Visualizing Out-of-view Objects in Head-mounted Augmented Reality

Uwe Gruenefeld
University of Oldenburg,
Germany
uwe.gruenefeld@uol.de

Abdallah El Ali
University of Oldenburg,
Germany
abdallah.el.ali@uol.de

Wilko Heuten
OFFIS - Institute for IT
Oldenburg, Germany
wilko.heuten@offis.de

Susanne Boll
University of Oldenburg,
Germany
susanne.boll@uol.de

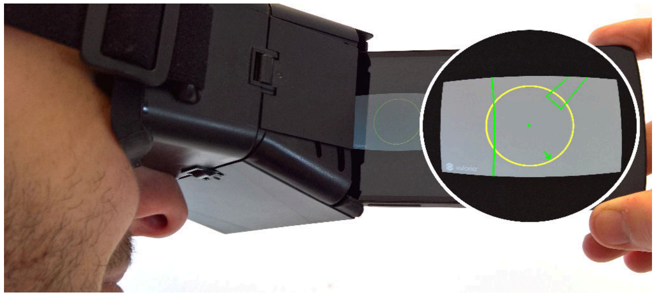


Figure 1: Adapted 2D off-screen visualization techniques in head-mounted Augmented Reality. *Best seen in color.*

MobileHCI '17, September 04-07, 2017, Vienna, Austria
ACM 978-1-4503-5075-4/17/09.
https://doi.org/10.1145/3098279.3122124

Abstract

Various off-screen visualization techniques that point to off-screen objects have been developed for small screen devices. A similar problem arises with head-mounted Augmented Reality (AR) with respect to the human field-of-view, where objects may be out of view. Being able to detect so-called out-of-view objects is useful for certain scenarios (e.g., situation monitoring during ship docking). To augment existing AR with this capability, we adapted and tested well-known 2D off-screen object visualization techniques (Arrow, Halo, Wedge) for head-mounted AR. We found that Halo resulted in the lowest error for direction estimation while Wedge was subjectively perceived as best. We discuss future directions of how to best visualize out-of-view objects in head-mounted AR.

Author Keywords

Augmented Reality; Head-mounted; Out-of-View Objects; Off-screen; Visualization Techniques; Peripheral Awareness

ACM Classification Keywords

H.5.m. [Information Interfaces and Presentation (e.g., HCI)]: Miscellaneous

Introduction

Recent advances in augmented reality (AR) technology enable a variety of new applications (e.g., for games [9] or for localization [6]). What all AR applications have in common is the idea of overlaying digital information on the real world. Combined with a head-mounted device, AR comes with advantages like hands-free operation. However, the problem of perceiving information about objects that are out of view still exists. Since the human field-of-view is naturally limited, spatially distributed objects outside of this range will not be perceived. AR could be used to augment human vision and extend this range. This capacity is particularly useful for scenarios such as ship docking, where objects often change their position or new objects with unknown positions appear or first-person multi-player games, where the position of other players is relevant.

In our approach we adapted 2D visualization techniques from the well-studied field of off-screen visualization, and applied them to AR for perceiving out-of-view objects. We employed a projection plane orthogonal to the user's line-of-sight in the user's view frustum and utilized the well-known 2D off-screen visualization techniques of Arrow [3, 7], Halo [1] and Wedge [5] to aid in visualizing out-of-view objects. To make the 2D visualizations applicable in 3D space, we used a two-step projection to translate 3D coordinates to 2D coordinates on the orthogonal projection plane (see Section Projection from 3D to 2D). We conducted a comparative user study to evaluate the performance of these techniques in AR.

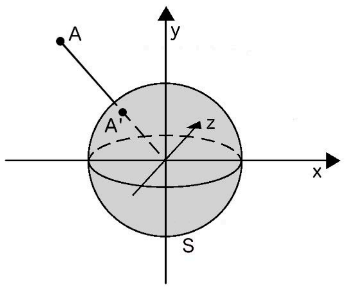
Our research contributions include:

1. An adaption of three 2D off-screen visualization techniques for head-mounted AR that serves as a baseline for future work.
2. An evaluation of the adapted techniques for visualization of out-of-view objects in AR.

