Enhanced Geometric Techniques for Point Marking in Model-Free Augmented Reality

Wallace S. Lages*
Center for HCI, School of Visual Arts
Virginia Tech
Blacksburg, Virginia, United States

Yuan Li†
Center for HCI, Dept. of Computer Science
Virginia Tech
Blacksburg, Virginia, United States

Lee Lisle‡
Center for HCI, Dept. of Computer Science
Virginia Tech
Blacksburg, Virginia, United States

Tobias Höllerer§
Dept. of Computer Science
University of California, Santa Barbara
Santa Barbara, California, United States

Doug A. Bowman¶
Center for HCI, Dept. of Computer Science
Virginia Tech
Blacksburg, Virginia, United States

ABSTRACT

Specifying points in three-dimensional (3D) space is an essential function in many augmented reality (AR) applications. When an environment model is not available, a straightforward solution is to perform geometric triangulation using two rays. However, naïve implementations suffer from low precision caused by technical limitations of AR devices and human motor constraints. To overcome these issues, we designed and evaluated two enhanced geometric techniques for 3D point marking. VectorCloud uses multiple rays to reduce the effects of pointing jitter, and ImageRefinement improves precision by allowing users to refine the 3D direction of the two rays on a static image of the target area. We conducted studies to understand the characteristics of these techniques in both an ecologically valid outdoor setting using a mobile AR display and in a more controlled setting using a virtual reality simulation. Our experiments demonstrate that both techniques improve the precision of 3D point marking, and that ImageRefinement is superior to VectorCloud overall. These results are particularly relevant in the design of mobile AR systems intended for use in large outdoor areas.

Index Terms: Human-centered computing—Mixed / augmented reality; Human-centered computing—Pointing; Human-centered computing—User interface design

1 INTRODUCTION

Specifying the three-dimensional (3D) location of points is an essential task in content creation for augmented reality (AR). We call this task point marking (or marking for short). 3D points in the real world can be used by AR systems to attach annotations, place virtual objects, delimit areas, build real-world references, and arrange user interface (UI) elements. For example, firefighters can use AR to mark safe passages or the location of civilians during an emergency. In architecture, marking physical landmarks can aid in measuring angles, distances or areas of interest.

Although point marking in AR has some similarity to 3D selection tasks, there are some important differences. The goal of 3D selection (or target acquisition) techniques is to identify one or more objects from a larger set [16]. In AR marking, however, the target point is not necessarily a discrete object that can be selected from the background. Often, users will wish to mark points on a surface, such as the ground or a wall. In addition, selection techniques require knowledge about the set of potential targets. Thus, they can only be used to mark real objects in AR when geometric information about the environment is available.

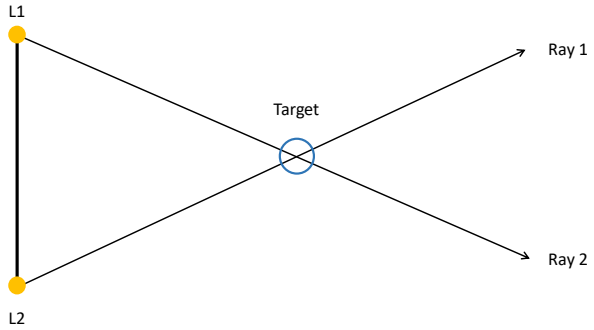

Figure 1: Geometric triangulation (top view). The target location is determined by the intersection of two rays (Ray 1 and Ray 2) cast by the user from locations L1 and L2, separated by a baseline.

Unfortunately, a model of the environment is not always available or accurate. Model reconstruction is limited by the range of AR devices' depth-mapping sensors, and even at near distances, reconstruction can fail without proper illumination or in the presence of reflective surfaces. This is especially true in wide-area AR (AR in large, outdoor environments), since points of interest are often too far away to map, and the environment is typically dynamic. Some systems try to address these limitations by using other sources such as offline models or aerial photographs [1, 11]. However, these solutions are limited to specific places (where such extra information is available) and times (changes in the real-world geometry will eventually make models inaccurate). Thus, techniques that can achieve point marking in model-free settings are desirable.

A classic approach to model-free marking is to triangulate the target position from different viewpoints [2, 20]. In this case, the target position can be determined by intersecting two or more 3D rays that pass through the target (Figure 1). We call this approach geometric triangulation. It requires the user to specify two rays from different locations. Compared to an approach where the user specifies a single ray and the distance to the target along that ray, triangulation avoids reliance on human distance perception, which is known to be inaccurate for virtual content at large distances [7, 12]. The performance of geometric triangulation, though, depends on the precision with which users can specify the two rays. This precision

*e-mail: wlages@vt.edu
†e-mail: yli92@vt.edu
‡e-mail: llisle@vt.edu
§e-mail: holl@cs.ucsb.edu
¶e-mail: dbowman@vt.edu

978-1-7281-4765-9/19/$31.00 ©2019 IEEE
DOI 10.1109/ISMAR.2019.00028

may be severely limited by human motor control issues (such as head or hand tremor) and/or technical issues (such as tracking jitter or drift).

In this paper, we describe two enhanced geometric triangulation techniques for model-free point marking. The VectorCloud technique uses multiple ray samples to reduce the influence of tremor and enables marking from more than two viewpoints. The ImageRefinement technique allows users to refine the direction of each ray by manipulating a cursor on an image of the target region. We performed studies to understand the properties and limitations of these techniques in both ecologically valid real-world outdoor settings using a mobile AR display, and in more controlled simulated environments using a Virtual Reality (VR) system. Our experiments show that both VectorCloud and ImageRefinement are more precise than the naïve geometric triangulation technique, and that ImageRefinement is the best technique overall. The video provided in the supplementary materials presents both the techniques and experiments.

2 Background and Related Work

Here we review related work on the limitations of human depth perception and previous geometric marking techniques.

2.1 Human Depth Perception

Two primary approaches may be considered for AR point marking: geometric triangulation and egocentric distance estimation. The latter approach only requires the user to specify the direction to the target position (a ray) and the distance of the target along that ray. However, the performance of egocentric distance estimation depends on the accuracy of human perception. The human ability to estimate egocentric distances varies depending on the distance being estimated and the method used for estimation. Cutting et al. [6] divided the perceptual space into three areas: personal space (under 2 meters), action space (up to 30 meters), and vista space (beyond 30 meters). Judgment in personal space is very good, since depth cues such as retinal disparity, convergence, and accommodation are effective at short viewing distances. In action space, the usefulness of these cues is greatly reduced. In vista space, the user must rely heavily on less efficient non-stereoscopic information, such as relative size, height in the visual field, and atmospheric effects.

