoc Wilcoxon signed rank test results, where Z is the statistical value and N is the total sample size [18].

Fig. 4 (a) shows the completion time of both single-axis and multi-axis tasks. One-way ANOVA with repeated measures yielded significant variation among all three methods for both conditions (($F(2, 185) = 13.90, p < 0.001$) for single-axis tasks and ($F(2, 53) = 17.40, p < 0.001$) for multi-axis tasks).

For single-axis tasks, a post-hoc Tukey test showed significant difference in completion time between RotBar and RotBall ($p < 0.001$), and between RotPlane and RotBall ($p < 0.001$). Average times taken for single axis tasks were 41.78, 39.06, and 90.85 seconds for RotBar, RotPlane, and RotBall, respectively. RotBall took more than double the time of the other methods.

For multi-axis tasks, a post-hoc Tukey test showed significant difference in completion time between RotBar and RotBall ($p < 0.001$), and between RotPlane and RotBall ($p < 0.001$). Average times taken for multiple axis tasks were 187.13, 181.78, and 54.46 seconds for RotBar, RotPlane, and RotBall, respectively. In contrast to single access tasks, RotBall took less than a third of the time of the other two methods.

Fig. 4 (b) shows the results of the misalignment of each method for both single-axis and multi-axis tasks. For single-axis tasks, Friedman's tests showed statistical significance between the different methods ($\chi^2(2) = 25.41, p < 0.001$). A post-hoc Wilcoxon signed rank test with Bonferroni correction showed significant differences between RotBar and RotBall ($Z = 3.62, r = 0.32, p < 0.001$), and between RotPlane and RotBall ($Z = 2.75, r = 0.24, p < 0.01$). For multi-axis tasks, Friedman's tests showed statistical significance between the different methods ($\chi^2(2) = 13.73, p < 0.01$). A post-hoc Wilcoxon signed rank test with Bonferroni correction showed significant differences between RotBar and RotBall ($Z = 3.04, r = 0.46, p < 0.01$), and between RotPlane and RotBall ($Z = 3.33, r = 0.50, p < 0.001$).

In terms of accuracy, we also show a cumulative frequency curve in Fig. 5 covering the percentages of how each method performed the matching for each task. For single-axis tasks, all three methods received an over 70% matching with the misalignment less than 3 degrees. For multi-axis tasks, RotBall received a higher rate of nearly 70% for matching with the misalignment of less than 3 degrees, while RotBar and RotPlane reached over 70% matching of misalignments less than 6 degrees.

Fig. 4 (c) shows the number of selections performed for accomplishing each task. One-way ANOVA with repeated measures yielded significant variation among all three methods for both conditions (($F(2, 185) = 35.94, p < 0.001$) for single-axis tasks and ($F(2, 53) = 4.70, p < 0.05$) for multi-axis tasks). For single-axis tasks, a post-hoc Tukey test showed significant difference between RotBar and RotBall ($p < 0.001$), and between RotPlane and RotBall ($p < 0.001$). For multi-axis tasks, a post-hoc Tukey test showed significant difference between RotBar and RotBall ($p < 0.05$), and between RotPlane and RotBall ($p < 0.05$).

In general, quantitative results revealed that for single-axis tasks, RotBar and RotPlane outperformed RotBall in both speed and accuracy, while for multi-axis tasks, RotBall oppositely was faster and more accurate than RotBar and RotPlane.

Friedman's tests showed no significant difference between methods for each survey item. We also calculated and show the system usability (SU) scoring in Fig. 4 (d) using Brooke's standard scoring

method [2]. All three methods received a medium scoring in average, 60.00 for RotBar, 59.55 for RotPlane and 52.05 for RotBall. A Friedman's test showed no significant difference in the scoring between each method. We also asked participants about their most preferred method. Among 11 participants, 4 voted for RotBar, 4 voted for RotPlane, and the remaining 3 voted for RotBall. In general, we found neither significant difference in the usability of the three methods, nor significant trend in the individual preference.

5 DISCUSSION

Experimental results revealed that RotBar and RotPlane were completely opposite of RotBall when comparing single- and multi- axis tasks in terms of speed, as outlined in Figure 4 (a). These results are in line with previous findings by Chen et al. [3] and partially support **H1a** and **H1b**, and suggests that it may be better to use a combination of the two types of method during 3D modeling or other object rotation tasks. RotBall is more appropriate for complex rotation tasks are necessary, but for smaller or single-axis rotations, RotBar or RotPlane would be faster. This tendency was present for the total number of controller presses, as shown in Fig. 4 (c).

