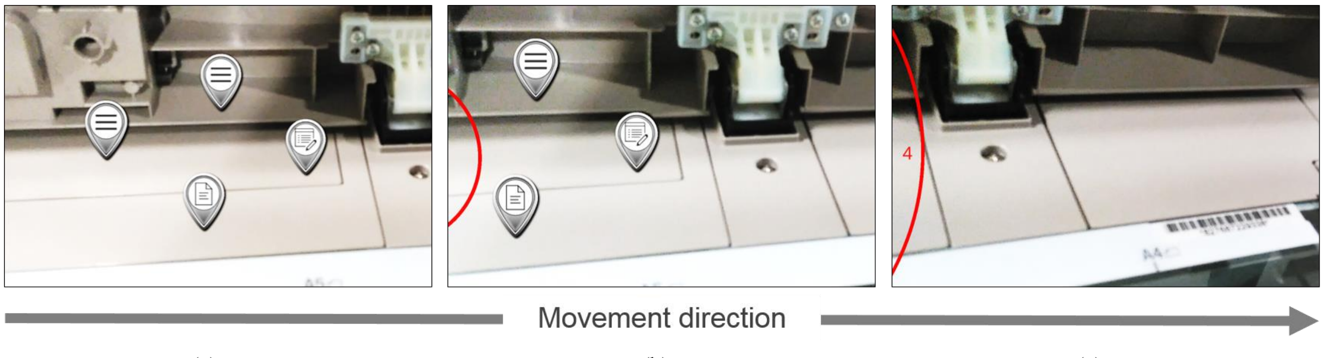
# [POSTER] Halo3D: a Technique for Visualizing Off-Screen Points of Interest in Mobile Augmented Reality



(a) (b) (c)

Figure 1: Halo3D shows the location of off-screen points of interest in mobile augmented reality. (a) 4 POIs on screen (b) 1 off-screen POI represented on screen as an arc (c) 4 aggregated off-screen POIs represented on screen as a single arc along with the number of aggregated POIs.

## Abstract

When working with mobile Augmented Reality (AR) applications, users need to be aware of relevant points of interest (POIs) that are located off-screen. These POIs belong to the context since they are not observable in the 3D first-person AR view on screen. The context in mobile AR can include a large number of POIs including locally dense clusters as in mobile AR applications for production plant machine maintenance. Existing solutions display 3D arrows or an area on the edges of the screen to represent the POIs of the context. These techniques display the direction but not the distance of each POI. We present Halo3D, a visualization technique that conveys the 3D direction and distance of off-screen POIs while avoiding overlap and clutter in a high-POI-density AR environment.

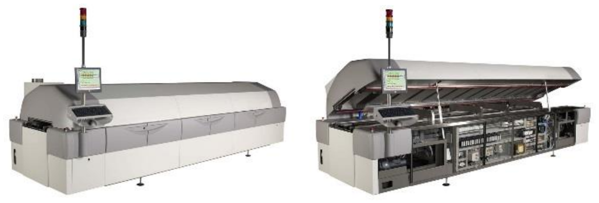
**Keywords**: Augmented reality, mobility, visualization, off-screen point of interest, Halo.

**Index Terms**: • Human-centered computing → Mixed / augmented reality • Human-centered computing → Interaction techniques

## 1 Introduction

Finding objects in the environment is a frequent need when using mobile Augmented Reality (AR) applications. The user's field of view of the environment is defined by the mobile device's camera (Fig. 1). The need is then to acquire information on digital objects anchored in the physical world (i.e. points of interest POIs) that are located outside of the current field of view (Fig. 1). We motivate this need by considering three use cases in the domain of industrial maintenance.

Use case 1: Machines that perform wave soldering for electronic boards are very large (Fig. 2). The effective maintenance of such equipment requires a real-time monitoring of many technical variables related to energy consumption, to the speed of motors (e.g. rotation per minutes) or to alarms indicating a lack of raw materials. These points of interest (POIs) can be located along the entire length of the machine.



(a) (b)

Figure 2: Welding furnace (a) closed (b) open.

Use case 2: When managing the failure of an electrical cabinet (Fig. 3) the operator shuts off power at a precise location of the electrical installation (e.g. "lookout/tagout" procedure) to repair the equipment. Once the work is completed, power is restored. To do so the operator has to manually turn on the circuit breakers and this must be done in a specific order. Each of the circuit breakers is a point of interest (POI) to locate.