AR devices also add additional perceptual challenges. Without an environment model, AR devices cannot properly render occlusion relationships with the real world, which would otherwise be a dominant depth cue to resolve distance ambiguities. Depth perception can also be distorted by the vergence-accommodation conflict, which occurs when the eyes converge at a virtual point but must focus on a screen located at a different distance. These and other issues are discussed in detail by Kruijff et al. [15]. On the other hand, AR overlays can be used to create additional pictorial depth cues or even distance estimation tools. However, in the absence of at least partially modeled geometry, such pictorial information has not been shown to significantly improve distance estimation [27].

Based on this background and our own experience, we argue that 3D point marking techniques based on human perceptual judgments in AR will have low accuracy and precision. Thus, we focus on geometric approaches not relying on human perception.

2.2 Geometric Marking Techniques

Among the first mobile applications for outdoor AR content creation were the Naval Research Laboratory BARS system [2] and the Tinmith-Metro modelling system [20]. These systems allowed users to sketch large architectural structures (such as buildings) using techniques inspired by CAD applications, in conjunction with physical triangulation and trilateration [1]. Baillot et al. [2] demonstrated the concept of geometric triangulation in an indoor environment, but never systematically studied the technique in wide-area AR. In Tinmith-Metro, users could create working planes, which could be specified without reference to any geometric information from the environment. For example, a user might create a plane containing the view direction vector and the gravity vector. Once created, users could mark points on the plane (e.g., by intersecting the plane with a ray) and manipulate existing objects along the plane [21]. Our techniques use refined versions of these early point construction techniques.

More recently, Polvi et al. [22] presented SlidAR, a system to mark points in AR. Similar to our techniques, it was designed to work without model information and uses Simultaneous Localization and Mapping (SLAM) to compute the camera pose, and eventually the target position. SlidAR first casts a ray through the camera center to define a line going through the target position. Then, the user can move a cursor along that line (from any available viewpoint) until it reaches the desired position. The geometric techniques we evaluated are similar, although the second step involves casting a second independent ray, instead of adjusting a cursor along the first ray. SlidAR was evaluated using targets within arm's reach, while our evaluation focused on distances in action and vista space (12.5m to 85.5m). The gesture annotation system presented by Nuernberger et al. is also conceptually similar to our geometric techniques [19]. However, their work focused on image-based scene reconstruction and therefore relied on a predefined scene model.

Figure 2: User's view of naïve geometric AR marking implementation.

3 Design of The VectorCloud Technique

We implemented a naïve version of the geometric triangulation approach (section 1) using the Microsoft HoloLens. To specify a 3D ray, the user rotates her head to align the center of a virtual crosshair with the point of interest in the real world. The crosshair is centered in the user view, as shown in Figure 2. Thus, 3D rays are defined by the user's head position and orientation, as tracked by the HoloLens SLAM system. Users press a button on a handheld controller to fix the first ray, then walk to a new location and repeat the process. The location of the target 3D point is calculated as the midpoint between the closest points on each ray, since two 3D rays may not intersect.

In our informal evaluation of this technique, we observed that marking was fairly accurate (the average of many attempts to mark the same point was near the target), but imprecise (there was high variability in the position of marked points over multiple attempts)[2]. This issue became worse at larger distances. We suspect that as distances increase, small angular errors in head orientation due to head tremor and noise in the tracking data can introduce substantial

[1] Process for determining a point position using distance instead of angles

[2] We use the term *accuracy* to refer to the error, relative to ground truth, of an individual marked point or the average error over many attempts, and the term *precision* to refer to the variability in point locations over many attempts.

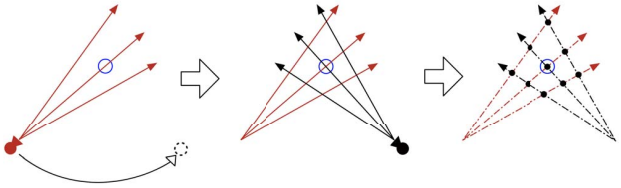

Figure 3: The VectorCloud technique. Users aim at the target from different locations while the rays are recorded. All pairs from two sets of rays are then intersected to obtain several sample positions.

errors in ray direction. Although it is easy to fixate the eyes on a distant target due to the vestibulo-ocular reflex [18], the same is not true of the head. Without the option to use eye tracking, our crosshair was attached to the head, and jitter of the crosshair relative to distant targets was obvious.

The accuracy of casting an individual ray depends on the stability of the user's head and their ability to press the button at a time when the crosshair is very close to the target. We thus set out to reduce the influence of human and system jitter by casting many rays in a multi-sampling approach. Assuming that the pointing error is random, consecutive attempts to mark the same target will lead to a cloud of points surrounding the desired target position. Therefore, computing a location estimate of the entire cloud should lead to a better estimate of the target point. We call this technique VectorCloud. The insight behind the VectorCloud technique is to assume that rays gathered from two different positions are independent, and so can be intersected in any order. Instead of computing the intersection of every pair of rays in sequence, we first store several samples of the target direction from each position. Then, we compute all possible intersections between the two sets of stored rays (Figure 3). We use the midpoint between the closest points on the two rays as the intersection.

By computing all intersections, the technique allows the user to mark from any number of arbitrary positions in space or even continuously sample as they walk. However, special care is needed in the case of continuous sampling and discrete sampling from more than two locations. When sampling from two discrete locations, ray intersections between pairs of rays from each location will always generate useful information, since every intersection will be valid. However, when using more than two discrete locations or continuous sampling it is necessary to choose which subsets to intersect. We implemented a simple solution that works in both cases: we subtract all pairs of consecutive vectors and split the samples into two sets at the point of maximal absolute difference. This solution is simple and works even when the user completely circles a target while sampling. However, it is not optimal, as it may intersect groups of rays with a short baseline. To prevent this, more sophisticated solutions could be used, like clustering the rays by angular difference. We leave this for future work.