Experimental results regarding accuracy suggested that for single-axis tasks, the per-axis methods were significantly more accurate than the arcball-based method. While for multi-axis tasks, the arcball-based method could significantly outperform the per-axis methods in accuracy. This result partially supports **H2**. We suspect that because we gave the participants unlimited time to complete a rotation, final accuracy ended up being more of a function of personal preference rather than a characteristic of a specific rotation method. Results might have been different if we capped the trial time, but we felt that this would not be representative of real world manipulation tasks such as 3D modeling or design in which users can take their time.

Although we expected that RotPlane would help users to align the two cubes as users could adjust the displacement of their gaze from the center, thus keeping the two cubes in focus at all times, our results and comments from the participants showed that this was not the case. We believe that the main reason for that was that participants relied on the corners to verify the quality of the alignment. As we automatically selected the initial orientation of the cube that would then be aligned with the user's gaze participants could not always rely on the point they were fixated on to determine the alignment quality.

Regarding interface designs, we purposely did not add a grid for visual guidance for all three methods to fundamentally test how raw eye gaze would perform. For eye gaze-based interactions, we expect that adding visual guidance, such as grid textures, could make it easier for users to gaze at specific positions and thus improve gaze accuracy and stability. However, inadequately integrated visual information could also lower performance, e.g. excessively detailed grids showing marks one degree apart might instead confuse the user, as he or she might have to continuously filter out information. Besides, the appearance of the visual guidance, e.g. colors, shapes and transparency, could also have impact on the usability. As such, it is worthwhile to further explore how different visual guidance could affect performance and to explore optimal visual guidance for eye gaze-based rotation.

5.1 Design Implications and Future Work

Although participants could align the cube with its target using all 3 methods and did rate them similarly, all methods were rated lower than the median score of previous studies that utilized the SUS [1]. For RotBar and RotPlane, we mapped a 360-degree rotation with equivalent rotation speed to each axis for achieving a maximum rotation range. Subjective comments gave that in the cases of small adjustments, the rotation speed was too fast compared with the eye gaze movements, e.g., rotating 1 degree was more difficult than rotating 90 degrees as the user had to perform very minute eye movement. Considering that practical rotation tasks often require small adjustments, we suggest an improved gaze-based rotation interface should include rotation speed adaptive to the task needs to enable huge fast adjustments and small precise adjustments simultaneously. Nonisomorphic mapping of rotation amounts has been shown to lead to faster rotations without a loss of accuracy [16] and could potentially be applied here as well.

We also observed an issue that would affect the results of the orientation tasks and increase users' frustration. As part of the task design in the experiment, participants also needed to check whether the two cubes were well-aligned during rotation. In some cases, e.g., when the corners shifted from a participant's central/paracentral vision into the peripheral vision during the manipulation, trying to match the corners would cause a sudden eye movement and accidentally trigger an unintended rotation. From our observation, the issue was likely due to the nature of the matching task, which required the user to view the cube in parallel to the manipulation. This could also explain the difference in our findings from those of Pathmanathan et al. [13], who collected feedback after a free manipulation task without a defined target object pose. One available way to tackle this issue would be improving the design of the interfaces to focus important information in the central and paracentral vision, such as showing a small copy of the rotated object at the eye gaze position. However, this could lead to clutter in the user's view. Alternatively, a world-in-miniature representation placed in an area that does not overlap with other content could also help user's to more easily verify the rotation without shifting their gaze. We consider this issue to be especially critical for eye gaze-based manipulation interfaces, especially for use cases requiring simultaneous matching and manipulation, which deserves further investigation.

In addition, this work mainly focuses on evaluating how eye gaze will perform on object rotation. Thus we implemented all tested methods with a controller to support selections. It is possible that completely hands-free control could have lowered the performance of the rotation methods due to the requirement of performing gaze-based selections simultaneously with the rotation. As such, it is also crucial to further explore optimal gaze-based selection methods for rotation interfaces. An inappropriate selection method would lead to a poor user experience due to problems such as Midas Touch problem. However, some existing solutions that require additional eye movements, e.g. "Pursuit" [27] or "DualGaze" [12], are not completely compatible with the rotation interfaces. Unlike selection tasks, rotation tasks ask for sustained and stable focus of attention on the interface, which does not allow extra eye movements during the manipulation. In our previous work [9], we found that while

gaze dwelling could work with gaze-based manipulation interfaces, this was difficult for some users to perform and might have caused frustration during the experiment. Sidenmark et al. [22] propose gaze-based selection methods that leverage eye and head movements, which can help free up eye gaze exploration during tasks. However, it remains unclear if the required extra head movements would affect the performance of manipulations. Thus, there is still a lack of efficient selection methods optimized for manipulation interfaces, which needs further development and investigation.