The two use cases above illustrate two types of tasks that the operators can perform during maintenance:

- Use case 1: obtain the overall machine's state without having a specific target (e.g. obtain the values of variables linked to several POIs), called exploration task. [6].

- Use case 2: move towards a well-defined location of the equipment (e.g. head towards a specific circuit breaker), called navigation task [6].

* email address

Use case 3: When performing a preventive maintenance, operators execute “maintenance campaigns”. A campaign consists of a single task repeated X times on a piece of equipment. For example, grease all the motors of a machine is a campaign. Each motor is a point of interest (POI). For campaigns, it is useful for the operator to know how far away the POIs are located in order to orientate himself towards the closest one from his position, and therefore to optimize his campaign task completion time.

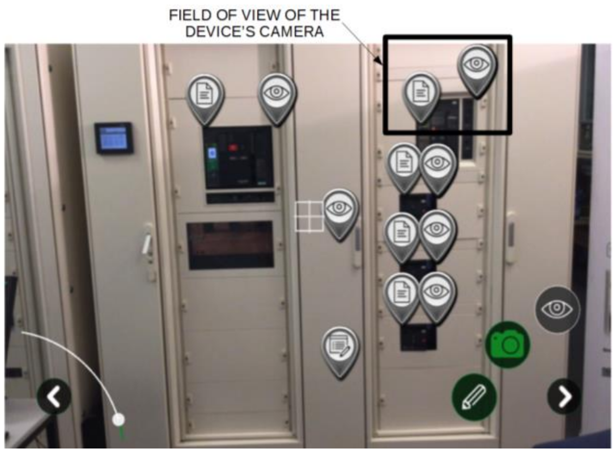


Figure 3: Screenshot of a Schneider Electric mobile AR application in front of an electrical cabinet. Global view of all the POIs (12 POIs) while the machine operator is far from the cabinet. The operator gets closer and points his tablet towards the upper part of the cabinet. The black rectangle shows the resulting field of view on the mobile device’s screen. In this context, the challenge is to indicate the location and the distance of the 10 off-screen POIs without overloading the graphical user’s interface.

In this context of industrial maintenance, mobile AR applications provide additional information on the state of the equipment (Fig. 3). However, it is not possible to simultaneously visualize all the POIs on screen. The narrow field of view of the camera limits the user’s spatial knowledge and implies frequent rotations and movements around the spot in order to explore the AR environment. This issue is even worst when using small-screen devices. Halo3D addresses this issue by visualizing the 3D location and distance of off-screen points of interests (POIs) on top of the 3D AR view. Halo3D avoids overloading the screen on which the main AR view is displayed. For the case of a high density of POIs (Fig. 3), Halo3D avoids overlap so that the display remains intelligible. The contributions of this paper are twofold:

- A new visualization technique, namely Halo3D, for mobile augmented reality. Adapted from an existing 2D visualization technique (Halo) [1], the technique provides visual cues on the 3D location of the off-screen POIs that are displayed on top of the 3D AR view;
- A new aggregation algorithm adapted to AR in order to avoid overlap and clutter (which is an issue identified for Halo [1]).

In the following sections, we review related literature in 2D and 3D off-screen object visualization. We then present the design of Halo3D by identifying and organizing its design elements.

## 2 Related work in off-screen object visualization

### 2.1 2D off-screen object visualization

Several visualization techniques exist to convey the location, the direction and the distance of off-screen objects in 2D.
Halo [1] allows the user to determine the location of off-screen objects: the technique traces a circle whose center is the distant object and the radius is the distance between the POI and the border of the screen (Fig. 4). The circle makes a slight intrusion on the screen. It allows the user to mentally complete the shape from the visible arc in order to find the location of the distant POI. However, as acknowledged by the authors Baudisch and Rosenholtz [1], this technique reaches its limits for the case of a high number of off-screen objects: the numerous circles overlap and make the display hard to read.

The Wedge system solves this issue by a triangle representation (Fig. 5) instead of a circle. The user extrapolates the triangle’s segments on screen to estimate where they intersect. This point of intersection indicates the location of the distant POI. The Wedge system has three degrees of freedom: rotation, aperture and intrusion into the visible space. The rotation is used to avoid overlaps with another wedge. The aperture and the intrusion provide distance cues: the farther the object is, the greater the intrusion into visible space and the wider the aperture. Wedge resolves the overlapping problem of Halo. Its main drawback is the intrusion on screen, which reduces the readability of the display for the case of AR applications.