Finally, once all intersections are computed, we can use any location measure to estimate the target position. The exact 3D shape of the distribution depends on the sampling error, the sampling positions, and the distance between them. We have explored different methods to estimate the location of the target, including using the mean, median, and mode; and performing asymmetric trimming before computing the mean. When rays come from two discrete points, the distribution of intersections has a positive skew, with most values concentrated closer to the user but a long tail beyond the target (Figure 4). This causes the mean to slightly overestimate the distance to the target position. Although the median can provide a more accurate estimate, the axis used for ordering needs to be aligned with the extremes of the distribution. Unfortunately, the correct axis is not well-defined when more than two viewpoints are

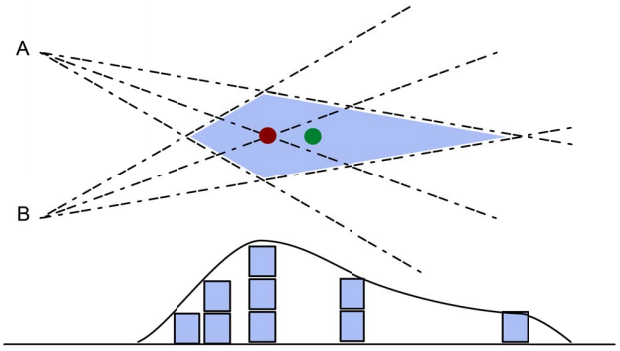

Figure 4: Top: top-down view of intersections generated from two viewpoints A and B. The red dot represents the target. From each viewpoint, we show three rays: one that hits the target and two on either side with the same amount of angular error. Due to the long tail, the mean (green dot) falls beyond the target. Bottom: a histogram showing the number of intersections computed at different distances.

used (or during continuous sampling). Thus, we used the mean in our implementation, since it is well defined in every case.

From the user's perspective, VectorCloud follows the same steps as the naïve geometric technique, except that the user needs to hold down the controller button for a few seconds at each location to gather multiple ray samples. Since we did not impose a limit on how many samples the user can gather, we assigned a second button to finalize the overall marking process.

4 Evaluation of VectorCloud

To quantify the hypothesized performance benefits of VectorCloud, we performed a within-subjects experiment comparing it to the naïve geometric technique. Although we originally implemented VectorCloud for the HoloLens in AR, we performed the experiment in an AR simulation using VR technology to avoid the tracking errors seen when using the HoloLens outdoors. With more precise tracking in VR, we reduced the variance introduced by hardware limitations and evaluated the techniques under more controlled circumstances. This approach, known as Mixed Reality Simulation, has been used in a variety of prior AR experiments in which either experimental control was critical or technological limitations made the use of real AR systems impractical [3, 10, 17].

In VR, we rendered the virtual crosshair and other simulated AR graphics with a customized shader to prevent occlusion by other geometry and with a transparency value close to their actual appearance in AR. We built a testing virtual environment containing six target lampposts at distances of 12.5m, 26.7m, 40.2m, 55.2m, 70.3m, and 85.5m. These distances were chosen to replicate a real-world scene we used for the informal AR evaluation (Figure 2). Since both techniques require the user to sample one or more rays from two separate locations, we marked two spots on the ground that were 1.92m from each other as the marking locations. The length of the baseline was not varied during the experiment because its effect on the techniques' performance is equivalent to that of target distance: using a longer baseline is equivalent to marking a closer target and vice versa. The VR scene is shown in Figure 5. Due to limited resolution and dynamic range of the VR display, we placed a small red sphere at the tip of each lamppost to indicate the exact point that should be marked (Figure 5).

Each participant marked each target eight times with each technique, resulting in 96 data points per participant. The same randomized target sequence was used for all participants and conditions. As participants completed one marking, they could see the number of the next target in the lower part of the display. The final position

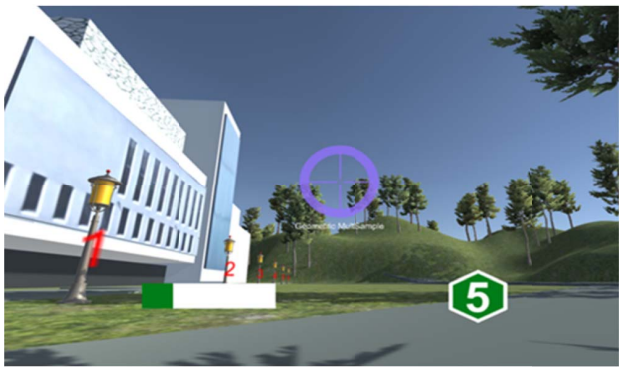

Figure 5: The VR environment used in the evaluation of VectorCloud.

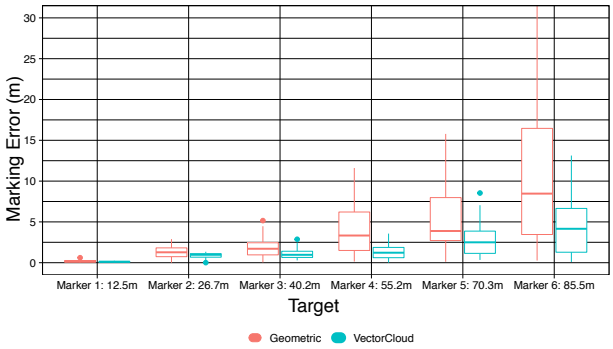

Figure 6: Marking error results with geometric and VectorCloud techniques.

of the marker was not displayed so that users could not learn from previous trials, and we counterbalanced the presentation order of the techniques. With VectorCloud, we required users to collect 300 samples from each position. When the system collected 300 samples, it notified the user by playing a sound and stopped recording more samples. The number of samples was chosen in an attempt to balance between speed and accuracy.

Our hypothesis was that VectorCloud would have similar accuracy but increased precision when compared to the naïve geometric technique. We also expected VectorCloud to improve user experience by relaxing the pointing precision requirement.

4.1 Apparatus

We used a consumer version HTC Vive head-mounted display (HMD). The Vive has two screens, each with a resolution of 1080x1200 pixels. The total horizontal field of view is 110 degrees. It was tracked with six degrees-of-freedom by the hybrid inertial-optical Lighthouse system. We also used a wireless Xbox controller for input. We used the 'RB' (right shoulder) button to cast rays in both techniques. In the VectorCloud technique, the 'LB' (left shoulder) button was used to complete the trial. The software used in the experiment was written in Unity3D.

4.2 Participants and Procedure

We recruited six graduate students and four undergraduate students (three female). Although ten participants is a small number in absolute terms, it is not uncommon for experiments that hypothesize a large difference between conditions [8, 24]. A post-hoc power and effect size analysis validated this design choice (see Section 4.3). Ages ranged from 21 to 39 years old, with the mean being 25.33. Most participants had used VR or AR at least once or twice before, while one participant had no prior experience with the technology. Nine of the participants were right-eye dominant, while one participant was left-eye dominant. The experiment was approved by the university's Institutional Review Board.

After providing informed consent, participants were asked to complete a background questionnaire containing demographic information. Next, we measured each participant's interpupillary distance (IPD) using a pupilometer (Sunwin digital meter) and adjusted the IPD value in the Vive accordingly. Because we wanted the participant to aim at the target with only the dominant eye, a Porta test [23] was included at the beginning of the session to identify the user's dominant eye. Based on the result, we blocked the display in the HMD for the non-dominant eye with a black cloth to minimize fatigue. After being introduced to the device, task, and techniques, participants were trained until they had sufficient confidence for the tasks. To ensure accurate tracking, we paid special attention during the experiment that the system never lost tracking of the HMD. After participants completed all trials, a post-questionnaire was presented to gather qualitative feedback about the techniques.

