Comparing Techniques for Visualizing Moving Out-of-View Objects in Head-mounted Virtual Reality

Uwe Gruenefeld*, Ilja Koethe† Daniel Lange‡, Sebastian Weiß§, Wilko Heuten¶
University of Oldenburg OFFIS - Institute for IT

Abstract

Current head-mounted displays (HMDs) have a limited field-of-view (FOV). A limited FOV further decreases the already restricted human visual range and amplifies the problem of objects receding from view (e.g., opponents in computer games). However, there is no previous work that investigates how to best perceive moving out-of-view objects on head-mounted displays. In this paper, we compare two visualization approaches: (1) Overview+detail, with *3D Radar*, and (2) Focus+context, with *EyeSee360*, in a user study to evaluate their performances for visualizing moving out-of-view objects. We found that using *3D Radar* resulted in a significantly lower movement estimation error and higher usability, measured by the system usability scale. *3D Radar* was also preferred by 13 out of 15 participants for visualization of moving out-of-view objects.

Index Terms: Human-centered computing—Visualization—Visualization techniques; Human-centered computing—Human computer interaction (HCI)—Empirical studies in HCI

1 Introduction

Over the past few years, head-mounted Virtual Reality (VR) devices have been steadily advancing from a technological point of view. These advances in VR technology allow it to be used for a variety of applications (e.g., training [14], simulation [20] or gaming [23]). However, in many of these applications, the limited field-of-view (FOV) of the VR device leads to objects receding from view (e.g., opponents in computer games). This is a problem because users cannot perceive these out-of-view objects, therefore they have no information about their positions or movements (e.g., to avoid accidents during ship docking a pilot has to keep track of multiple potentially occluded tugboats assisting a vessel at the same time [10]). Furthermore, due to the restricted human visual range [15], increasing the FOV of VR devices [22, 27] would not solve this problem. In previous work, different techniques have been proposed to visualize the positions of out-of-view objects on head-mounted devices (e.g., *3D Radar* [3] or *EyeSee360* [9]). Thus far, all developed techniques have been evaluated with out-of-view objects located at fixed positions. However, visualizing only static objects out of view is insufficient for many scenarios because the movement of these objects can be crucial. For example, when a user plays a VR spaceship game in which they must avoid being attacked by an opponent, knowledge of both position and movement becomes critical for a successful counterattack. Besides in computer games, moving out-of-view objects may be relevant in traffic scenarios (e.g., when determining whether it is safe to overtake [18]), or in monitoring tasks (e.g., when assessing the position and movement of tugboats during the docking process of large container vessels [10]).

*e-mail: uwe.gruenefeld@uol.de
†e-mail: ilja.koethe@uol.de
‡e-mail: daniel.lange@offis.de
§e-mail: sebastian.weiss@offis.de
¶e-mail: wilko.heuten@offis.de

In this paper, we compare two well-known visualization approaches (Overview+detail and Focus+context) in a user study to evaluate their performances for the visualization of moving out-of-view objects. For each approach, we selected a representative visualization technique from prior work: (1) *3D Radar* [3] for Overview+detail, and *EyeSee360* [9] for Focus+context. To gain novel insights into how well these techniques visualize moving out-of-view objects, we compare their performances for three different kinds of movements derived from various use cases. Therefore, we conducted a laboratory user study with 15 participants in VR. We measured movement estimation error, assessed usability with the System Usability Scale (SUS) [4] and evaluated subjective performance with an individual questionnaire.

2 Related Work

Off-screen visualization techniques Off-screen visualization techniques can be classified into two main approaches: Overview+detail and Focus+context [6, 11]. Overview+detail shows a miniature map of the surrounding area, while Focus+context overlays the screen borders with context information. Focus+context techniques were initially based on fisheye views, which convey distorted views of the off-screen space [24]. Later, only points of interest from the off-screen space were visualized on the screen and represented by abstract shapes. These techniques still follow the Focus+context approach but are called Contextual views (e.g., arrows pointing into off-screen space [5]).