EdgeRadar [8] is a visualization technique based on the Focus+Context principle [5]: the system uses a small portion on each edge of the screen to display non-visible areas of the environment (context) whereas the center area is the zoomed-in user’s work zone (focus) (Fig. 6). The distant POIs are displayed as in-context thumbnail icons. EdgeRadar displays all the POIs as well as their relative distance to one to another. The spatial relationship between POIs is not available with Halo or Wedge techniques. A comparison between Halo and EdgeRadar [8] shows better accuracy with the latter when tracking off-screen objects, and a users’ preference for EdgeRadar. However, this technique reaches its limits when the number of POIs is high: the small context area is overloaded and becomes illegible.

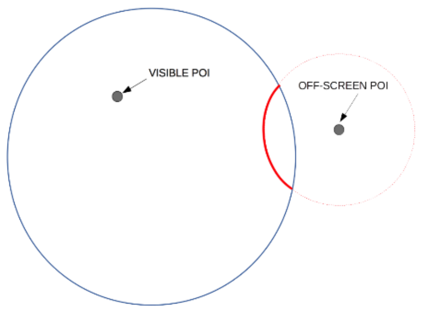


Figure 4: Halo 2D visualization technique [1] for one off-screen POI (schema inspired by [1]).

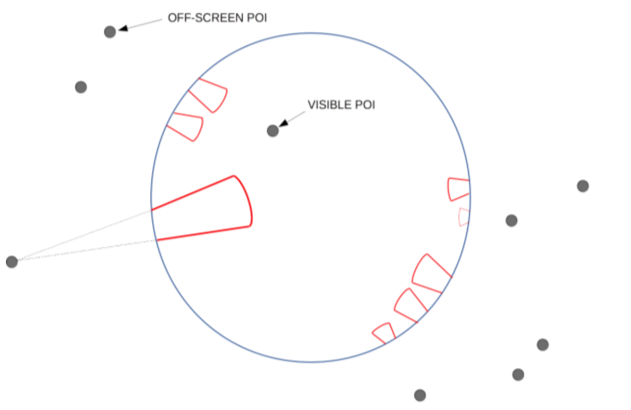


Figure 5: Wedge 2D visualization [9] technique for off-screen POIs (schema inspired by [9]).

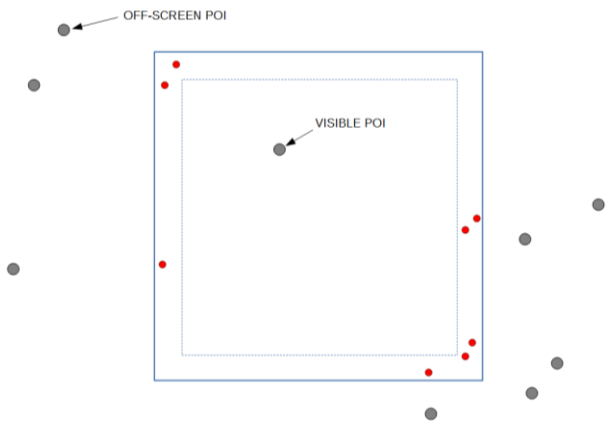


Figure 6: EdgeRadar focus+Context 2D visualization technique [8] (schema inspired by [8]).

These visualization techniques consider a 2D environment. They have been implemented only with 2D maps. Thus, they have to be adapted to take into account a 3D environment in mobile AR. 3D visualization implies displaying additional information such as the POI's height (y position) or the physical distance between the user and a POI as the crow flies. Our objective is to provide a 3D visualization of off-screen objects in mobile AR to enhance users' navigation.

## 2.2 3D off-screen object visualization

The use of 3D arrows [4] indicates the 3D direction to follow in order to reach off-screen POIs. The arrows are displayed directly onto the egocentric AR view. The navigation towards a POI is improved because the user knows if a POI is located in front or behind him. However, when the number of off-screen objects is high, the occlusion between the arrows (Fig. 7) seriously reduces the users' navigation performance.

