Halo Content: Context-aware View Management for Non-invasive Augmented Reality

Jason Orlosky, Kiyoshi Kiyokawa
Osaka University
Osaka, Japan
{orlosky@ist, kiyo@ime.cmc} .osaka-u.ac.jp

Takumi Toyama, Daniel Sonntag
German Research Center for Artificial Intelligence, Kaiserslautern, Germany
{takumi.toyama, daniel.sonntag} @dfki.de

ABSTRACT
In mobile augmented reality, text and content placed in a user's immediate field of view through a head worn display can interfere with day to day activities. In particular, messages, notifications, or navigation instructions overlaid in the central field of view can become a barrier to effective face-to-face meetings and everyday conversation. Many text and view management methods attempt to improve text viewability, but fail to provide a non-invasive personal experience for the user.

In this paper, we introduce Halo Content, a method that proactively manages movement of multiple elements such as e-mails, texts, and notifications to make sure they do not interfere with interpersonal interactions. Through a unique combination of face detection, integrated layouts, and automated content movement, virtual elements are actively moved so that they do not occlude conversation partners' faces or gestures. Unlike other methods that often require tracking or prior knowledge of the scene, our approach can deal with multiple conversation partners in unknown, dynamic situations. In a preliminary experiment with 14 participants, we show that the Halo Content algorithm results in a 54.8% reduction in the number of times content interfered with conversations compared to standard layouts.

Author Keywords
Augmented reality; view management; face detection; non-invasive; occlusion reduction; automation.

ACM Classification Keywords
H.5.2 User Interfaces: *Theory and methods;* H.5.2 User Interfaces: *User interface management systems (UIMS)*

INTRODUCTION
With the growing number of wearable, head worn, and head up displays, the need to manage content in a user's field of view is becoming increasingly important. Products like the Google Glass and Epson Moverio give users the ability to overlay virtual information directly onto their field of view, allowing for improved information display while mobile. There have accordingly been many attempts to address related view management problems, many of which focus on improving content readability and visibility [7]. Additionally, many algorithms have been designed to effectively manage labels on environmental objects and the resulting virtual clutter from those objects [2, 4, 6]. Other management schemes tend to focus on occlusion problems and making sure both content and labeled object are consistently visible [1, 14].

However, these methods typically focus on environment-centric text, which refers to labels that are previously registered to existing objects in the real world or in pre-defined content [7]. In contrast, user-centric items such as e-mails or personal notifications have only recently been targeted for mobile view management. Unlike labeling of known 3D objects or environments which may be stationary, view management of user-centric information must often rely on real time analysis of a more dynamic environment.

In this paper, we focus on content that can interfere with interpersonal interactions. For example, a pedestrian reading notifications or following navigation instructions in a head worn display (HWD) may stop to ask for directions. Our goal is to prevent virtual information from interfering with the following conversation or interpersonal interaction. To accomplish this, we detect faces in the scene and move content along a series of layout dependent vectors, pushing it up and away from its usual fixed screen location. This prevents text from occluding other people in the field of view (FOV), as shown in both images in Figure 1. This strategy can be applied to various social situations, chance outdoor meetings, and everyday conversations.

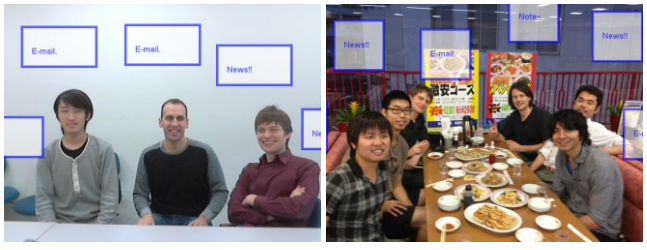


Figure 1. The Halo Content algorithm applied to billboard style text notifications in conversations with multiple participants in different environments.