4.3 Results

Figure 6 shows the absolute marking error of the naïve geometric technique and the VectorCloud technique when marking targets 1-6. Both techniques were accurate, although the naïve geometric technique tended to result in larger error than the VectorCloud technique, especially as target distance increased. We also note that the accuracy of the geometric technique was higher than what we observed during our informal outdoor evaluation. This supported our suspicion that tracking errors were limiting the accuracy of this class of techniques. More importantly, the variance for VectorCloud was smaller than that of the geometric technique for each target, and this difference increased with target distance.

When analyzing the marking error data, the hypothesis of equal variances was rejected ($p < 0.008$) by a heteroscedastic version of the Morgan-Pitman test (using the HC4 estimator) [5, 26]. We fitted a polynomial multiple regression on the variance data (adjusted $R^2 = 0.97$, standard error $= 0.82$). The variability of both techniques appears to grow with the square of the distance. The difference between the regressed coefficients for the first-degree term is 7.13 ($p < .0001$), and for the second-degree term 3.1 ($p < 0.001$). The quadratic model had a significantly better fit than a line ($p < 0.002$). The standard deviation of VectorCloud for the last target was 3.5 times smaller than that of the geometric technique, which represents a considerable gain in precision.

To confirm the statistical validity of the experiment, we performed a post-hoc analysis of the achieved power for a test of variance equality using G*Power [9]. The calculated $(1-\beta)$ probability with $alpha = 0.05$ and two-tailed test was 94%. This means that our experiment had a significant chance of detecting a true difference between the techniques even with the small sample size used. The reason is the large effect size (12.2 variance ratio) which becomes more evident at the longer distances used in our experiment. We also inspected the variance on the fronto-parallel plane and, as expected, this variance was visibly smaller with VectorCloud. Figure 7 shows a comparison between the view plane variability of the geometric and VectorCloud techniques accumulated for all targets.

4.4 Discussion

The multi-sample approach of VectorCloud significantly improved the marking precision by computing a better estimate of the target position. For both techniques, the standard error increases with the square of the target distance. Given a constant angular error, the amount of linear error in a plane parallel to the view plane at the target distance is $\tan(\varepsilon) \times d$, leading to an increase in the variance

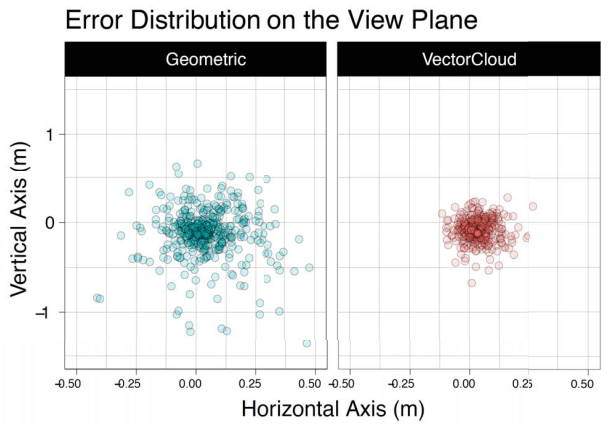

Figure 7: Variability on the view plane is also reduced with the VectorCloud technique.

Figure 8: User's view of ImageRefinement in the HoloLens. The user adjusts the ray by moving the crosshair in the 2D image.

by the square of this factor. Similarly, while developing a model for distal pointing, Kopper et al. [14] found that movement time increased with the square of the task difficulty index (which was modeled as an inverse function of angular size).

Although the magnitude of the pointing errors quickly increases with distance, the reduction in variability due to VectorCloud's multisampling made the actual error much smaller than what would be predicted by a model based on information about human head orientation stability. Skavenski et al. [25] report that head rotations, even while participants try to be as still as possible, can reach angles of 0.75 degrees. At an 85-meter distance, this would correspond to distance estimation errors of up to 40 meters (far below the VectorCloud standard error of 5 meters and closer to the geometric standard error of 20 meters).

Using the mean as a measure of location of the point distribution was satisfactory. For both techniques, the difference between the mean and the veridical distance was well within the variability created by head and system jitter (approximately 22% of standard error).

While the improvement from the naïve geometric technique to VectorCloud is promising, we still observed several limitations of the technique itself. As it does not provide any feedback on the sampled rays, the user has no way to improve accuracy, because she does not know how many samples she needs or how precise she has to be. In addition, the user does not have fine-grained control of the final ray direction.

5 Design of The ImageRefinement Technique

As we noted in Section 3, the key determinant of accuracy in geometric marking is the precision of specifying the ray directions for triangulation. The precision of ray specification in the naïve geometric technique suffers because of head and tracker jitter. While VectorCloud reduces the impact of head tremor, it does not eliminate it completely. Another way to improve geometric point marking is to decouple the specification of the rays' directions from the position and orientation of the user's head, so that each ray is more accurate. This led us to the creation of a technique based on progressive refinement [13].

In the ImageRefinement technique, the user first aims roughly at the target using the head and presses the controller button. Instead of generating a ray, the technique captures an image using the AR headset's onboard camera. We crop and enlarge the image so that only the central region around the targeting reticle is shown, resulting in an image with the appearance of a 2.5x zoom. We display the image in the headset at a comfortable distance and display a crosshair at the center of the image (Figure 8). Next, in a refinement stage, the user adjusts the direction of the ray by manipulating the crosshair in 2D image coordinates to specify the location of the target in the image. Crosshair manipulation uses an analog joystick on a handheld input device.

The origin of the ray is the position of the user's head at the time the image was taken. The direction of the ray is computed by considering the forward vector of the camera at the time the image was captured, the horizontal and vertical offset of the cursor in pixels from the center of the image and the camera's horizontal and vertical fields of view. In our implementation, we define a vector in camera coordinates from the camera to the offset cursor. Transforming this vector into world coordinates results in the refined ray. The user then repeats the process from a second location, and the intersection between the two rays is calculated as in the naïve geometric technique. We call this technique ImageRefinement (IR).

IR results in two high-quality rays, and gives users control over the final direction of each ray. Because of the refinement step, it does not require precise aiming using the head. It also allows users to rest their head and neck muscles while refining the ray, since the system recalls the user's head position when they took the snapshot. Unlike VectorCloud, users of IR do not have to keep their heads in the same position for an extended period of time.

6 Comparison of Enhanced Techniques

Having established that it is possible to increase the precision of geometric marking (Section 4), we also wanted to compare our two enhanced techniques to explore the trade-offs between them. We hypothesized that IR would result in higher precision than VectorCloud due to its ability to make fine-grained adjustments to ray direction and its lack of reliance on head stability. However, we questioned how large this effect would be, and whether IR would result in a higher hit rate when we defined an accuracy threshold around targets.