A disadvantage of the Overview+detail approach is the cognitive load required to mentally integrate all views [6], while context information along the borders is more in line with the human frame of reference [13]. However, compared to Overview+detail, Focus+context techniques are not able to keep the proportions of 3D space, since they compress all information along the borders [12] (e.g., two objects that are off-screen to the right have to be projected along the right screen border, regardless of how far away from each other they are). Therefore, it is especially hard to preserve topological information for off-screen objects. While this has been improved for Focus+context approaches with 2D positions on small screen devices [12], it has not been solved for Focus+context approaches with 3D positions on head-mounted devices.

In previous research, Contextual views was identified as having the best performance for the visualization of off-screen objects on small-screen devices [5]. One of the first Contextual views techniques was Halo [1]. It uses circles drawn with their centers around the off-screen objects and cuts the border of the screen slightly. However, a problem of Halo is cluttering, which is the accumulation of many Halos in corners. To avoid cluttering, researchers developed Wedge [11], which uses isosceles triangles and requires less space. In commercial applications such as computer games, off-screen visualizations based on Overview+detail approaches are frequently used. In 3D games such as Eve: Valkyrie (2016)[1] or Elite Dangerous[2], a radar-like visualization is used. Similar approaches have been studied in research with 'worlds in miniature' [2]. However, all off-screen visualization techniques are mainly evaluated using

[1] `www.evevalkyrie.com`, last retrieved April 1, 2019
[2] `www.elitedangerous.com`, last retrieved April 1, 2019

small screen devices. They are not developed for the visualization of out-of-view objects on head-mounted devices.

Out-of-view visualization techniques In recent work, different Focus+context techniques for visualization of out-of-view objects have been investigated. Gruenefeld et al. [8] adapted Arrow, Halo, and Wedge to head-mounted Augmented Reality (AR). Their results showed that all of these techniques are applicable for head-mounted devices, but their approach was limited to 90 degrees in front of the user. Therefore, they developed HaloVR and WedgeVR, which make use of 3D shapes to guide to out-of-view objects [7]. However, the 3D shape can only encode the head movement required to bring the out-of-view object into the user's FOV, giving no information about the position of the object in 3D space. To visualize the positions of multiple out-of-view objects at the same time, a new visualization technique called *EyeSee360* was proposed [9]. *EyeSee360* uses a radar-like visualization to display out-of-view objects in the user's periphery. However, *EyeSee360* has only been investigated for static objects out of view that do not change position. Therefore, it is unclear to what degree this technique supports moving objects.

Recently, Bork et al. [3] compared six different visualization techniques for guidance towards out-of-view objects. They suggested two new Overview+detail techniques: *3D Radar* and Mirror Ball, and evaluated them against four existing Focus+context techniques: 3D Arrows [25], Aroundplot [13], *EyeSee360* [9] and sidebARs [26]. They found significantly lower completion times and better usability when using *EyeSee360*. Further, *3D Radar* was the best performing Overview+detail technique. However, they again only evaluated the guidance to static objects with fixed positions in 3D space.

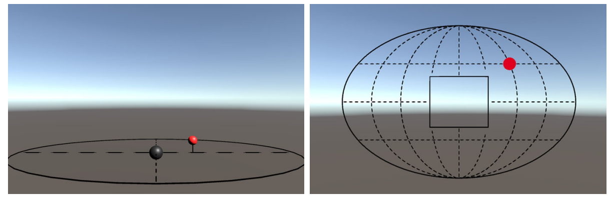
Visualization of moving objects To our knowledge, there is no previous work investigating the visualization of moving out-of-view objects in head-mounted devices. However, it has been investigated for non-visual displays (e.g., with tactile [16] or auditory cues [17]) and off-screen visualizations. For off-screen visualization, Mueller et al. [21] attached an LED strip to the borders of a tablet and used the resulting ambient light to visualize the directions to off-screen objects (first in a virtual setup [19]). In a user study, they showed that ambient light displays can be used to visualize moving off-screen objects. However, it remains unclear how one can best visualize moving out-of-view objects on HMDs.