IUI'15, March 29 - April 01 2015, Atlanta, GA, USA
Copyright 2015 ACM 978-1-4503-3306-1/15/03□$15.00
http://dx.doi.org/10.1145/2678025.2701375

More specifically, we first utilize face detection to search for potential interaction targets in the environment. Faces are then constantly evaluated for persistence (whether or not the detected face is still in a user's field of view despite detection failures), and for conversation potential (the probability that a persistent face will engage in conversation at a certain distance). Once the face analysis portion of the system is complete, a layout management algorithm then actively moves content to ensure that other people in the user's field of view remain visible. In many cases, the algorithm forms what looks like a halo of content around other people in the environment, as can be seen in Figures 2 C and 3 B, hence the name Halo Content. In contrast with other similar algorithms, our system can deal with numerous environmental objects, handles multiple conversation partners and screen elements, and allows for temporary off-screen placement.

PREVIOUS WORK

Up to now, many view management algorithms have been proposed to manage virtual content. A majority of related algorithms attempt to minimize occlusion of virtual labels relative to a target object. For example, Tatzgern et al. define 2D and 3D labeling techniques to ensure that both labels and leader lines do not cross or occlude each other [10]. Similarly, Reitmayr et al. propose a method for semi-automatic annotation in combination with simultaneous localization and mapping algorithms [8]. Makita et al. affixed trackers to users in the real world, and developed a method for managing annotations around the users' bodies as they moved along in real time [6]. Most of these strategies are good for virtual immersion applications, gaming, when labels are fixed to a single, stationary location, or when 3D knowledge of the scene is already known. However, they are not necessarily ideal for mobile environments or face-to-face conversations. Several other strategies for managing mobile content include physical interaction strategies, for example pasting content on a nearby surface [3]. Other, more specific automation strategies have also been applied for managing content in vehicles, such as that of Tsai et al. [11]. The most recent and closest work to this one is the dynamic text management algorithm proposed by Orlosky et al. [7]. Dynamic text management uses background color and texture to maximize viewability of user-centric text for unknown environments. Object recognition is suggested as a potential way to deal with other environmental situations, which have implemented in this work.

In contrast to most of the work mentioned above, we focus on the user's interpersonal interactions, rather than readability or virtual clutter. Our strategy is to combine object recognition techniques with layout management to achieve augmented reality that is non-invasive. Additionally, no prior knowledge of the scene is necessary, both multiple conversation partners and virtual elements can be dealt with, and the approach can be applied with other object recognition algorithms for real-time use.

METHODOLOGY

Simply put, we want the user to be able to carry out everyday conversations and activities without having to worry about closing and/or managing text. To accomplish this, three primary steps are used to prevent text from entering the conversation/gesture area. These steps include defining layouts, face detection and processing, and managing direction and movement of screen items based on position, size, and number of detected faces.

Defining Layouts

Although environment centric layout management methods can usually manage multiple elements [2, 10], current user-centric text management systems often focus on the current window of content in the user's field of view [7]. Since users are often presented with a number of different user centric data items or notifications with limited screen size, we sought to manage multiple items in a constrained space. We chose layouts in which content can simultaneously move away from objects of concentration, and still have a minimal chance of occluding other content, regardless of the size and number of faces in the user's field of view. Although a number of different layouts are possible, we predefined layouts for anywhere between 1 and 6 blocks of content for demonstration and testing, as shown in A of Figure 2. Each of these layouts is designed in such a way that if any element is moved outward, it has a minimal chance of occluding an adjacent element. Additionally, order is always preserved during movement. This means content will always appear in the order the user last left it, preventing users from having to search for icons or widgets that have moved to a different screen location. Addition or removal of elements can also be accomplished without reordering. A simple example of how content would move within a layout is shown in B and C of Figure 2. Virtual elements lie on a number of vectors that run through the center of each piece of content. Each element is then checked to see if it occludes any faces in the scene, and moved away from its original location by a user-defined distance to prevent occlusion.