The use of the VectorCloud technique is simple. Similar to the naïve geometric technique, it involves only looking at the target and pressing buttons to define the ray. IR, on the other hand, is more complex, as it requires an additional refinement step for each ray. Based on these observations, we hypothesized that VectorCloud would be faster than IR, and that some users might prefer its simplicity. However, the refinement step also gives IR users more direct control over the result of marking (potentially leading to higher confidence), and it does not require users to keep their heads still during use (potentially leading to greater comfort). Thus, we hypothesized that most users would prefer IR overall.

Additionally, we wanted to understand how the two main features

of IR—decoupling head position from ray direction and providing a zoomed image—affected performance individually. We therefore created an ImageRefinement "no zoom" variant (IRNZ) in which the image shown to the user was not digitally zoomed. The three techniques (VectorCloud, IR, IRNZ) were then evaluated in two controlled experiments.

6.1 AR Simulation Experiment

We implemented IR and IRNZ in the VR application used to evaluate VectorCloud (Section 4). This allowed us to compare the three techniques in a controlled setting.

Since we had already learned how distance affects performance on the marking task, we used only three distances in this experiment—26.7m, 55.2m, and 85.5m (targets 2, 4, and 6 from the first study). Thus, our experiment was a 3 (technique) $\times$ 3 (distance) within-subjects repeated-measures experiment. The dependent variables were marking error (absolute distance from marked point to target), hit rate (percentage of marking attempts with an error less than 4.25 meters, which is 5% of the farthest target's distance), time to mark the target, and usability as measured by a modified SUS questionnaire.

6.1.1 Apparatus

As in the experiment described in section 4.1, we used a consumer version HTC Vive HMD with Lighthouse tracking and a wireless Xbox controller for input. IR and IRNZ were both controlled by pressing the 'RB' button to take a picture of the scene. The image was captured by a virtual camera positioned halfway between the left- and right-eye cameras used to render the scene. We rendered the image so that when it was displayed in the HMD, detail in the image was maintained; it was always a 640×360 image regardless of zoom level. Users could then use the left joystick to control the cursor. VectorCloud was controlled by holding down the RB button until the system indicated that the user had collected 300 samples (by playing a sound) and stopped collecting more samples. The number of samples was chosen to match the earlier VectorCloud experiment.Taking 300 samples took 2.5 seconds. After the second set of samples was gathered, the user pressed the 'A' button to confirm the input and complete the marking. The software was developed in Unity3D.

6.1.2 Participants and Procedure

This experiment gathered data from 24 participants (five female) with a mean age of 24.5 (standard deviation of 4.8). All of them had prior experience with AR or VR, with eight of the participants utilizing these technologies regularly. Six participants used contact lenses, nine used glasses, and the remaining nine had good uncorrected vision. The experiment was approved by the university's Institutional Review Board.

Participants first read and signed an informed consent describing the experiment and possible side-effects. After that, the participants filled out a background questionnaire on their previous experiences with AR/VR as well as demographic data. Next, we gathered participants' physiological data by measuring their interpupillary distance (IPD) and determining their dominant eye, using the same tools as in section 4.2.

We instructed participants that they should emphasize accuracy over speed. We then presented the first technique (order of technique presentation was counter-balanced using a Latin square). A training procedure guided the participants through marking three targets. Once they confirmed that they were comfortable with the technique's use, we started the formal trials. Participants marked the virtual lampposts in a pseudorandom order that included marking targets 2, 4, and 6 six times each for a total of eighteen trials. If a marking was more than 4.25 meters away from the actual position of the target,

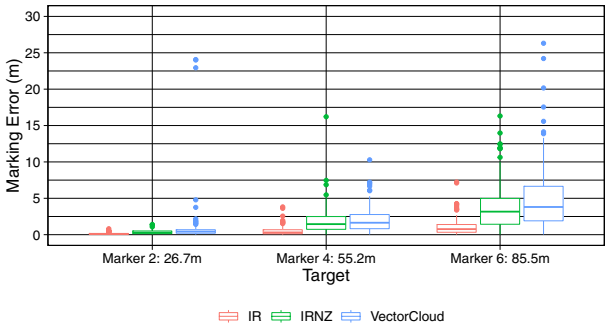

Figure 9: Marking error in the AR simulation experiment.

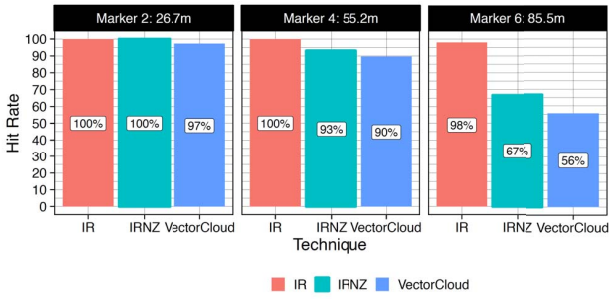

Figure 10: Hit rate in the AR simulation experiment.

the system indicated a "miss" by playing a sound. Otherwise, a positive feedback tone was played.

After each technique, participants completed a modified system usability scale (SUS) questionnaire [4] about the technique they just finished. We replaced question 5 on the original SUS ("I found the various functions in this system were well integrated") with "I was physically comfortable using this technique" since the original question did not apply to an individual technique. We also asked participants what strategies they used and why, and what comments they had about the technique, if any.

After completing the same steps for the second technique, participants filled out a final questionnaire asking them to specify which technique they preferred, and to list any issues they had while using the techniques.

6.1.3 Results

Figure 9 shows the marking error for each target and technique combination. We used a linear mixed model to analyze the effects of the independent variables, including fixed effects for technique and target and random effects for subject. The subject variance in the fitted model was small (0.2285) but indicated some benefit of including subject as a random effect.

We first analyzed the data using a model including an interaction term and found significant effects of technique and target, but no interaction effect. When we removed the interaction term from the model, we again found significant main effects of technique ($F(2,34.5) = 35.85, p = 3.8e^{-09}$) and target ($F(2,48.3) = 43.37, p = 1.7e^{-11}$). Furthermore, with post-hoc pairwise analysis we found significance between all pairs of techniques and all pairs of targets ($p < .001$). All t-test statistics were adjusted using Satterthwaite's method to estimate the effective degrees of freedom. The differences between the techniques are visible in Figure 9, with

	Target 2	Target 4	Target 6	Overall
IR	13.30	13.46	14.24	13.66
IRNZ	13.82	14.5	16.72	15.02
VC	16.31	17.13	17.80	17.08
Overall	14.48	15.03	16.26	

Table 1: The mean time for marking a target in VR for each target & technique, as well as overall means across targets or techniques.