3 Visualizing moving objects out of view

To investigate how to best visualize moving objects out of view, we first identified two relevant visualization approaches from related work: Overview+detail and, Focus+context. Both approaches are suitable for visualizing out-of-view objects; however, in previous work, the Focus+context approach worked best [3, 9]. However, all out-of-view objects tested in previous work were always located at fixed positions and not moving in 3D space. For moving objects out of view, we hypothesize that the compression of information along the borders in the Focus+context approaches not only leads to losing the proportions of 3D space, but also makes it harder to understand movement. Therefore, we expect the Overview+detail approach to work best for visualizing moving out-of-view objects. To investigate this, we selected a representative technique for each approach: (1) *Radar3D* for Overview+detail, and (2) *EyeSee360* for Focus+context. We then compared them in a user study.

3.1 3D Radar Technique

3D Radar is a visualization technique that is frequently used in various computer games (e.g., Elite Dangerous). Our implementation is based on previous work [3]. Figure 1a shows how *3D Radar* looks. *3D Radar* uses a sphere in its center to represent the user. A circle around this sphere represents zero on the y-axis. The zeros on the x- and z-axes are represented by two dotted lines. To avoid cluttering the foveal vision of the user, we moved the technique from the center of the screen to the bottom of the screen, directly in front of the user. Each out-of-view object is represented by a sphere (called a proxy). Each proxy is placed relative to the sphere in the center where it is placed relative to the user in the real world. If the object is higher or lower than the user, a line is drawn between the proxy and ground circle to show where on the circle the object would be without height.



(a) *3D Radar.* (b) *EyeSee360.*

Figure 1: Selected visualization techniques. The red sphere represents the same out-of-view object in both techniques.

3.2 EyeSee360 Technique

EyeSee360 [9] is a technique for visualizing the 3D positions of out-of-view objects in the user's periphery. Figure 1b shows how *EyeSee360* looks. *EyeSee360* concentrates information about out-of-view objects onto a grid system located in the user's periphery. This grid system compresses 3D position information onto a single 2D plane. The inner rectangle of *EyeSee360* represents the FOV of the current user, and the area outside the rectangle represents the area outside of the user's view. Each dotted line represents a 45° section of the user's view. The horizontal line expresses the altitude of the object, while the vertical curved lines represent the horizontal direction of the object. For example, the red dot (called a proxy) in Figure 1b represents an out-of-view object that is exactly 90° to the right and 45° up. The color of the proxy represents the distance to that object. It uses a color gradient from red to blue, where red is close and blue is far away.

3.3 Investigated Types of Movement

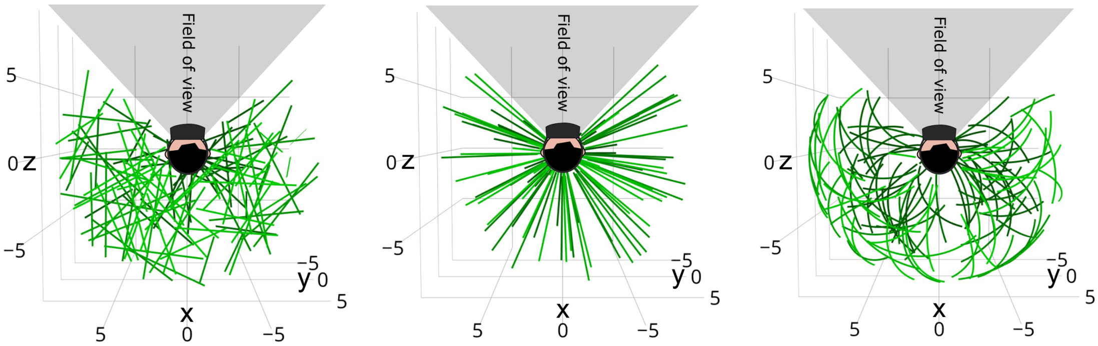
In our comparative user study we want to compare different types of movement. These movement types are derived from various scenarios. We distinguish three kinds of movement:

(M1) Linear movement from any point A to any point B.

(M2) Distance movement towards or away from the user.

(M3) Orbital movement around the user.