To handle a high density of POIs in AR environments, Aroundplot [11] defines a Focus+Context visualization. Like the 2D EdgeRadar technique (Fig. 6), a small portion on each screen's edges (context zone) is used to show the location of the distant POIs. For example, the region on the right side of the screen displays a thumbnail for each POI that is reachable if the user moves the tablet to the right. The small size of the context area makes it difficult to estimate the range of motion required to reach an off-screen POI. To address this issue, Aroundplot dynamically magnifies the context in the direction of the movement, to provide more details on this part of the environment.

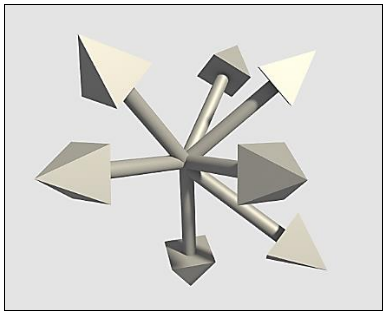


Figure 7: Occlusion between 3D arrows (schema inspired by [11]).

An experimental study [11] that compared the three visualization techniques 2D radar, 3D arrows and Aroundplot showed that the participants tracked the off-screen objects with a much lower error rate and task completion time with Aroundplot. Indeed, the occlusion between the 3D arrows led many users to give up their current task, whereas the 2D radar view did not provide any information on the height (y position) of the POIs. Aroundplot allows the user to visualize a high density of off-screen objects in all directions. However, the distance between the POIs and the user is not displayed.

The 3D Halo Circle technique [13] addresses this limitation by enabling the user to determine the direction and the distance of off-screen objects in a 3D virtual reality environment. Based on the Halo 2D visualization [1], the radius of the circle indicates the distance between the user and a POI as the crow flies. The visualization metaphor gives the feeling to the user to be inside the circle. However, the farther the POI is, the greater the intrusion into the visible space. Moreover when a POI is far away, it becomes difficult to estimate the radius of the circle and thus the location of the POI. Besides, the visualization of many POIs causes overlap: this leads to a high cognitive load, and the user is unable to mentally complete the circles in order to get the locations of the POIs.

## 2.3 Synthesis and objective

The existing off-screen POIs visualization only partially deals with the third dimension, the depth. 3D arrows are useful to point a direction in space but do not indicate whether the POI is far or close to the user. The Aroundplot technique displays POIs around the user but does not provide any distance cues. The 3D Halo Circle technique implements a solution that considers depth but whose intrusion into visible space is too important to be used in mobile augmented reality. Aroundplot is the only technique to be efficient when visualizing a high density of off-screen objects but does not consider the depth of POIs.
Based on the current state of the art and the users' need as illustrated with our three maintenance use cases, the objective is to design a visualization technique for mobile AR that shows direction and distance of off-screen POIs, while dealing with a high density of POIs. The Halo3D visualization technique presented in the following section meets this objective.

## 3 HALO3D: DESIGN ELEMENTS

The off-screen POIs visualization technique that we propose is based on the Halo 2D technique [1]. Halo (2D) has been shown to be easy to understand and minimizes the space used on screen [9]. The two key design issues related to the adaptation of Halo to mobile AR are: representation of depth (section 3.1) and management of a high density of POIs (section 3.2).

## 3.1 Third dimension: depth

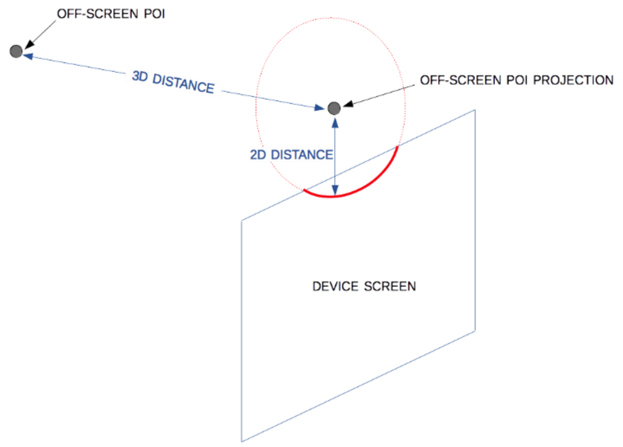
In mobile augmented reality, two types of information are useful to enhance the user's spatial knowledge of his environment:
- The range of motion required to bring an off-screen POI on screen;
- The physical distance between the user and a POI.