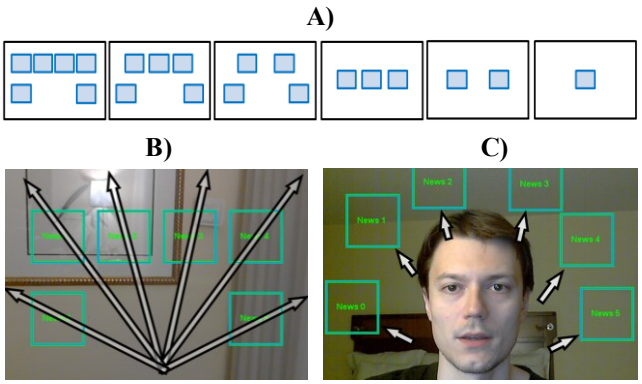


Figure 2. A) Diagram of content layouts allowing for between one and six virtual elements, B) example of movement vectors for a six element layout, and C) content that no longer occludes the conversation.

Face Detection and Persistence

The face processing library we used is from OpenCV, which implements a Haar classifier for face detection. Alone, this is not enough to guarantee consistent detection and subsequently smooth, consistent layout management. The problem of inconsistent detection, also referred to as a persistence problem, exists with many real time detection algorithms, including detection of markers for augmented reality and optical character recognition. In our case, regardless of several failed detection frames, content should still remain out of the path of the user's conversation.

In order to accomplish this, we first define a persistence variable (*Pf*) for each detected face, which functions like a threshold. Once a face is detected, it is loaded into a resizable array with a predefined *Pf*, the detected size, and *x,y* position. If multiple faces are detected, they are all inserted into the array within the same frame. After all detected faces have been processed, the persistence variable of any detected faces that had previously existed within the array is decremented by one. Any instance of a detection that has persistence of 0 or below is removed from the array. As a result, faces that have existed in at least one in the last *Pf* frames affect the content layout algorithms below. This means that even if face detection fails in several frames, content is still kept away from the person or people in the conversation. *Pf* regulates how long a face persists within the array, so a higher *Pf* (assuming more detection failures) will keep content away longer. One other benefit of this approach is that text movement exhibits a smoothing effect, since faces are more consistently present. Next, we had to figure out how to minimize occlusion of faces and content.

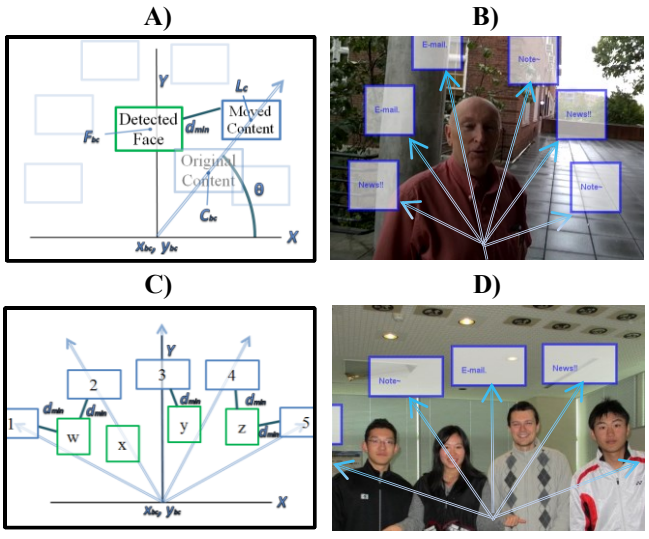


Figure 3. A) Diagram showing the geometry of displacement direction and distance calculation for an individual vector, B) corresponding managed content for one person and 6 virtual elements, C) geometry for a multi-face example with 4 people and 5 elements, and D) corresponding managed content. (Note that two elements in D are off-screen.)

Direction, Movement, and Dealing with Multiple Faces

In contrast with text readability, we are more concerned with the viewability of people in the conversation, so we designed a view management method that prioritizes visibility, is less invasive, and provides easy access to off-screen information.