In general, the movement of out-of-view objects can be described by two points: a start point A and an end point B of the movement. The movement between these two points can then be a linear or non-linear movement. As a first general type of movement, we consider the linear movement between any point A and any point B *(M1)*. This kind of movement is for example relevant for determining whether an out-of-view object will cross the user's path in the future (e.g., to know whether it is safe to overtake). The next type of movement *(M2)* is a special case of *(M1)*. It describes objects moving towards or away from the user. These movements are especially critical in computer games in which a user wants to know if an opponent is moving towards or away from them. Besides linearly moving out-of-view objects, we also consider orbital movement around the user *(M3)*. This kind of movement is relevant to the user because a change in direction requires a different head movement for localizing an out-of-view object. It is also relevant in monitoring tasks (e.g., to assess the movement of tugboats around a larger container vessel).



(a) Linear movement (M1). (b) Distance movement (M2). (c) Orbital movement (M3).

Figure 2: Randomly generated movements tested in study (color gradient from light green (y=10) to dark green (y=-10)). Each line represents a tested movement. The origin of the coordinate system (0,0,0) was calibrated to the head of the participants.

4 Experiment

4.1 Study Design

To evaluate the performance of each visualization strategy for moving out-of-view objects, we conducted a within-subjects controlled laboratory study in VR. We investigate whether the dependent variables, movement estimation error and estimation error of start and end positions, are influenced by the independent variables, Visualization (*3D Radar* vs. *EyeSee360*) and Type of Movement (*M1* vs. *M2* vs. *M3*). We calculated the movement estimation error as the angle between the movement vector (from start to end position) and the user's assessment of this movement vector (assessed start to end position). The estimation error of the start position is calculated as the angle between the start position of the movement and the user's assessment of the position. The estimation error of the end position is calculated similarly. For all trials, we set the number of objects to one at a time, the movement duration to five seconds, and the movement length to eight meters. Our repeated-measures within-subjects factorial design results in six different conditions.

We expect *3D Radar* to perform better than *EyeSee360* with regard to movement estimation error for all movements (H_1) because the Overview+detail approach is able to keep the proportions of 3D space, therefore allowing the user to better assess the movement. Thus, we hypothesize that *3D Radar* will also be subjectively perceived as best (H_2).

4.2 Procedure

The study was divided into two counter-balanced blocks, with each block testing one visualization technique (*3D Radar* and *EyeSee360*). In each block, we tested three different types of movement (*M1* vs. *M2* vs. *M3*) with seven iterations[3]. The tested types of movement were randomly generated and are fully visualized in Figure 2. The randomly generated movements of the first block were stored and tested in a randomized order for the second block. This ensured that we tested the same movements for both techniques.

All movements we tested have the same length (eight meters) and are within a sphere S with a diameter of ten meters (one meter in our VR environment represents one meter in the real world). For each tested movement, we randomly selected a start position A on the surface of the sphere S. The end position B was then selected depending on the type of movement. For linear movement *(M1)*, we randomly selected a point B with a distance of eight meters to point A. For distance movement *(M2)*, point B is the result of point A divided by five. Since point A has a distance of ten meters, point B has a distance of two meters from the user, and the resulting movement has a length of eight meters. For orbital movement *(M3)*, we randomly selected a point B on the sphere S with the great-circle distance[4] of eight meters on sphere S. For each movement, we checked if all points of that movement were within the sphere S, not closer than two meters from the user, and outside of the user's FOV. If this was not the case, we generated a new random movement. Further, we randomly selected 50% of the movements and inverted their directions.

We started our experiment with a short introduction to out-of-view objects and VR. Afterwards, participants started with the two blocks. Each block started with three test trials (not included in results), along with an explanation of the visualization technique and the task to achieve. After each block, we asked participants to fill out an system usability scale (SUS) questionnaire [4], which allows to measure usability. Further, at the end of the experiment, participants were asked to fill out our individual subjective questionnaire and a demographic questionnaire. Overall, each participant took approximately 45 minutes to finish the experiment.