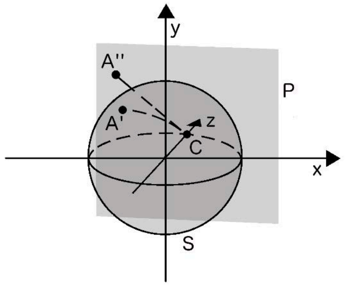
Related work

How to best perceive information about out-of-view objects is still an open issue in head-mounted AR. Since our work is based on existing off-screen visualization techniques, we cover a small subset of research in that space. The three main approaches used to overcome small displays in 2D include: Overview+detail, Contextual views, and Focus+context [5, 4]. Contextual views and Focus+Context both overlay the screen borders with context information while Overview+Detail shows a miniature map of the surrounding area. A disadvantage of the miniature map is the cognitive load required to mentally integrate all views, while context information along the borders is more in line with the human frame of reference. Contextual views and Focus+Context differ in the kind of transition between focus and context. In the Focus+Context approach the transition is soft (e.g., fisheye-views that convey a distorted view [10]) and for Contextual views the transition is hard (e.g., arrows pointing into off-screen space [3]). Since Contextual views are distortion free due to the hard transitions, we utilized them in our work with head-mounted AR.

One of the first Contextual views was presented by Zellweger et al. [12], who provided contextual information along the borders but users found it difficult to guess the actual position of the off-screen objects. Therefore, Halo was suggested as an improvement [1]. It uses circles drawn with their center around the off-screen object and cut the border of the screen slightly. However, a problem of Halo is cluttering, which is the accumulation of many Halos in corners. In Arrow the smaller shape of arrows is used to point towards off-screen objects. Several studies compared Halo with Arrow approaches [3, 7], where Arrows with fixed sizes performed worse than Halo while scaled arrows performed slightly better. Also the amount of visible objects have a high impact on the performance. To avoid clutter-



(a) First Step: Projection of 3D coordinate A on sphere S.



(b) Second Step: Project sphere coordinate A' on plane P.

Figure 2: Example of proportional projection.

ing, researchers developed Wedge [5], which uses less space with isosceles triangles. However the smaller form can lead to an inaccurate understanding of the off-screen object's position. In our approach, we choose Arrow, Halo, and Wedge since they are well studied and can be easily applied onto the projected out-of-view objects.

General Approach

We restricted ourselves to 90 degrees of 3D space in front of the user. Therefore, we avoided off-screen objects behind the user, as this makes the adaptation more complex. For example, an object that is exactly 180 degrees behind the user can be represented ambiguously on both the left and right sides of the viewing plane. Furthermore, as a first step we wanted to evaluate if the projection plane in the users view frustum was a feasible option for encoding direction information. In a first comparative study, we implemented the three selected visualization techniques (Arrow, Halo, Wedge) in video see-through AR. With video see-through AR, a camera image is looped onto a screen directly in front of the user's eyes. This was implemented with Vuforia[1] in Unity[2] using Google Cardboard[3]. As an additional minor contribution, we were able to evaluate the feasibility of Google Cardboard as a cheap and fast development platform for off-screen AR visualizations.

Projection from 3D to 2D

A key aspect of our approach is the projection of out-of-view objects in 3D space onto a 2D plane (projection plane) and further applying 2D techniques that indicate direction towards an off-screen object. This direction information should consider the human frame-of-reference which means that objects behind the user should be indicated by turning the head to the side and not upwards. Additionally, the mapping needs to be proportional for all directions. The proportional projection method we apply here draws on the mercator projection[4].

An example mapping is shown in Figure 2. The user's head serves as the origin for the coordinate system. First, we map point A, which represents an out-of-view object, onto sphere S. The mapped point A' is the intersection point of the line segment between the point A and the origin point of the sphere S (see Figure 2a). Second, we map point A' onto the plane P (see Figure 2b), which is placed orthogonal to the x and y axis at contact point C. Then we calculate the shortest line on the surface of the sphere S with monotonic y-values between C and A'. Finally, we map the line on the plane while keeping the length information and the same angle from C. The point A'' at the end of this line is the 3D to 2D mapped point.

Implementation

Our implementations of the 2D off-screen object visualization techniques (Arrow, Halo, Wedge) adapted to head-mounted AR are shown in Figure 3. A brief overview of the visualization techniques is provided below:

> **Arrow** points towards the off-screen objects location. The arrow itself scales depending on how far it is from the object; a bigger arrow is used to indicate a closer distance and a smaller arrow is used for a location further away [3, 7].
>
> **Halo** surrounds off-screen objects with rings. The center of the ring is exactly at the off-screen object's position. The rings are just large enough to be on-screen [1].