Direction and Movement Algorithm

Using the previously mentioned layouts shown in Figure 2, vectors are first defined that start from a point at the bottom center of the screen (x_{bc}, y_{bc}) and run through the centroid of each virtual element at angle θ, as shown in images A and C of Figure 3. For multiple elements, the vectors run outwards towards the left, upper, and right borders of the screen as shown in C. Content can then move along each of these vectors whenever a face comes too close to a virtual element. Movement logic on the vector and distance (*Lc*) from (x_{bc}, y_{bc}) can be described by the following:

$$IF(Dv < thresh \;\&\&\; Fbc < Cbc), then:\ Lc = dmin + (Fbc - Cbc)$$

where *Dv* is the minimum distance from the detected face to the nearest vector, *dmin* is the desired minimum distance from a detected face to moved content, and *Fbc* and *Cbc* are the Euclidean distances from each face to the origin and the content block under consideration to the origin, respectively. Each element is then moved along its respective vector so that there is a final distance of *dmin* pixels between the element and the nearest face.

Multiple Faces

As seen in C and D of Figure 2, multiple faces are handled in a similar way to a single face. For every virtual element present, the distance to each face in the scene is first checked, and the element is then moved if necessary. For example, virtual elements in Figure 2 C (blue boxes) are moved *dmin* away from detected faces (green boxes). When a virtual element occludes more than one detected face, the closer face is used in the calculation. Algorithmically, this means that we loop through the array containing persistent faces, and check for nearby virtual elements in every frame. This logic is also defined by the pseudo code shown below:

```
WHILE camera is on
  RUN face detection on current frame
    ADD any detected faces to persistence array with Pf
    DECREMENT Pf of any existing faces in array by 1
      FOR each virtual element
        FOR each face in persistence array
          IF distance between face and content vector (Dv)
              of nearest element < dmin
              AND face-origin distance (Fbc) <
              content-origin distance (Cbc)
          THEN use dmin to set new content distance (Lc)
          ENDFOR //incremented through all faces
        ENDFOR //incremented through all virtual elements
ENDWHILE
```

This idea was inspired by previous strategies that employ potential fields for content movement [4]. In the case of multiple faces occluding a single element, movement is based on the nearest face with respect to the origin (x_{bc}, y_{bc}). Since processing power is a concern for mobile devices, the distance is measured from a bounding box on both rendered content and nearby faces, thus simplifying the calculation, much like Minkowski Sums are used to detect collisions in gaming applications [12]. A running average is also taken for the position of each element, so content appears to smoothly move away from any faces coming into the user's field of view. A representative sample of resulting movement for different element layouts and number of conversation partners is shown in B and D of Figure 3.

Although less likely, the case of a user's face entering from the top of the screen must be dealt with differently since content blocks are never migrated downwards to avoid occluding bodies or hand gestures. In this case, as soon as a new face comes within *dmin* of the vector corresponding to the closest element, the element is migrated up and over the face and kept at *dmin* pixels away from the face from that time forward. Though we are still testing for other exceptions, the movement scheme we propose appears to generally solve occlusion problems in this context.

EVALUATION

When evaluating our system, we conducted a simple test to find out how well the Halo Content algorithm can prevent text from interfering with a number of different conversations. To do so, we asked participants to view a variety of frames taken from 3 different videos. Simulated content was then managed for each frame with the Halo Content algorithm, and compared to corresponding standard layouts as baselines. Participants evaluated content on each frame as "would interfere" or "would not interfere."

Setup

We started by taking three videos encompassing a variety of different conversational situations, including a chance outdoor meeting, a four-person research discussion, and an in-office consultation. These videos were taken from a first person perspective using a head worn display so that head movements and interactions would be recorded. 20 frames were extracted randomly from each of these videos, and frames without faces were rejected and replaced at random. We then applied the Halo Content algorithm to 3 different standard layouts, and for combinations of 2, 3, and 6 simulated blocks of content as shown in Figure 2 A, for a total of 60 processed frames at 640x480pixels (px). We also created a corresponding set of 60 frames with the same 3 layouts and block sizes, but did not apply the management algorithm to provide a baseline for comparison. The blocks of content were displayed as white, semi-transparent billboards containing a single randomly selected text notification. Sizes differed with respect to number of blocks present, with 200x180px, 150x200px, and 100x100px for the 2, 3, and 6 block layouts, respectively.