In each iteration, the user had to focus on a static point in front of them. Then the out-of-view object moved from the generated start position to the generated end position (always five seconds). The moving out-of-view object was invisible and was only visualized with a proxy in the current tested visualization technique. After the movement was over, the visualization technique disappeared, and the user was asked to draw the movement they saw with a tracked controller from start to end position. A dotted line indicated the input to the user. It was possible to reenter the perceived movement. Further, participants were allowed to input relative positions (e.g., the point (10,0,0) could be inputted as (1,0,0) if all other points of that movement were scaled accordingly).

4.3 Implementation

Both visualization techniques are implemented in Unity3D[5], a 3D game development platform, and the HTC Vive HMD, a head-mounted VR device. We adjusted both techniques to the maximum distance (ten meters) of the tested movements. For *3D Radar* we adjusted the black circle to the maximum distance, and in *EyeSee360* we encoded the distance of out-of-view objects with the proxy color from blue (maximum distance) to red (minimum distance).

4.4 Participants

We recruited 15 participants (6 female), aged between 18 and 35 (M=23, SD=4.36). None of them suffered from color vision impairment. All had normal or corrected to normal vision. Participants with corrected vision wore contact lenses.

[3] This number was derived from pretesting.

[4] `en.wikipedia.org/wiki/great-circle_distance`, last retrieved April 1, 2019

[5] `www.unity3d.com`, last retrieved April 1, 2019

4.5 Results

For the experiment, we consider the effects of the two factors (Visualization, Type of Movement) on movement estimation error and estimation error of start and end positions.

Estimation error of start and end positions The mean estimation errors of start positions for the visualization techniques are: *3D Radar*=39.9° and *EyeSee360*=33.6°. Normality here was not assumed because the Shapiro-Wilk test was significant ($p < 0.001$). A Wilcoxon Signed-rank with Holm-Bonferroni correction revealed a significant difference between the two visualization techniques ($W = 20353$, $Z = -2.801$, $p = 0.005$, $\phi = 0.16$). The mean estimation errors of end position for the techniques are: *3D Radar*=37.2° and *EyeSee360*=37.4°. Normality was not assumed because the Shapiro-Wilk test was significant ($p < 0.001$). A Wilcoxon Signed-rank with Holm-Bonferroni correction revealed no significant difference between the two visualization techniques ($W = 25766$, $Z = 0.545$, $p = 0.587$, $\phi = 0.03$).

Movement estimation error The mean movement estimation errors for the techniques are: *3D Radar*=30.7° and *EyeSee360*=46.1°. Normality here was not assumed because the Shapiro-Wilk test was significant ($p < 0.001$). A Wilcoxon Signed-rank with Holm-Bonferroni correction showed a significant effect of visualization technique on movement estimation error ($W = 33788$, $Z = 5.503$, $p < 0.001$, $\phi = 0.31$).

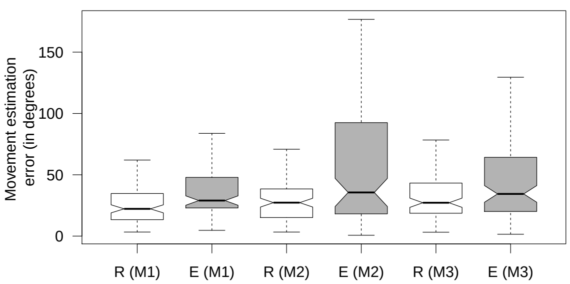


Figure 3: Boxplot of movement estimation error (R=*3D Radar*; E=*EyeSee360*; top whisker to box: first quartile; box to bottom whisker: fourth quartile; box: second and third quartile separated by median)).

To further understand the movement estimation errors for the visualization techniques we compared the mean errors for the different types of movement (see Figure 3). We ran a Friedman test that revealed a significant effect of visualization technique and movement type on movement estimation error ($\chi^2(5)$=23.275, $p<0.001$, N=15). A post-hoc test using Wilcoxon Signed-rank with Holm-Bonferroni correction showed significant differences between all conditions (see Table 1).

Table 1: Pairwise comparison for different movement types.