IR achieving the highest accuracy, followed by IRNZ, and then VectorCloud. The hit rate data (Figure 10) also reflects this, with IR achieving a 98% hit rate at 85.5 m.

Timing measurements were only considered for attempts that resulted in a 'hit'. This ensured we were only measuring attempts where the participants were working to be as accurate as possible. The mean times for hits with each technique and target combination can be seen in Table 1, along with overall means for each technique. After performing a similar analysis as for marking error, we found a significant main effect of both technique ($F(2,23.2) = 27.9, p = 6.7e^{-07}$) and target ($F(2,23.2) = 11.5, p = 3.5e^{-04}$) on the time to mark a target. Post-hoc analysis with Satterthwaite's method revealed significant differences between all pairs of techniques ($p < .05$).

We also performed an ANOVA on the effects of technique on total SUS score. We found that the techniques were significantly different from each other in terms of SUS score ($F(2,69) = 12.37, p = 2.56e^{-05}$). Furthermore, post-hoc comparisons with Tukey's adjustments revealed that all pairs of techniques were significantly different. The mean score for IR was 89.7, while the means for IRNZ and VC were 74.8 and 63.2 respectively.

6.1.4 Discussion

Our findings show that ImageRefinement results in significantly higher accuracy than VectorCloud. This can be seen even when the benefit of zooming is removed from the technique. However, including zoom does have an effect on the technique, as IR is significantly more accurate than IRNZ, and the difference between the two is larger than the difference between IRNZ and VectorCloud.

The timing data show that despite its two-step process, ImageRefinement can still be faster than VectorCloud. However, it should be noted that we tuned these techniques for marking accuracy; VectorCloud could potentially be faster if we used a lower number of samples.

IR also appealed more to users. Users gave it a higher score than VectorCloud by an average of 26 points on the 100-point SUS scale, with 23 of the 24 participants indicating that IR was their preferred technique. The last participant indicated they preferred VectorCloud, but their SUS data gave IR a score of 97.5 and VectorCloud a score of 90. Users commented that they liked how IR allowed "the freedom to fail," since they could refine their rays using the crosshair on the image. The qualitative feedback for VectorCloud indicated that nine users tried to "hold their breath" while using the technique. Furthermore, two expressed frustration with the technique since it did not give them feedback while using it, and two stated that the technique caused neck pain.

From this experiment we conclude that, in an idealized setting, the ImageRefinement approach was better than VectorCloud for marking tasks requiring high levels of accuracy, and that IR also improved the overall user experience, producing a feeling of greater control. The zoom feature in this study was also beneficial, allowing rays to be even more accurate.

6.2 Outdoor AR Experiment

Since the AR simulation in the study described in Section 6.1 was less than realistic in some ways (e.g., high-resolution camera images, opaque display, highly accurate tracking), we also wanted to explore how our enhanced marking techniques work in a real-world AR setting. We therefore ported the VR application code to work on the HoloLens. While VectorCloud ported over easily, we had to adjust the method of calculating the rays for IR and IRNZ, since we now were working with a real camera. We used the HoloLens API to transform pixel coordinates from the camera image into a ray based on the factory-measured camera intrinsic parameters and view transformation.

The HoloLens has a 2.4-megapixel camera, meaning that zoomed images are quite blurry. To maintain consistency with the previous VR evaluation, we cropped the images from the HoloLens' camera to simulate zooming. However, since the camera's resolution is only 2.4 megapixels, this results in a fairly blurry "zoomed" image. In the VR implementation, we generated a 640×480 pixel image for the user to refine their target. In AR, however, we had to crop the camera's output to a 400×225 pixel image for IRNZ, while IR cropped the output to a 208×117 pixel image. In all other respects, we designed this experiment to be as similar as possible to the AR simulation experiment (Section 6.1) so that we could compare the differences between the two.

6.2.1 Apparatus

We used the Microsoft HoloLens along with a a bluetooth Microsoft Xbox One controller. Due to issues with the version of Unity we were using at the time, in combination with the HoloLens and the controller, we had to change the control of the cursor in the ImageRefinement techniques to use the 'A', 'B', 'X', and 'Y' buttons as directional inputs (for 'down', 'right', 'left', and 'up' respectively). Capturing an image with ImageRefinement techniques took about 0.5 seconds, as opposed to the instantaneous image capture in the AR simulation. Otherwise, the control scheme was the same as the VR experiment. VectorCloud was implemented in the same way as in section 4, except that we required 200 samples instead of 300, since it took the same amount of time to collect 200 samples on the HoloLens as it took to collect 300 samples in the VR experiment.

To improve the contrast of the HoloLens display outdoors, we built a "sunglasses" adapter which allowed the crosshair and the experiment instructions to be visible in daylight. The adapter was attached to the exterior of the device and was built using duct tape, Lego pieces, and shaded film.

In addition, we used an Apple iPad to communicate with the HoloLens during the experiment. This gave the researcher feedback from the system on how the participant was progressing in the experiment through a text console. The iPad also gave the experimenter a way to reset the HoloLens spatial anchor (a known position that is used by the HoloLens to help maintain a fixed coordinate system) and correct any large drift caused by using the HoloLens outdoors.

6.2.2 Participants and Procedure

We gathered data from eighteen participants (four female) with a mean age of 21.5 (standard deviation of 2.18). All of them had a background in engineering and only one of the eighteen had no experience with AR/VR displays. Five participants used contact lenses, seven used glasses, and six had perfect vision. The experiment was approved by the university's Institutional Review Board.

Participants first read and signed an informed consent describing the experiment and potential side-effects. After that, the participants filled out a background questionnaire on their previous experiences with AR/VR as well as demographic data. Once these were completed, the participants completed the Microsoft setup program for the HoloLens that calibrates the system for each individual user and gets them accustomed to the display.

Participants were taken to the testing area, which was in a nearby courtyard (Figure 2). We defined two positions on the ground with a baseline of 1.92m for the user to stand on while marking the target. The lamp posts found in this area were used as targets, and were at

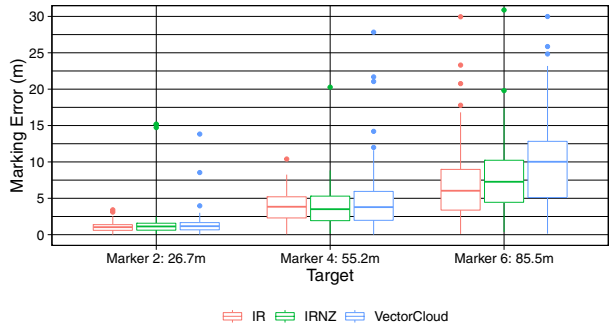

Figure 11: Marking error in the outdoor AR experiment.