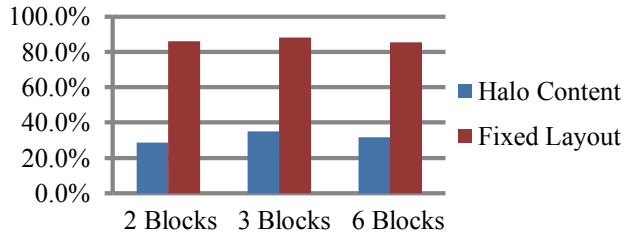
A total of 14 volunteers, 9 male and 5 female, with a mean age of 31.9, participated in the experiment. We employed a within subjects design, where each participant evaluated the management method on each of the 120 frames, for a total of 1680 evaluations. The order of conditions in each interface was randomized between participants to eliminate any ordering effects.

Results

Across all participants (group A), 86.5% of frames using layouts without management were evaluated as interfering, in comparison with 31.8% for those using Halo Content. A two-way analysis of variance shows a significant difference ($F_{(5,13)}$=26.42, P<.0001) between ratings of the two display methods across all sizes and numbers. Percentage of content rated as interfering with respect to management method and number of elements is shown in A of Figure 4.

We also noticed a clear division of ratings within the experiment. Out of the 14 participants, 5 rated a majority of content as interfering, regardless of whether it was managed by Halo Content or in a static layout. For the remaining 9 (group B), interference of the static layouts remained about the same, but resulting interference of Halo Content was significantly reduced. As shown in B of Figure 4, only 11.3% of frames managed by Halo Content were evaluated as interfering for these participants, resulting in a 73.3% reduction in interference compared to static layouts. A slight effect was found for number of elements ($F_{(2,8)}$=2.67, P<.05) for this group, which suggests that an increasing number of elements may result in increased interference.

A) Percentage of content that interfered (group A)



B) Percentage of content that interfered (group B)

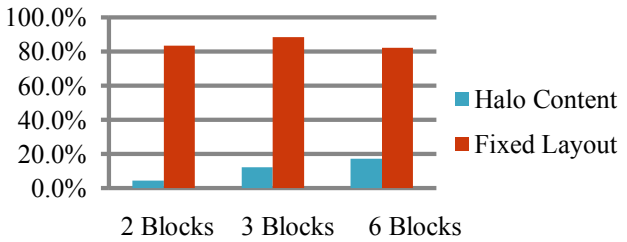


Figure 4. A) Table showing interference of the Halo Content algorithm compared to typical on-screen layouts for 2, 3, and 6 element layouts. B) The same as A, but excluding users who were unsatisfied when almost any virtual content was present in the scene.

DISCUSSION

The initial results of our experiments suggest that Halo Content may be a good way to prevent certain types of augmentative and virtual content from becoming invasive in conversational situations. As HWDs, AR applications, and virtual content increase, it will make sense to have a number of management algorithms in place for different situations. For example, Halo Content might be used for conversations, whereas 3D labeling techniques or visibility management might be used for industrial tasks. Of course, there are tradeoffs between using this vector based strategy versus other algorithms. For example, it may be difficult to mix environment relative labeling with user-centric e-mails. While vectors provide a very fast way for a user to manage mobile content, adaptations would also be necessary to view content with a shared view or perspective.

Although to some extent we provide robustness to failed face detections in our algorithm, it is important to note that this paper does not solve persistence problems completely. Face or object detection algorithms are out of the scope of this paper, but we plan to develop other methods to improve persistence as future work. Additionally, placement depth needs to be taken into account. While our strategy works well for monoscopic HWDs with a fixed focal plane, stereoscopic displays would benefit from content placed at the same depth as the user's gaze [9]. Eye tracking may be a potential solution to this problem. One other benefit of Halo Content is that it is applicable to devices other than see-through HWDs. There are many applications for immersive virtual reality, heads-up systems, and other static see-through displays. The algorithm and layout methods proposed here are easily adaptable to other virtual spaces. One good application example would be the migration of text or content away from vehicles for drivers when on the highway. The face recognition algorithm would be replaced with vehicle recognition, and layouts could be expanded for HUD sized displays.