3D Radar	EyeSee360	Type	P-value	ϕ-value
27.1°	37.9°	*M1*	< 0.001***	0.26
29.0°	53.5°	*M2*	< 0.001***	0.28
36.1°	47.1°	*M3*	0.050*	0.14

System usability scale For SUS scores, *EyeSee360* scored 47.5. This score is clearly below acceptable usability and far from the SUS score reported in the *EyeSee360* paper (which was 68 for head-mounted AR) [9]. This may be due to the fact, that in these scenarios, the out-of-view objects were moving, and *EyeSee360* did not seem to be able to visualize these moving objects. However, *3D Radar* scored 71.2, which is over the threshold for acceptable usability. Our results show that *3D Radar* is usable for moving out-of-view objects in head-mounted VR, while *EyeSee360* is not.

Subjective questionnaire At the end of the study, we asked participants to answer four questions with 5-point Likert items. Participants stated that they were able to exactly determine the movements of out-of-view objects with *3D Radar* (Md=4, IQR=0), but not with *EyeSee360* (Md=2, IQR=1.5). Furthermore, they stated that they were able to determine the movement quickly with *3D Radar* (Md=4, IQR=0), but not with *EyeSee360* (Md=2, IQR=1). Overall, 13 participants preferred *3D Radar*, while only two preferred *EyeSee360*.

5 Discussion and Limitations

Overview+detail vs. Focus+context Focus+context approaches do not have the ability to keep the proportions of 3D space, since they compress all information along the borders of the screen. In our results, we showed that *EyeSee360* performed worse than *3D Radar* with regard to movement estimation error. We think that this is mainly due to the Focus+context approach that is used for *EyeSee360*. We predicted this outcome in hypothesis H_1, and therefore we can accept our hypothesis H_1. However, the Overview+detail approach has disadvantages when it comes to quickly locating out-of-view objects (e.g., a proxy in *EyeSee360* is already encoding in which direction the user should move their head to locate the out-of-view object).

Linear movement Almost all participants (n=12) perceived the movement of linearly moving objects as orbital in *EyeSee360*, while the participants perceived it as linear with *3D Radar*. Although, some participants had difficulties drawing straight lines, the lines of trials with *3D Radar* were less curved. We think this is due to the mapping of 3D direction information onto a 2D plane in *EyeSee360*.

Distance movement The encoding of distance with color was problematic in *EyeSee360* (cp. Figure 3 (M2)). Here, users often mixed up the direction of the movement (i.e., perceived objects as if they were approaching when actually they were moving away, or vice versa) resulting in a high movement estimation error up to 180°. The problem is that color has no spatial attributes, and therefore encoding 3D distance with color is problematic. We think that changing the proxy size depending on the distance to the out-of-view object may improve the perception of distance in *EyeSee360*.

Orbital movement In our study, we investigated different types of movement. Interestingly, *3D Radar* performed better than *EyeSee360* for all three types of movement. However, both visualization techniques had problems with visualization of orbital movement. Here, the orbital movement could be better reproduced by all participants with *3D Radar*. With *EyeSee360*, it seemed to depend strongly on how intuitively the color coding and movement was perceived. We think that orbital movement was problematic for participants in general because it was harder to draw that movement with the tracked controller.

Ecological validity In our study, we evaluated both techniques in a simple 'ground and sky' scene. We think this is a suitable approach to gather first insights into how well moving objects out of view are perceived in the tested techniques. However, future work should evaluate the two techniques in more realistic scenes.

Different parameters In our work, we focused on comparing two different approaches for visualizing moving objects out of view. Therefore, we reduced the complexity of our user study by reducing the number of independent variables in our design. However, future work should investigate how users can perceive the movements of multiple out-of-view objects. Further, movements with different durations and lengths can be tested.

6 Conclusion

In this paper, we compared two visualization approaches, Overview+detail and Focus+context, for moving out-of-view objects in a user study. We selected one representative visualization technique for each approach: (1) *3D Radar* for Overview+detail and, (2) *EyeSee360* for Focus+context. Our results show that *3D Radar* objectively and subjectively works best for moving out-of-view objects. Furthermore, *3D Radar* can encode more information, such as the orientation or size of an object out of view. In future work, we want to test both techniques in more realistic scenarios (e.g., VR games).