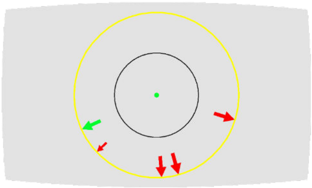
---

[1] https://www.vuforia.com

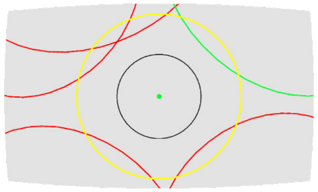
[2] https://unity3d.com

[3] https://vr.google.com/cardboard/

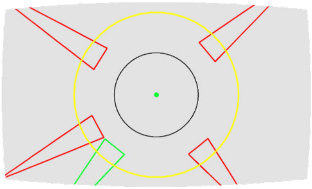
[4] https://en.wikipedia.org/wiki/Mercator_projection



(a) Arrow [3, 7]



(b) Halo [1]



(c) Wedge [5]

Figure 3: Adapted 2D off-screen visualization techniques (Arrow, Halo, Wedge). *Best seen in color*.

Wedge uses isosceles triangles to represent the position of each off-screen object. The tip of the triangle is at the object's position. They make room for each other to avoid cluttering [5].

Instead of using the border of the screen to show contextual information, we used a circle (seen as yellow circle in Figure 3). This is more in line with the peripheral capabilities of the human vision system [11, 8]. Moreover, with a circle border both orthographic and radian projections are equivalent.

A green dot in the center of the screen served as a cursor for communicating the user's guessed position for the out-of-view object. The dot was controlled by head movement and could be confirmed with a bluetooth remote control. Additionally, we used a black circle fixed in the environment around the green cursor to control the visibility of the visualization. As long as the pointer stays within the black circle the visualization is visible. If the pointer leaves the black circle the visualization becomes invisible. We had to limit the visibility to avoid simple approximation approaches by the participant while they estimate the position of the out-of-view objects. The visualization techniques are shown in red by default and green when highlighted.

Study design

To evaluate the performance of the adapted techniques (Arrow, Halo, Wedge), we conducted a within-subjects controlled laboratory study. Our study had two independent variables, technique with three levels (Arrow vs. Halo vs. Wedge) and number of objects with three levels (one vs. five vs. eight). We varied the number of shown out-of-view objects since Halo suffers from on-screen cluttering (cf., [3, 7]), and during pilot tests found that eight objects was the threshold for cluttering. We used quantitative methods to evaluate user performance, where our dependent variable was direction error. Additionally, we gave participants the SUS questionnaire [2] in order to gain insight into the perceived usability.

The direction error here is the angular error, which is the angle between the user's assessment of the out-of-view objects position and the correct position in 3D space. We did not measure task completion time since we were primarily concerned with direction error in this preliminary study. We were also unsure of how Google Cardboard would perform with these visualization techniques because prelimitary trials revealed simulator sickness effects during task that measured completition time.

For this study, we asked: **Which 2D adapted visualization techniques (Arrow, Halo, Wedge) performs best with respect to direction accuracy and perceived usability of varying out-of-view objects?**

Since Wedge outperformed both Halo and Arrow in prior research [5] for a smaller amount of objects, we hypothesized that:

H_1**: Wedge would result in better user performance (i.e., lower angular direction error) than both Halo and Arrow.**

Procedure

Participants were first given a demo of the Google Cardboard device where they could test out the different visualization techniques. The within subjects study was divided into three blocks, where each block tested one visualization technique. We counter-balanced the blocks across all participants. Each visualization technique was tested with one, five and eight out-of-view objects. Each number of out-of-view objects was tested five times (which was deemed suf-

Comparison	P-value	ϕ-value
Halo, Arrow	< 0.001	0.46
Wedge, Arrow	< 0.001	0.35
Wedge, Halo	< 0.01	0.20

Table 1: Pairwise comparision of visualization-techniques.

ficient from pilot testing). The amount of out-of-view objects in every run was randomized. Given the foregoing, we had 3 (blocks) x 3 (number of objects) x 5 (iterations) resulting in 45 runs per participant.

In each run, the out-of-view object was highlighted green (see Figure 3) and the participant had to guess the position without seeing the out-of-view object. A green cursor controlled by a wireless remote allowed participants to select the out-of-view object's position. To avoid getting the exact position of an out-of-view object through head movement, the visualization technique was only visible in a small area directly in front of the participant. Moving the green cursor out of a black circle disabled the visualization technique and the participant had to guess the out-of-view object's position by the affordances the technique offered. After each block, participants had to fill out an SUS questionnaire [2] about the technique in that block. At the end of the study, participants filled out a general information form (age, gender, rated their experience with head-mounted devices on a 5-point Likert-scale, where 1 is strongly disagree and 5 is strongly agree, and they were asked if they suffered simulator sickness). Overall each participant took approximately 45 minutes to finish the experiment. During our study we did not consider eye movement, therefore we told the participants to keep looking at the green dot in the center of the screen.