Future work also includes the extension of eye gaze-based manipulation methods to scaling operations and full combined 9-DoF translation, rotation, and scaling tasks, with the challenge of designing a universally functional interface.

6 CONCLUSION

In this work, we designed and explored the usability of three different methods, RotBar, RotPlane and RotBall, that are suitable for eye gaze-based rotation of 3D objects. RotBar makes use of three bars to handle the per-axis rotation and acquires a rotation determined by how much the eye gaze positions from the center of the bar. RotPlane enables per-axis rotation using three orthogonal planes and calculates a rotation based on an angular trajectory on the plane. RotBall combines a traditional arcball that allows arbitrary rotations with a surrounding ring that is supportive for user-perspective roll rotation. Experimental results showed that RotBar and RotPlane were faster and more accurate in performing single-axis rotations, but that RotBall greatly outperformed the other two methods for multi-axis rotations. As none of the methods were clearly preferred by participants the methods could be selected specifically for the task at hand, such as RotBar or RotPlane for rotating an avatar or a virtual box placed on the floor and RotBall for rotating a virtual model that the user wants to look at from all sides.

Our interaction design can serve as a significant step towards 3D modeling, editing, and interaction using pure eye gaze, and these methods will be advantageous for individuals that have limited use of their hands or arms during interaction. We also hope that this work will help pave the way for new eye gaze-based manipulations moving forward.

ACKNOWLEDGMENTS

This work was funded in part by the United States Office of Naval Research Global, Grant #N62909–18–1–2036.

REFERENCES

[1] Aaron Bangor, Philip Kortum, and James Miller. 2009. Determining what individual SUS scores mean: Adding an adjective rating scale. *Journal of usability studies* 4, 3 (2009), 114–123.

[2] John Brooke. 1996. SUS-A quick and dirty usability scale. In *Usability evaluation in industry*. London: Taylor and Francis, 189–194.

[3] Michael Chen, S. Joy Mountford, and Abigail Sellen. 1988. A Study in Interactive 3-D Rotation Using 2-D Control Devices. *SIGGRAPH Computer Graphics* 22, 4 (1988), 121–129.

[4] Anja Groß. 2018. *Benutzerinteraktion in Virtual Reality mittels Eye Tracking*. B.S. thesis.

[5] Robert J. K. Jacob. 1995. Eye Tracking in Advanced Interface Design. In *Virtual Environments and Advanced Interface Design*. Oxford University Press, Inc., 258–288.

[6] Nicholas Katzakis, Kazuteru Seki, Kiyoshi Kiyokawa, and Haruo Takemura. 2013. Mesh-grab and arcball-3d: Ray-based 6-dof object manipulation. In *Proceedings of the 11th Asia Pacific Conference on Computer Human Interaction*. 129–136.

[7] Mohamed Khamis, Carl Oechsner, Florian Alt, and Andreas Bulling. 2018. VRpursuits: Interaction in Virtual Reality Using Smooth Pursuit Eye Movements. In *Proceedings of the 2018 International Conference on Advanced Visual Interfaces (AVI '18)*. Article 18, 8 pages.

[8] Joseph J LaViola Jr, Ernst Kruijff, Ryan P McMahan, Doug Bowman, and Ivan P Poupyrev. 2017. *3D user interfaces: theory and practice, 2nd ed.* Addison-Wesley Professional.

[9] Chang Liu, Alexander Plopski, and Jason Orlosky. 2020. OrthoGaze: Gaze-based three-dimensional object manipulation using orthogonal planes. *Computers & Graphics* 89 (2020), 1–10.

[10] Diako Mardanbegi, Tobias Langlotz, and Hans Gellersen. 2019. Resolving Target Ambiguity in 3D Gaze Interaction through VOR Depth Estimation. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems (CHI '19)*. 1–12.

[11] Diako Mardanbegi, Benedikt Mayer, Ken Pfeuffer, Shahram Jalaliniya, Hans Gellersen, and Alexander Perzl. 2019. EyeSeeThrough: Unifying Tool Selection and Application in Virtual Environments. In *2019 IEEE Conference on Virtual Reality and 3D User Interfaces (VR)*. 474–483.

[12] Pallavi Mohan, Wooi Boon Goh, Chi-Wingand Fu, and Sai-Kit Yeung. 2018. DualGaze: Addressing the Midas Touch Problem in Gaze Mediated VR Interaction. In *2018 IEEE International Symposium on Mixed and Augmented Reality Adjunct (ISMAR-Adjunct)*. 79–84.