In mobile AR, the user works on a 3D egocentric view. However, Halo (2D) displays the location and the distance of off-screen objects on a top-down 2D orthographic view (maps). To adapt Halo to a 3D environment, the technique consists in projecting the POI onto the screen's plane of the mobile device. The distance of the projected POI from the border of the screen is called *2D Distance*, as shown in Figure 8-a. To visualize the 2D distance, the technique draws a circle with a radius equal to the distance between the projected POI and the border of the screen plus an offset to make the circle partially visible on screen. For example, we consider one POI initially displayed on screen (Fig. 8-b). Moving the device to the right makes the POI disappear into off-screen space, and an arc (a partial part of the computed circle) is displayed on the left edge of the screen (Fig. 8-c). The arc enlarges as the projected POI moves away from the screen's border (Fig. 8-d). The 2D distance, represented as an arc, conveys the range of motion required to bring a distant POI into the user's field of view: the larger the circle is, the wider is the movement required to bring the point of interest on screen.

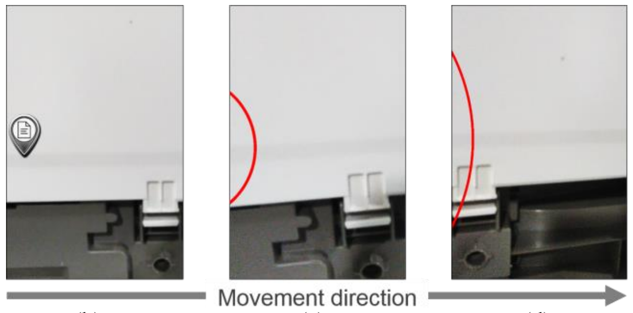
Let us now consider the case of two horizontally aligned POIs, one located behind the other: The orthographic top-down view of figure 9-a shows the user's position and the 2 POIs' locations above a machine. When the POIs are outside of the camera's field of view, the visualization technique draws two circles of identical radius (Fig. 9-b). Indeed, the corresponding projected POIs are at the same distance from the border of the screen. In this situation there is no way to determine which POI is the closest one. This information is important for instance for maintenance operators when performing campaigns, as explained in use case 3. To improve user's navigation performance, it is useful to indicate how far away from the user the POIs are located. We call this distance, *3D Distance*. For the design of Halo3D we explore 3 visual features for conveying the 3D distance:
- the circle's transparency,
- the circle's color,
- the circle's width.

Using transparency for instance, the arc of the closest POI would be more opaque than the arc of the second POI as shown in Fig. 9-c. Another design option is to use the color of the line: the arc of the closest POI would be red (color used in the Halo [1] study), whereas the arc of the farthest POI would have a neutral white color.

For the design of Halo3D, other visual features have been explored: continuity of the line (dotted line, hatching) and multicolor line. For example, the closer the user is from a POI, the more the corresponding Halo would become hatched/multicolored. When getting farther from the POI, the user would obtain a continuous arc (without cuts)/a uniform color arc. These solutions increase the visual variability of the arc. However, the industrial environment is heavily loaded with machines, cables, sensors, etc. Adding visual variability for each arc to this already dense environment in colors and shapes increases the difficulty to perceive the locations of the off-screen POIs. For the final design of Halo3D, we therefore consider three visual features, namely transparency, color and thickness, that preserve the visual uniformity of the arc displayed on screen.



(a)



(b) (c) (d)

Figure 8: Halo3D and 2D distance (a) Schema explaining how to calculate the 2D distance for one off-screen POI. (b) 1 POI is visible in the user's field of view. (c) The user moves from left to right: the POI is now off-screen and represented on screen as an arc. (d) The arc is enlarged because the 2D distance of the POI from the screen increases.

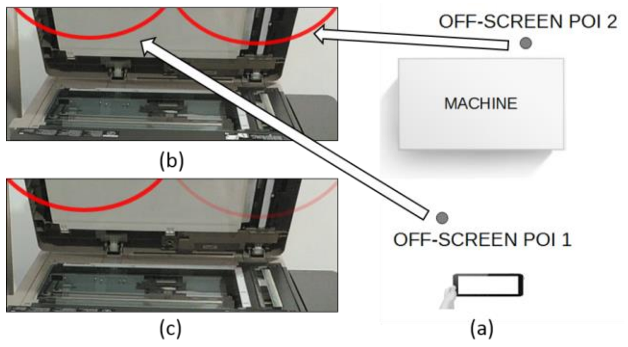


Figure 9: Halo3D and 3D distance: (a) Initial configuration: top-down orthographic view of 2 off-screen POIs horizontally aligned and located above a machine. (b) Visualization of 2 off-screen POIs: the two corresponding arcs are identical and do not allow us to determine how far away from the user the POIs are located (c) Visualization of 2 POIs using the transparency of the arc to indicate how far away from the user the off-screen POIs are located. The opaque arc on the left indicates that the corresponding POI is close to the user.