Participants

We recruited 22 participants[5] (7 females), aged between 20 and 38 years (M=25.5, SD=3.7). None of them suffered from color vision impairments. All had normal or corrected to normal vision. Ten of the participants had already experience with head-mounted displays.

Results

Direction Error We consider the effects of each of the two factors Technique and Number of Objects on direction error. The mean errors are Arrow=6.53°, Halo=3.87° and Wedge=4.47°. Our data does not follow a normal distribution (Shapiro-Wilk-Test ($p < 0.001$)), and thereafter we compared more than two matched groups using the non-parametric Friedman test. Friedman test revealed a significant effect of visualization technique on direction error ($\chi^2(2)=159.88$, $p < 0.001$, N=22). Post-hoc test using Wilcoxon Signed-rank with Bonferroni correction showed significant differences between the three groups (see Table 1).

Next, we ran a Friedman test to evaluate whether there was a significant effect of Number of Objects on direction error. First, we looked at interactions between Number of Objects and Technique. Here, we did not find a significant effect ($\chi^2(2)=3.08$, $p = 0.21$, N=22). Then, we looked into the Number of Objects for each visualization technique separately. Here again, there were no significant effects for Arrow ($\chi^2(2)=2.17$, $p = 0.34$, N=22), Halo ($\chi^2(2)=0.67$, $p = 0.71$, N=22) and Wedge ($\chi^2(2)=4.74$, $p = 0.09$, N=22). The mean direction error for all combinations of visualization techniques and number of objects are shown as a boxplot in Figure 4.

Perceived Usability With respect to perceived usability, Wedge (70) was deemed on average to be most usable, where it just passes the accepted SUS literature threshold score of 70. Halo (66) was a runner up, and Arrow (61) was perceived to have poor usability.

---

[5] For mean effect sizes of ($f = 0.2$), at least 390 data points are necessary, wherein with the planned study design this makes for testing at least 9 participants. We calculated this value with G*Power under two-way ANOVA ($\alpha = 0.05$ and $1 - \beta = 0.9$). We based it on three techniques with three different number of objects for each, which makes nine in total. The numerator df is (3 - 1) * (3 - 1) = 4.

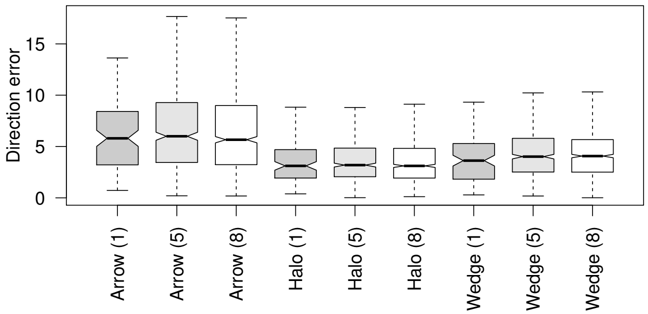


Figure 4: Boxplot of mean direction error.

Discussion

Our initial results indicate differences among the established techniques which we adapted for head-mounted AR. In line with our hypothesis H_1, we saw that Halo and Wedge performed better with respect to direction error than Arrow. However, we expected Wedge to outperform Halo, due to Halo's known cluttering effects [5]. Therefore, our hypothesis H_1 did not hold. However, with respect to perceived usability, Wedge performed slightly better than Halo.

Moreover, we observed that direction error increased when the angle between the users line of sight and out-of-view objects increased. In other words, the error increases when users had to turn their heads more. Furthermore, the use of an orthogonal plane in front of the user is a problem for higher degree values because the transferred techniques only indicate head-movements towards the off-screen objects. To summarize, our approach is feasible for visualizing the position of objects in 3D space 90° in front of the user, but for 180° or even 360° we have to further adapt Arrow, Halo, and Wedge or come up with a new solution.

Additionally, we observed how video-see-through AR in platforms such as Google Cardboard perform. We discovered that one advantage of our approach was that participants did not suffer from simulator illness. Since all participants stated that in the general information form in the end of the study. Participants however, did face a slightly delaying picture of their environment, which they compensated by moving their head slower. Therefore, video-see-through might not be well-suited for tasks that involve fast movement, such as measuring the time to search for an out-of-view object. However, from a development perspective, the Google Cardboard platform was simple to use for video-see-through AR. With Unity and Vuforia there exists a beginner friendly development environment that is cost effective and easy to setup.