References

[1] P. Baudisch and R. Rosenholtz. Halo: A technique for visualizing off-screen objects. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '03, pp. 481–488. ACM, New York, NY, USA, 2003. doi: 10.1145/642611.642695

[2] B. Bell, T. Höllerer, and S. Feiner. An annotated situation-awareness aid for augmented reality. In *Proceedings of the 15th Annual ACM Symposium on User Interface Software and Technology*, UIST '02, pp. 213–216. ACM, New York, NY, USA, 2002. doi: 10.1145/571985.572017

[3] F. Bork, C. Schnelzer, U. Eck, and N. Navab. Towards efficient visual guidance in limited field-of-view head-mounted displays. *IEEE Transactions on Visualization and Computer Graphics*, 24(11):2983–2992, Nov 2018. doi: 10.1109/TVCG.2018.2868584

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

[5] S. Burigat, L. Chittaro, and S. Gabrielli. Visualizing locations of off-screen objects on mobile devices: A comparative evaluation of three approaches. In *Proceedings of the 8th Conference on Human-computer Interaction with Mobile Devices and Services*, MobileHCI '06, pp. 239–246. ACM, New York, NY, USA, 2006. doi: 10.1145/1152215.1152266

[6] A. Cockburn, A. Karlson, and B. B. Bederson. A review of overview+detail, zooming, and focus+context interfaces. *ACM Comput. Surv.*, 41(1):2:1–2:31, Jan. 2009. doi: 10.1145/1456650.1456652

[7] U. Gruenefeld, A. E. Ali, S. Boll, and W. Heuten. Beyond halo and wedge: Visualizing out-of-view objects on head-mounted virtual and augmented reality devices. In *Proceedings of the 20th International Conference on Human-Computer Interaction with Mobile Devices and Services*, MobileHCI '18, pp. 40:1–40:11. ACM, New York, NY, USA, 2018. doi: 10.1145/3229434.3229438

[8] U. Gruenefeld, A. E. Ali, W. Heuten, and S. Boll. Visualizing out-of-view objects in head-mounted augmented reality. In *Proceedings of the 19th International Conference on Human-Computer Interaction with Mobile Devices and Services*, MobileHCI '17, pp. 81:1–81:7. ACM, New York, NY, USA, 2017. doi: 10.1145/3098279.3122124

[9] U. Gruenefeld, D. Ennenga, A. E. Ali, W. Heuten, and S. Boll. Eyesee360: Designing a visualization technique for out-of-view objects in head-mounted augmented reality. In *Proceedings of the 5th Symposium on Spatial User Interaction*, SUI '17, pp. 109–118. ACM, New York, NY, USA, 2017. doi: 10.1145/3131277.3132175

[10] U. Gruenefeld, T. C. Stratmann, Y. Brueck, A. Hahn, S. Boll, and W. Heuten. Investigations on container ship berthing from the pilot's perspective: Accident analysis, ethnographic study, and online survey. *TransNav, the International Journal on Marine Navigation and Safety of Sea Transportation*, 12(3):492–498, Sept. 2018. doi: 10.12716/1001.12.03.07

[11] S. Gustafson, P. Baudisch, C. Gutwin, and P. Irani. Wedge: Clutter-free visualization of off-screen locations. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '08, pp. 787–796. ACM, New York, NY, USA, 2008. doi: 10.1145/1357054.1357179

[12] D. Jäckle, J. Fuchs, and H. Reiterer. Topology-Preserving Off-screen Visualization: Effects of Projection Strategy and Intrusion Adaption. Technical report, University of Konstanz, 2017.

[13] H. Jo, S. Hwang, H. Park, and J. hee Ryu. Aroundplot: Focus+context interface for off-screen objects in 3d environments. *Computers & Graphics*, 35(4):841 – 853, 2011. Semantic 3D Media and Content. doi: 10.1016/j.cag.2011.04.005

[14] N. W. John, S. R. Pop, T. W. Day, P. D. Ritsos, and C. J. Headleand. The implementation and validation of a virtual environment for training powered wheelchair manoeuvres. *IEEE Transactions on Visualization and Computer Graphics*, 24(5):1867–1878, May 2018. doi: 10.1109/TVCG.2017.2700273

[15] J. Kalat. *Biological psychology*. Nelson Education, 2015.