Beyond the choice of the visual feature, a second design decision is related to how to apply it according to the evolving 3D distance. When considering transparency for instance, the choice consists of making the arcs of close POIs more opaque than the arcs of

distant POIs or vice-versa. This design decision, namely visual transition decision, can be based on the relevance of close or far POIs according to the user's task.
To sum up two orthogonal design axes have to be considered to determine how to represent the 3D distance of a POI: (1) which visual feature to use (transparency, color or thickness) and (2) how to make the visual feature evolve with the 3D distance (e.g. visual transition). The couple (visual feature, visual transition) defines the *visual technique*. We are currently conducting a comparative experimental study to select the visual technique to represent 3D distance. Our assumption is that participants will prefer a visual feature that minimizes the visual overload on screen (transparency) and a visual transition that, as in real life, enhances the POI's visibility when the user gets close to them.

## 3.2 Aggregation

Halo3D inherits the identified clutter problem of Halo [1] for the case of a high density of POIs: it leads to many overlapping arcs and a visual overload of the graphical user interface on screen. To avoid this problem, a solution is to aggregate the POIs when the number of overlaps reaches a fixed limit. The HaloDot technique [7] implements this solution for the Halo 2D visualization when applied to maps. The aggregation algorithm first considers a grid overlaid on the map that divides the space into cells. It then aggregates two off-screen POIs if they are located within the same cell, reducing the number of overlapping arcs on screen. Finally, HaloDot displays on the corresponding arc the number of POIs that have been aggregated.

Let us consider a 2D map and two off-screen POIs close to the screen's edge and close to each other, such as the distance between them is greater than the distance between a POI and the border of the screen (Fig. 10-a). The two POIs will be aggregated because they are located within the same cell, and a single arc will be displayed (Fig. 10-b). To obtain two arcs, the user would have to zoom in on screen. This would space out the two POIs so that they would belong to different cells. The aggregation is therefore based on the distance between the POIs in the physical world.
Let us now consider the same example applied to mobile AR. The first difference is the camera's point of view. With HaloDot the view is an orthographic top-down one whereas in AR the point of view is egocentric. When the user performs a downwards or upwards movement, the distance between the POIs projected onto the screen's plane does not change. Only the distances between the projected POIs and the border of the screen are modified. Thus, the POIs would always be in the same cell. The aggregation would be constant as long as the user does not physically move towards the POIs. Indeed moving closer to POIs will then space out their locations. Applying the HaloDot algorithm would thus lead to an unsuitable behavior for mobile AR: POIs close to the screen's edge and close to each other would be aggregated even if their respective circles do not overlap on screen (Fig. 10-b).

The purpose of the proposed Halo3D aggregation is to reduce the overload of the graphical user interface when there are many overlapping arcs. It is a visual issue only. The Halo3D aggregation algorithm therefore examines the intersection of the circles rather than the distance between the corresponding POIs: let S be the set of circles at a given time. For each circle h, compute the set of circles that intersect h. If the set contains more than two circles, the algorithm aggregates all the corresponding POIs, and removes their halos from the initial set S of circles.