Conclusion

In this paper, we introduce Halo Content, and find that it can reduce the invasiveness of virtual augmentations, while still providing easy access to content for the user. Since face and body detection algorithms are not perfectly robust, we also provide a way to account for facial persistence, which results in more consistent management and smoothed movement. Additionally, this framework can be used with other object or feature recognition algorithms, making it scalable and applicable to augmented reality applications in a wide variety of other fields.

REFERENCES

1. Ajanki, A., Billinghurst, M., Gamper, H., Järvenpää, T., Kandemir, M., Kaski, S., ... & Tossavainen, T. (2011). An augmented reality interface to contextual information. *Virtual reality, 15(2-3)*, (pp. 161-173).
2. Bell, B., Feiner, S., & Höllerer, T. (2001, November). View management for virtual and augmented reality. In *Proceedings of the 14th annual ACM symposium on User interface software and technology* (pp. 101-110).
3. Ens, B. M., Finnegan, R., & Irani, P. P. (2014, April). The personal cockpit: a spatial interface for effective task switching on head-worn displays. In *Proceedings of the 32nd annual ACM conference on Human factors in computing systems* (pp. 3171-3180).
4. Hartmann, K., Ali, K., & Strothotte, T. (2004, January). Floating labels: Applying dynamic potential fields for label layout. In *Smart Graphics* (pp. 101-113).
5. Ishiguro, Y., & Rekimoto, J. (2011, March). Peripheral vision annotation: noninterference information presentation method for mobile augmented reality. In *Proceedings of the 2nd Augmented Human International Conference* (p. 8).
6. Makita, K., Kanbara, M., & Yokoya, N. (2009, June). View management of annotations for wearable augmented reality. In *Multimedia and Expo, 2009. ICME 2009. IEEE International Conference on* (pp. 982-985).
7. Orlosky, J., Kiyokawa, K., & Takemura, H. (2013, March). Dynamic text management for see-through wearable and heads-up display systems. *In Proceedings of the 2013 international conference on Intelligent user interfaces* (pp. 363-370).
8. Reitmayr, G., Eade, E., & Drummond, T. W. (2007, November). Semi-automatic annotations in unknown environments. *In Mixed and Augmented Reality, 2007.* ISMAR 2007. 6th IEEE and ACM International Symposium on (pp. 67-70).
9. Tan, D. S., & Czerwinski, M. (2003). Effects of visual separation and physical discontinuities when distributing information across multiple displays. *In Proc. Interact* (Vol. 3, pp. 252-255).
10. Tatzgern, M., Kalkofen, D., Grasset, R., & Schmalstieg, D. (2014, March). Hedgehog labeling: View management techniques for external labels in 3D space. *In Virtual Reality (VR), 2014 iEEE* (pp. 27-32).
11. Tsai, H. C. (2013, November). Safety view management for augmented reality based on MapReduce strategy on multi-core processors. *In ITS Telecommunications (ITST), 2013 13th International Conference on* (pp. 151-156).
12. Van Den Bergen, G. (2001, March). Proximity queries and penetration depth computation on 3d game objects. *In Game developers conference* (Vol. 170).
13. Wierzbicki, R. J., Tschoeppe, C., Ruf, T., & Garbas, J. U. (2013). EDIS-Emotion-Driven Interactive Systems. *Other Publications of the AMEA Association*, (1).
14. Zhang, F., & Sun, H. (2005, December). Dynamic labeling management in virtual and augmented environments. *In Computer Aided Design and Computer Graphics, 2005. Ninth International Conference on* (pp. 6-pp).