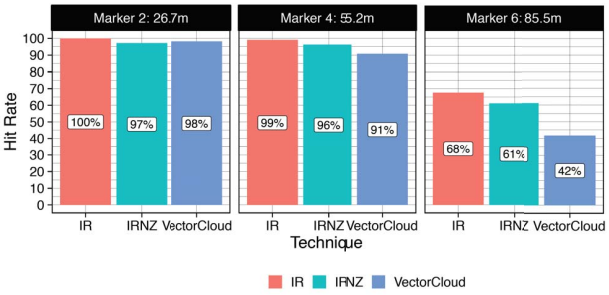

Figure 12: Hit rate in the outdoor AR experiment.

the same distances as in the AR simulation experiment (12.5, 26.7, 40.2, 55.2, 70.3, and 85.5 meters). The participants were instructed to emphasize accuracy over speed for all trials. We then introduced the first technique (order of techniques was counterbalanced using a Latin square) and guided them through a training period where they marked three targets. Once the participants felt comfortable with the technique, we began the formal trials. As in the AR simulation experiment, participants had to mark targets 2, 4, and 6 six times each, and they were given auditory feedback by the system when they hit or missed the target. For this experiment the range considered to be a 'hit' was extended to be 10% of the maximum target distance (8.5 meters), because pilot testing revealed that there would be excessive misses with the 5% threshold used in the prior experiment. Between every other trial, participants were asked to gaze at a specific target while standing at a specific position, so that the experimenter could reset the virtual anchor. This was done to actively correct any HoloLens tracking drift and reset the coordinate system so that we could minimize errors due to tracking.

After each technique, participants filled out the same questionnaire as in the previous experiment. After all three techniques had been completed, they completed the same post-experiment questionnaire as in the prior study.

6.2.3 Results

Figure 11 shows the marking error for each technique target pair. Again, we used a linear mixed-model to analyze the effects of the independent variables. With the interaction term included in the model, we found only a significant main effect of target ($F(2,137.5) = 349.4, p = 2.2e^{-16}$). As before, the interaction term between technique and target was not significant, so we removed it from the model for further analysis. Without the interaction term, we found significant main effects of both technique ($F(2,28.4) = 6.35, p = .0052$) and target ($F(2,932.3) = 353.8, p = 2.2e^{-16}$). A post-hoc pairwise analysis found that IR was significantly more accurate than VectorCloud ($T(24.78) = 4.199, p = 2.93e^{-05}$), and that IRNZ was significantly more accurate than VectorCloud ($T(24.78) = 2.5, p = .0126$). However, IR was not significantly different from IRNZ ($p = .0893$). Hit rates for each target-technique combination can be seen in Figure 12.

As in the prior experiment, timing measurements were only considered for attempts that resulted in a 'hit.' Mean times for a hit for each technique and target combination can be seen in Table 2, along with overall means for each technique and target separately. A mixed model using random slopes for target and technique indicated there was a significant main effect of both technique ($F(2,17.5) = 14.2, p = .00022$) and target ($F(2,33.07) = 80.94, p = 11.8e^{-13}$) on marking time. Post-hoc analysis with Satterthwaite's method revealed that IR was significantly slower than VectorCloud ($T(17) = 3.57, p = .00235$) and that IRNZ was also significantly slower than VectorCloud ($T(16.9) = 4.6, p = .000248$). As in the AR simulation experiment, it should be noted that marking time for VectorCloud is linked to the number of samples it needs from each position.

On the SUS questionnaire, IR scored a mean of 84.7 with a standard deviation of 14.4, IRNZ scored a mean of 73.5 with a standard deviation of 17.3, and VectorCloud scored a mean of 72.4 with a standard deviation of 14.5. An ANOVA found a significant effect of technique on score ($F(2,51) = 3.53, p = .0367$). However, post-hoc comparisons with Tukey's adjustments found no significant difference between any pair of the three techniques.

6.2.4 Discussion

As in the AR simulation experiment, the ImageRefinement approach resulted in significantly better accuracy than VectorCloud. However, the differences in this study were much smaller, likely due to the limitations of current real-world AR systems. Four participants complained about the image resolution by saying the image was fuzzy or that IRNZ had a sharper image than IR. In addition, the general performance decrease compared to the VR experiment seems to indicate that the HoloLens still had issues tracking the scene correctly. We tried to compensate for this by regularly resetting the anchor, but some small amount of error likely occurred regardless. The effects of tracking issues could potentially be isolated from the effects of poor camera resolution by looking into how tracking affected VectorCloud, since it does not use the camera at all. However, there might be additional factors also affecting both techniques. We leave this for future work.

The marking time for VectorCloud was similar to the prior experiment, as expected. However, both ImageRefinement techniques took much longer, which resulted in VectorCloud being significantly faster in this study. We attribute the increased time for the ImageRefinement techniques to the slow speed of the HoloLens image capture function, increased difficulty in seeing whether the cursor was over the target with the fuzzy and semi-transparent image, and the button-based control scheme. Again, we note that VectorCloud could be tuned for quicker marking (but lower precision) by using fewer samples.

We did not find significant differences among the techniques

	Target 2	Target 4	Target 6	Overall
IR	19.77	23.65	25.46	22.96
IRNZ	20.53	24.47	29.40	24.80
VC	16.36	18.59	20.05	18.33
Overall	18.89	22.24	24.97	

Table 2: The mean time for marking a target in AR for each target & technique, as well as overall means across targets or techniques.

based on the SUS data. However, participants' qualitative feedback was similar to the AR simulation study. Three participants indicated that they had to strain to keep VectorCloud on target and five participants indicated that they held their breath while using VectorCloud. Furthermore, most participants (13) still preferred IR overall, while three participants indicated a preference for VectorCloud, and two preferred IRNZ. Preference was not as overwhelmingly in favor of IR as in the prior experiment, which might be explained by the change in camera resolution and control scheme.

7 Conclusions and Future Work

The specification of 3D points is a fundamental task in AR systems, and it is not always possible to rely on accurate models of the real-world environment to aid 3D marking. Current AR devices can build an environment model using depth sensing, but only within a limited range. In this work, we have explored three variants of the geometric triangulation approach to 3D marking that rely on neither an environment model nor human depth perception.

Our findings indicate that geometric marking can be reasonably accurate at distances up to 85 meters, but that the naïve geometric technique suffers from a lack of precision, especially at larger distances. VectorCloud is more precise than the basic geometric technique, but the lack of user control over the final ray direction leaves room for further improvement. Users perform better with the ImageRefinement approach and also prefer it to VectorCloud.