[16] O. B. Kaul and M. Rohs. Haptichead: A spherical vibrotactile grid around the head for 3d guidance in virtual and augmented reality. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*, CHI '17, pp. 3729–3740. ACM, New York, NY, USA, 2017. doi: 10.1145/3025453.3025684

[17] H. Kerdegari, Y. Kim, and T. J. Prescott. Head-mounted sensory augmentation device: Comparing haptic and audio modality. In N. F. Lepora, A. Mura, M. Mangan, P. F. Verschure, M. Desmulliez, and T. J. Prescott, eds., *Biomimetic and Biohybrid Systems*, pp. 107–118. Springer International Publishing, Cham, 2016.

[18] A. Löcken, W. Heuten, and S. Boll. Supporting lane change decisions with ambient light. In *Proceedings of the 7th International Conference on Automotive User Interfaces and Interactive Vehicular Applications*, AutomotiveUI '15, pp. 204–211. ACM, New York, NY, USA, 2015. doi: 10.1145/2799250.2799259

[19] A. Löcken, H. Müller, W. Heuten, and S. C. Boll. Exploring the design space of ambient light displays. In *CHI '14 Extended Abstracts on Human Factors in Computing Systems*, CHI EA '14, pp. 387–390. ACM, New York, NY, USA, 2014. doi: 10.1145/2559206.2574793

[20] C. D. Loreto, J. Chardonnet, J. Ryard, and A. Rousseau. Woah: A virtual reality work-at-height simulator. In *2018 IEEE Conference on Virtual Reality and 3D User Interfaces (VR)*, pp. 281–288, March 2018. doi: 10.1109/VR.2018.8448292

[21] H. Müller, A. Löcken, W. Heuten, and S. Boll. Sparkle: An ambient light display for dynamic off-screen points of interest. In *Proceedings of the 8th Nordic Conference on Human-Computer Interaction: Fun, Fast, Foundational*, NordiCHI '14, pp. 51–60. ACM, New York, NY, USA, 2014. doi: 10.1145/2639189.2639205

[22] J. Orlosky, Q. Wu, K. Kiyokawa, H. Takemura, and C. Nitschke. Fisheye vision: Peripheral spatial compression for improved field of view in head mounted displays. In *Proc. SUI '14*, pp. 54–61. ACM, New York, NY, USA, 2014. doi: 10.1145/2659766.2659771

[23] M. Rietzler, F. Geiselhart, and E. Rukzio. The matrix has you: Realizing slow motion in full-body virtual reality. In *Proceedings of the 23rd ACM Symposium on Virtual Reality Software and Technology*, VRST '17, pp. 2:1–2:10. ACM, New York, NY, USA, 2017. doi: 10.1145/3139131.3139145

[24] M. Sarkar and M. H. Brown. Graphical fisheye views. *Communications of the ACM*, 37(12):73–83, 1994.

[25] T. Schinke, N. Henze, and S. Boll. Visualization of off-screen objects in mobile augmented reality. In *Proceedings of the 12th International Conference on Human Computer Interaction with Mobile Devices and Services*, MobileHCI '10, pp. 313–316. ACM, New York, NY, USA, 2010. doi: 10.1145/1851600.1851655

[26] T. Siu and V. Herskovic. Sidebars: Improving awareness of off-screen elements in mobile augmented reality. In *Proceedings of the 2013 Chilean Conference on Human - Computer Interaction*, ChileCHI '13, pp. 36–41. ACM, New York, NY, USA, 2013. doi: 10.1145/2535597.2535608

[27] R. Xiao and H. Benko. Augmenting the field-of-view of head-mounted displays with sparse peripheral displays. In *Proc. CHI '16*, pp. 1221–1232. ACM, New York, NY, USA, 2016. doi: 10.1145/2858036.2858212