Conclusion and Future work

In this paper we compared three off-screen visualization techniques for head-mounted augmented reality with respect to their performance for visualization of out-of-view objects. We found out that Halo *objectively* performed best while Wedge *subjectively* performed best. In the future, our goal is to expand our work in the 3D space to 180° or 360°. In this respect, we will investigate if a curved projection plane performs better in minimizing direction error. Furthermore, we would like to investigate the encoding of distance information in 3D space and adapting the visualization area to optical-see-through devices (e.g., Hololens) which suffer a very narrow field of view.

Acknowledgments

We thank the Ministry of Science and Culture of Lower Saxony for supporting us with the graduate school *Safe Automation of Maritime Systems (SAMS)*.

REFERENCES

1. Patrick Baudisch and Ruth Rosenholtz. 2003. Halo: A Technique for Visualizing Off-screen Objects. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '03)*. ACM, New York, NY, USA, 481–488. `DOI:` `http://dx.doi.org/10.1145/642611.642695`

2. John Brooke and others. 1996. SUS-A quick and dirty usability scale. *Usability evaluation in industry* 189, 194 (1996), 4–7.

3. Stefano Burigat, Luca Chittaro, and Silvia Gabrielli. 2006. 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)*. ACM, New York, NY, USA, 239–246. `DOI:` `http://dx.doi.org/10.1145/1152215.1152266`

4. Andy Cockburn, Amy Karlson, and Benjamin B. Bederson. 2009. A Review of Overview+Detail, Zooming, and Focus+Context Interfaces. *ACM Comput. Surv.* 41, 1, Article 2 (Jan. 2009), 31 pages. `DOI:` `http://dx.doi.org/10.1145/1456650.1456652`

5. Sean Gustafson, Patrick Baudisch, Carl Gutwin, and Pourang Irani. 2008. Wedge: Clutter-free Visualization of Off-screen Locations. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '08)*. ACM, New York, NY, USA, 787–796. `DOI:` `http://dx.doi.org/10.1145/1357054.1357179`

6. Florian Heller, Jayan Jevanesan, Pascal Dietrich, and Jan Borchers. 2016. Where Are We?: Evaluating the Current Rendering Fidelity of Mobile Audio Augmented Reality Systems. In *Proceedings of the 18th International Conference on Human-Computer Interaction with Mobile Devices and Services (MobileHCI '16)*. ACM, New York, NY, USA, 278–282. `DOI:http://dx.doi.org/10.1145/2935334.2935365`

7. Niels Henze, Benjamin Poppinga, and Susanne Boll. 2010. Experiments in the Wild: Public Evaluation of Off-screen Visualizations in the Android Market. In *Proceedings of the 6th Nordic Conference on Human-Computer Interaction: Extending Boundaries (NordiCHI '10)*. ACM, New York, NY, USA, 675–678. `DOI:http://dx.doi.org/10.1145/1868914.1869002`

8. Adam M. Larson and Lester C. Loschky. 2009. The contributions of central versus peripheral vision to scene gist recognition. *Journal of Vision* 9, 10 (2009), 6. `DOI:http://dx.doi.org/10.1167/9.10.6`

9. Leif Oppermann, Lisa Blum, and Marius Shekow. 2016. Playing on AREEF: Evaluation of an Underwater Augmented Reality Game for Kids. In *Proceedings of the 18th International Conference on Human-Computer Interaction with Mobile Devices and Services (MobileHCI '16)*. ACM, New York, NY, USA, 330–340. `DOI:http://dx.doi.org/10.1145/2935334.2935368`

10. Manojit Sarkar and Marc H Brown. 1994. Graphical fisheye views. *Commun. ACM* 37, 12 (1994), 73–83.

11. Hans Strasburger, Ingo Rentschler, and Martin Jüttner. 2011. Peripheral vision and pattern recognition: A review. *Journal of Vision* 11, 5 (2011), 13. `DOI:` `http://dx.doi.org/10.1167/11.5.13`

12. Polle T. Zellweger, Jock D. Mackinlay, Lance Good, Mark Stefik, and Patrick Baudisch. 2003. City Lights: Contextual Views in Minimal Space. In *CHI '03 Extended Abstracts on Human Factors in Computing Systems (CHI EA '03)*. ACM, New York, NY, USA, 838–839. `DOI:` `http://dx.doi.org/10.1145/765891.766022`