[13] Nelusa Pathmanathan, Michael Becher, Nils Rodrigues, Guido Reina, Thomas Ertl, Daniel Weiskopf, and Michael Sedlmair. 2020. Eye vs. Head: Comparing Gaze Methods for Interaction in Augmented Reality. In *ACM Symposium on Eye Tracking Research and Applications* (Stuttgart, Germany) *(ETRA '20 Short Papers)*. Association for Computing Machinery, New York, NY, USA, Article 50, 5 pages.

[14] Ken Pfeuffer, Benedikt Mayer, Diako Mardanbegi, and Hans Gellersen. 2017. Gaze + Pinch Interaction in Virtual Reality. In *Proceedings of the Symposium on Spatial User Interaction*. 99–108.

[15] Thammathip Piumsomboon, Gun Lee, Robert W Lindeman, and Mark Billinghurst. 2017. Exploring natural eye-gaze-based interaction for immersive virtual reality. In *2017 IEEE Symposium on 3D User Interfaces (3DUI)*. IEEE, 36–39.

[16] Ivan Poupyrev, Suzanne Weghorst, and Sidney Fels. 2000. Non-Isomorphic 3D Rotational Techniques. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (The Hague, The Netherlands) *(CHI '00)*. Association for Computing Machinery, New York, NY, USA, 540–547. https://doi.org/10.1145/332040.332497

[17] Kari-Jouko Räihä and Oleg Špakov. 2009. Disambiguating ninja cursors with eye gaze. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. 1411–1414.

[18] Robert Rosenthal. 1994. Parametric Measures of Effect Size. In *H. Cooper & L. V. Hedges (Eds.), The handbook of research synthesis*. 231–244.

[19] Ken Shoemake. 1985. Animating Rotation with Quaternion Curves. In *Proceedings of the 12th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '85)*. 245–254.

[20] Ken Shoemake. 1992. ARCBALL: A User Interface for Specifying Three-Dimensional Orientation Using a Mouse. In *Proceedings of the Conference on Graphics Interface '92*. 151–156.

[21] Ludwig Sidenmark, Christopher Clarke, Xuesong Zhang, Jenny Phu, and Hans Gellersen. 2020. Outline Pursuits: Gaze-Assisted Selection of Occluded Objects in Virtual Reality. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems (CHI '20)*. 1–13.

[22] Ludwig Sidenmark and Hans Gellersen. 2019. Eye&Head: Synergetic Eye and Head Movement for Gaze Pointing and Selection. In *Proceedings of the 32nd Annual ACM Symposium on User Interface Software and Technology* (New Orleans, LA, USA) *(UIST '19)*. Association for Computing Machinery, New York, NY, USA, 1161–1174.

[23] Ludwig Sidenmark, Diako Mardanbegi, Argenis Ramirez Gomez, Christopher Clarke, and Hans Gellersen. 2020. BimodalGaze: Seamlessly Refined Pointing with Gaze and Filtered Gestural Head Movement. In *ACM Symposium on Eye Tracking Research and Applications* (Stuttgart, Germany) *(ETRA '20 Full Papers)*. Association for Computing Machinery, New York, NY, USA, Article 8, 9 pages.

[24] Junbong Song, Sungmin Cho, Seung-Yeob Baek, Kunwoo Lee, and Hyunwoo Bang. 2014. GaFinC: Gaze and Finger Control interface for 3D model manipulation in CAD application. *Computer-Aided Design* 46 (2014), 239–245.

[25] Sophie Stellmach and Raimund Dachselt. 2013. Still Looking: Investigating Seamless Gaze-supported Selection, Positioning, and Manipulation of Distant Targets. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. 285–294.

[26] Mélodie Vidal, Andreas Bulling, and Hans Gellersen. 2013. Pursuits: Spontaneous Interaction with Displays Based on Smooth Pursuit Eye Movement and Moving Targets. In *Proceedings of the 2013 ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp '13)*. 439–448.

[27] Mélodie Vidal, Ken Pfeuffer, Andreas Bulling, and Hans W Gellersen. 2013. Pursuits: eye-based interaction with moving targets. In *CHI'13 Extended Abstracts on Human Factors in Computing Systems*. 3147–3150.

[28] Colin Ware and Harutune H. Mikaelian. 1986. An Evaluation of an Eye Tracker As a Device for Computer Input. *SIGCHI Bull.* 17, SI (May 1986), 183–188.