Our results can be applied to indoor or outdoor AR applications where environment model information is unavailable or unreliable. In the future, we will seek to build on these techniques in real-world wide-area AR applications. We are currently exploring the use of enhanced geometric marking in AR annotation and tour planning, architectural massing studies, and multi-user AR calibration.

Acknowledgments

The authors gratefully acknowledge funding support from the Immersive Sciences program in the Office of Naval Research. The authors also thank Feiyu Lu, the study participants, and the anonymous reviewers.

References

[1] C. Arth, C. Pirchheim, J. Ventura, D. Schmalstieg, and V. Lepetit. Instant outdoor localization and slam initialization from 2.5 d maps. *IEEE Trans. Vis. Comput. Graph.*, 21(11):1309–1318, 2015.

[2] Y. Baillot, D. Brown, and S. Julier. Authoring of physical models using mobile computers. In *Proceedings Fifth International Symposium on Wearable Computers*, pp. 39–46. IEEE, 2001.

[3] D. A. Bowman, C. Stinson, E. D. Ragan, S. Scerbo, T. Höllerer, C. Lee, R. P. McMahan, and R. Kopper. Evaluating effectiveness in virtual environments with mr simulation. In *Interservice/Industry Training, Simulation, and Education Conference*, vol. 4, 2012.

[4] J. Brooke et al. Sus-a quick and dirty usability scale. *Usability evaluation in industry*, 189(194):4–7, 1996.

[5] F. Cribari-Neto. Asymptotic inference under heteroskedasticity of unknown form. *Computational Statistics & Data Analysis*, 45(2):215–233, 2004.

[6] J. E. Cutting and P. M. Vishton. Perceiving layout and knowing distances: The integration, relative potency, and contextual use of different information about depth. In *Perception of space and motion*, pp. 69–117. Elsevier, 1995.

[7] A. Dey, A. Cunningham, and C. Sandor. Evaluating depth perception of photorealistic mixed reality visualizations for occluded objects in outdoor environments. In *Proceedings of the 17th ACM Symposium on Virtual Reality Software and Technology*, pp. 211–218. ACM, 2010.

[8] O. Erat, W. A. Isop, D. Kalkofen, and D. Schmalstieg. Drone-augmented human vision: Exocentric control for drones exploring hidden areas. *IEEE Transactions on Visualization and Computer Graphics*, 24(4):1437–1446, April 2018. doi: 10.1109/TVCG.2018.2794058

[9] F. Faul, E. Erdfelder, A. Buchner, and A.-G. Lang. Statistical power analyses using g* power 3.1: Tests for correlation and regression analyses. *Behavior research methods*, 41(4):1149–1160, 2009.

[10] J. L. Gabbard, J. E. Swan, and D. Hix. The effects of text drawing styles, background textures, and natural lighting on text legibility in outdoor augmented reality. *Presence: Teleoperators & Virtual Environments*, 15(1):16–32, 2006.

[11] T. Höllerer, J. Wither, and S. DiVerdi. anywhere augmentation: Towards mobile augmented reality in unprepared environments. In *Location Based Services and TeleCartography*, pp. 393–416. Springer, 2007.

[12] J. A. Jones, J. E. Swan II, G. Singh, E. Kolstad, and S. R. Ellis. The effects of virtual reality, augmented reality, and motion parallax on egocentric depth perception. In *Proceedings of the 5th symposium on Applied perception in graphics and visualization*, pp. 9–14. ACM, 2008.

[13] R. Kopper, F. Bacim, and D. A. Bowman. Rapid and accurate 3d selection by progressive refinement. In *2011 IEEE Symposium on 3D User Interfaces (3DUI)*, pp. 67–74. IEEE, 2011.

[14] R. Kopper, D. A. Bowman, M. G. Silva, and R. P. McMahan. A human motor behavior model for distal pointing tasks. *International journal of human-computer studies*, 68(10):603–615, 2010.

[15] E. Kruijff, J. E. Swan, and S. Feiner. Perceptual issues in augmented reality revisited. In *2010 IEEE International Symposium on Mixed and Augmented Reality*, pp. 3–12. IEEE, 2010.

[16] J. J. LaViola Jr, E. Kruijff, R. P. McMahan, D. Bowman, and I. P. Poupyrev. *3D user interfaces: theory and practice*. Addison-Wesley Professional, 2017.

[17] C. Lee, S. Bonebrake, D. A. Bowman, and T. Höllerer. The role of latency in the validity of ar simulation. In *2010 IEEE Virtual Reality Conference (VR)*, pp. 11–18. IEEE, 2010.

[18] R. Lorente de Nó et al. Vestibulo-ocular reflex arc. *Arch Neurol Psychiatry*, 30(2):245–91, 1933.

[19] B. Nuernberger, K.-C. Lien, L. Grinta, C. Sweeney, M. Turk, and T. Höllerer. Multi-view gesture annotations in image-based 3d reconstructed scenes. In *Proceedings of the 22nd ACM Conference on Virtual Reality Software and Technology*, pp. 129–138. ACM, 2016.

[20] W. Piekarski and B. H. Thomas. Interactive augmented reality techniques for construction at a distance of 3d geometry. In *Proceedings of the workshop on Virtual environments 2003*, pp. 19–28. ACM, 2003.

[21] W. Piekarski and B. H. Thomas. Augmented reality working planes: A foundation for action and construction at a distance. In *Proceedings of the 3rd IEEE/ACM International Symposium on Mixed and Augmented Reality*, pp. 162–171. IEEE Computer Society, 2004.

[22] J. Polvi, T. Taketomi, G. Yamamoto, A. Dey, C. Sandor, and H. Kato. Slidar: A 3d positioning method for slam-based handheld augmented reality. *Computers & Graphics*, 55:33–43, 2016.

[23] G. d. Porta. *De refractione optices parte : libri novem ...* Apud Io. Iacobum Carlinum & Antonium Pacem, Neapoli, 1593.

[24] I. Poupyrev, S. Maruyama, and J. Rekimoto. Ambient touch: Designing tactile interfaces for handheld devices. In *Proceedings of the 15th Annual ACM Symposium on User Interface Software and Technology*, UIST '02, pp. 51–60. ACM, New York, NY, USA, 2002. doi: 10.1145/571985.571993

[25] A. Skavenski, R. Hansen, R. M. Steinman, and B. J. Winterson. Quality of retinal image stabilization during small natural and artificial body rotations in man. *Vision research*, 19(6):675–683, 1979.

[26] R. R. Wilcox. *Understanding and applying basic statistical methods using R*. John Wiley & Sons, 2016.

[27] J. Wither and T. Hollerer. Pictorial depth cues for outdoor augmented reality. In *Ninth IEEE International Symposium on Wearable Computers (ISWC'05)*, pp. 92–99. IEEE, 2005.