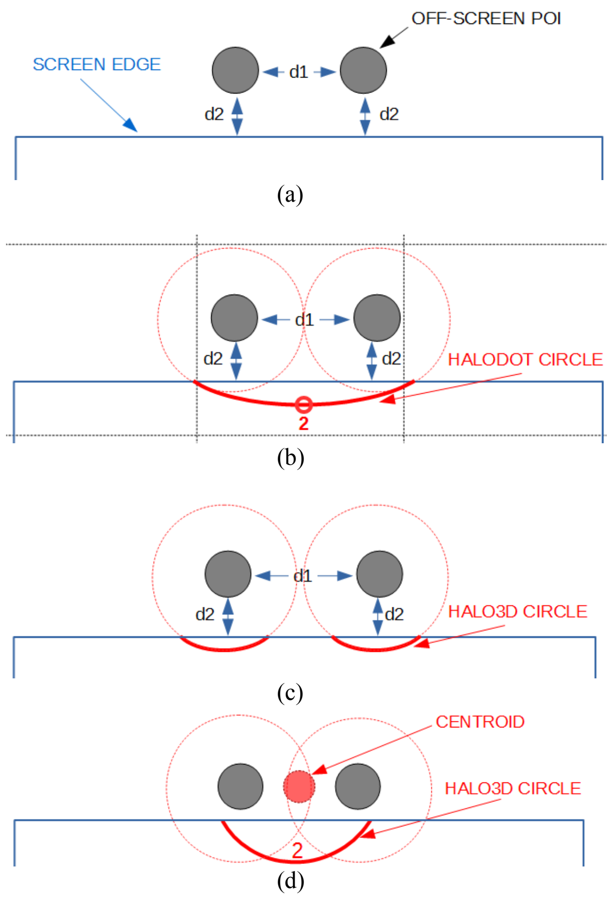


Figure 10: (a) Initial configuration with two POIs close to each other and close to the edge of the screen such as d2<d1 (b) Result with the HaloDot algorithm: the POIs have been aggregated (c) Result with the Halo3D aggregation algorithm: the POIs have not been aggregated because the resulting circles do not overlap (d) Aggregation of two POIs only when their respective circles overlap.

Finally, the technique displays the number of POIs that have been aggregated (Fig. 10-c et 10-d). The aggregation algorithm is recursive to ensure no overlap on screen. It iterates as long as there are overlapping arcs. The order between the POIs is therefore not relevant. A test with 8 POIs clustered in several groups illustrates the algorithm efficiency that progressively aggregates the POIs (Fig 11) to avoid overlapping arcs.

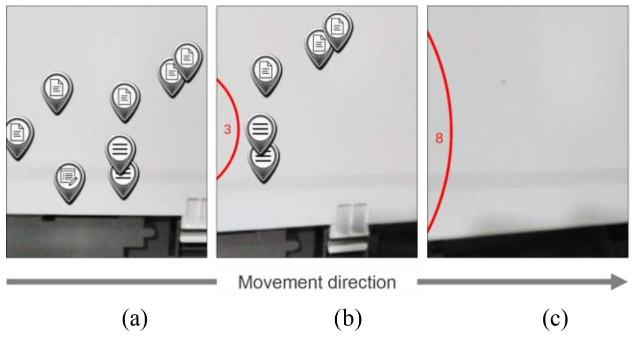


Figure 11: (a) 8 POIs on screen. (b) The user moves from left to right: 3 POIs are now aggregated. (c) All the POIs are aggregated.

To define the circle of aggregated POIs, the algorithm calculates the centroid of the POIs. The halo circle of this centroid is displayed. The calculated 3D distance is the physical distance between this centroid and the user. However, this distance alone is not enough to represent a set of POIs. It would be relevant to also consider the closest and farthest POIs of the set: the interior border (resp. exterior) of the halo circle could be used to represent the 3D distance of the closest POI (resp. farthest one). In between, the halo circle would represent the 3D distance of the centroid.

## 4 Conclusion and future work

We proposed Halo3D, a new visualization technique for mobile augmented reality. Halo3D provides 3D navigation cues on the location of off-screen points of interest, and works on a 3D egocentric view. We also proposed a new aggregation algorithm adapted to AR, in order to avoid overlapping halos on screen.
Beyond augmented reality applications, the design principles described in this paper could also be applied to virtual reality applications. For instance the distance between the user and points of interest placed around him as well as the spatial knowledge of an environment are often needed in VR games.

One identified limitation of Halo3D is related to the display of POIs located in front or on the back of the user. As future work we plan to study how to enhance the visual representation of the 3D distances between the user and POIs in order to allow the users to distinguish the POIs in front from the ones on the back.

The experimental study we are currently conducting compares the designed visual techniques to display the distances (as the crow flies) between the user and the POIs. The first collected feedback of 4 participants suggests that they prefer the transparency as a visual feature to indicate how far from the user the POIs are located. One explanation given by a participant is that the visual features “transparency” and “thickness” provide different levels of visual intrusion on screen according to the distance between the user and a POI. For example, the halo could be nearly invisible (transparency) or very thin (thickness) if the user gets close to the POI. On the contrary when using the color, the circle is always visible on the screen. Only the color changes. This first feedback verifies our hypothesis that the users prefer a visual feature that minimizes the visual intrusion on screen. Completing the study will allow us to confirm or reject our hypothesis that users prefer a visual transition that increases the POI’s visibility when the user gets close to it, as in the real world.

After the on-going experimental study in order to define the design parameters of Halo3D, we plan to compare Halo3D to other off-screen visual techniques like 2D/3D arrows, Aroundplot or an Overview+Detail visualization system (2D radar views or mini map) [3]. Previous studies have already evaluated different visualization techniques of off-screen POIs in the context of augmented reality [2, 11,12].
In particular we plan an experimental study to compare Halo3D to an arrow-based technique that we will conduct with maintenance operators. Halo3D is currently implemented in a Schneider Electric mobile AR application (Fig. 3). To reproduce an industrial environment, large posters representing the four sides of an electrical cabinet will be printed. The aim is to simulate a realistic and heavy loaded environment with several colors and shapes. The study will focus on user’s navigation performance and will compare Halo3D to the current Schneider Electric’s navigation technique, a 2D arrow-based visualization technique without aggregation. This study is planned with machine operators in several countries (in particular Japan and France).

## References

[1] Patrick Baudisch and Ruth Rosenholtz. Halo: a technique for visualizing off-screen objects. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (CHI '03), pages 481-488. ACM, New York, NY, USA, 2003.

[2] Stefano Burigat and Luca Chittaro. Visualizing references to off-screen content on mobile devices: A comparison of Arrows, Wedge, and Overview+Detail. In *Interact. Comput*, volume 23, pages 156-166. March 2011.

[3] Stefano Burigat and Luca Chittaro. On the effectiveness of Overview+Detail visualization on mobile devices. In *Personal Ubiquitous Comput*, volume 17, pages 371-385. February 2013.

[4] Luca Chittaro and Stefano Burigat. 3D location-pointing as a navigation aid in Virtual Environments. In *Proceedings of the working conference on Advanced visual interfaces* (AVI '04), pages 267-274. ACM, New York, NY, USA. 2004.

[5] Andy Cockburn, Amy Karlson, and Benjamin B. Bederson. A review of overview+detail, zooming, and focus+context interfaces. In *ACM Comput. Surv*, volume 41, 31 pages. January 2009.

[6] Rudolph P. Darken. Wayfinding in large-scale virtual worlds. In I. Katz, R. Mack, and L. Marks, editors, *Conference Companion on Human Factors in Computing Systems* (CHI '95), pages 45-46. ACM, New York, NY, USA. 1995.

[7] Tiago Gonçalves, Ana Paula Afonso, Maria Beatriz Carmo, Paulo Pombinho. HaloDot: Visualization of the Relevance of Off-Screen Objects. In *Proceedings of SIACG*, pages 117–120. 2011.

[8] Sean G. Gustafson and Pourang P. Irani. Comparing visualizations for tracking off-screen moving targets. *In CHI '07 Extended Abstracts on Human Factors in Computing Systems* (CHI EA '07), pages 2399-2404. ACM, New York, NY, USA. 2007.

[9] Sean G. Gustafson, Patrick Baudisch, Carl Gutwin, and Pourang P. Irani. Wedge: clutter-free visualization of off-screen locations. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (CHI '08), pages 787-796. ACM, New York, NY, USA. 2008.

[10] Kasper Hornbaek. Navigation Patterns and Usability of Overview+Detail and Zoomable User Interfaces for Maps, Technical Report No.2001–11 HCIL, University of Maryland. 2001.

[11] Jo Hyungeun, Sungjae Hwang, Hyunwoo Park, Jung-hee Ryu. Around plot: Focus context interface for off-screen objects in 3D environments. In *Computers & Graphics*, volume 35, pages 841-85. 2011.

[12] Torben Schinke, Niels Henze, and Susanne 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), pages 313-316. ACM, New York, NY, USA. 2010.

[13] Matthias Trapp, Lars Schneider, Norman Holz, Jürgen Dollner. Strategies for visualizing points-of-interest of 3D virtual environments on mobile devices. In *Proceedings of the sixth international symposium on LBS & TeleCartography*. 2009.