Spatial Constancy of Surface-Embedded Layouts across Multiple Environments

Barrett Ens[1], Eyal Ofek[2], Neil Bruce[1], Pourang Irani[1]

[1]University of Manitoba
Winnipeg, Canada
{bens, bruce, irani}@cs.manitoba.ca

[2]Microsoft Research
Redmond, WA
eyalofek@microsoft.com

Figure 1. Transitions of application window layouts to world-fixed coordinates are derived from a common body-centric layout (a). This approach maintains relative spatial consistency while integrating application layouts into diverse surroundings (b, c). © B. Ens

ABSTRACT

We introduce a layout manager that exploits the robust sensing capabilities of next-generation head-worn displays by embedding virtual application windows in the user's surroundings. With the aim of allowing users to find applications quickly, our approach leverages spatial memory of a known body-centric configuration. The layout manager balances multiple constraints to keep layouts consistent across environments while observing geometric and visual features specific to each locale. We compare various constraint weighting schemas and discuss outcomes of this approach applied to models of two test environments.

Author Keywords

Head-worn displays; HWD; HMD; window manager; view management; spatial constancy; visual saliency.

ACM Classification Keywords

H.5.3 [Information interfaces and presentation]: User interfaces

INTRODUCTION

A new generation of head-worn displays (HWDs) is rapidly advancing, and lightweight form factors such as Microsoft Hololens are capable of reliably detecting the wearer's surroundings in real time. This spatial information can be leveraged to integrate personal information displays into the environment to support analytic tasks that rely on multiple sources of information [3, 6, 7]. However, the ideal placement remains an open research question; although much work has explored configurations in display space, little attention has been given to content layout on the surface structure of a sensed 3D model of the environment.

This paper explores the transition of window layouts from body-centric to world-based reference frames [6]. For example, imagine a mobile HWD content manager that arranges your favourite apps in a body-centric 'bubble'. When you arrive at your home or office, you can 'pin' this window layout onto the surrounding surfaces (Figure 1). Some key requirements we apply to such transitions are 1) to integrate content with existing surface structure, 2) to maintain the spatial relationship of windows so the user can locate apps quickly, and 3) to prevent app windows from occluding important objects in the environment.

We propose using the device wearer's egocentric coordinate system as a reference frame for world-fixed spatial layouts. This approach serves the dual purpose of leveraging reliance on body-centric spatial memory and maintaining consistency between different environments. However, layouts must also respect geometric differences between different spaces, for instance to avoid overlapping surface boundaries or occluding scene objects. We developed a layout manager that balances multiple constraints, including spatial constancy, visual salience, surface fit, window overlap and relative order.

RELATED WORK

A line of work following Bell et al. [2] on view management for augmented reality uses constraint-based

SUI '15, August 08 - 09, 2015, Los Angeles, CA, USA
 ISBN 978-1-4503-3703-8/15/08…$15.00
DOI: http://dx.doi.org/10.1145/2788940.2788954

algorithms for managing virtual content, typically to keep object labels from overlapping and close to their point of origin. Constraints are often combined using force-based algorithms [10], however a greedy approach has been noted to increase dynamic layout stability [9]. We instead apply a random walk approach [8, 14] that observes global layout constraints. Although some recent work has used vanishing line detection to align virtual content with real-world surfaces [13], view management generally occurs in 2D display space or on a set of view-aligned planes [21]. In contrast, we are interested in 3D spatial layouts and draw from early work by Feiner et al. [7] and Billinghurst et al. [3] that imagined multiple windows arranged in body-centric configurations or anchored to world objects.

We follow a use case similar to the Office of the Future [16], in which applications are embedded on walls and other surfaces in the environment. Thus our work overlaps with research on projection-based systems that can detect and adapt to the surrounding 3D structure [5, 15, 17]. One closely-related work [22] describes a multi-projector window-manager that maximizes available projection space, but does not address background saliency. Following the vision of such works on a HWD platform presents specific challenges, in particular the limited field of view (FoV) of current displays [6]. To help mitigate this issue, we aim to leverage spatial memory [1, 17] by applying a constraint of spatial constancy [18, 20], which has received little attention in the context of spatial user interfaces.

A LAYOUT MANAGER FOR SPATIAL CONSTANCY

We created a layout manager for see-through, stereoscopic HWDs that embeds virtual 2D application windows in a 3D environment. Using a sensor-generated model, layouts are created at run time based on the current pose (i.e. position and orientation) of the user. Each generated layout balances several constraints (described below) to arrive at a given layout. The main goals of the layout manager are threefold:

1) *Conform to surface structure* – Virtual app windows are superimposed on real-world surfaces, observing attributes such as surface boundaries and occluding objects.

2) *Maintain layout consistency* – We apply a spatial constancy constraint to maintain window positions relative to the user. Additional constraints try to maintain relative order and prevent overlap [2, 9, 21].

3) *Preserve background information* – Window positions are adjusted to prevent interference with important scene content. While there are many possible attributes to observe (e.g. colour, texture, contrast, object edges [9]), we focus on visual salience [9], which models human visual importance.

Implementation

We implemented our layout manager using Unity3D on a desktop computer with an NVIDIA Quadro 600 GPU. We created two mock environments for development and testing, resembling a typical office and living room (Figure 2a, b). Layouts are viewed through an Epson BT-100 stereoscopic HWD with 23° diagonal FoV, tethered by composite video input. By tracking the HWD with a Vicon system, virtual content appears through the HWD to be accurately superimposed on the physical environments.

Algorithm

Our layout manager follows a Monte Carlo approach [11] shown to be effective for creating constraint-bases layouts in 3D space [8,14]. Input consists of data extracted form a mesh model and a single photo of each environment. The mesh models (Figure 2g, h) are created with Kinect Fusion [12] and the photos (Figure 2a, b) are taken with a typical SLR camera with a wide-angle lens (110°). We begin by searching the vertices of the mesh models for regions of uniform surface normal, from which we extract a set of surface polygons (Figure 2c,d) using a greedy search with Hough transforms [19]. Meanwhile, we compute a saliency map of both scenes using the AIM saliency algorithm of Bruce and Tsotsos [4] (Figure 2e, f). We chose this saliency method from many available options because of the high contrast and preserved boundary regions in the saliency map. Finally, we calibrate the model with image space (Figure 2g, h) to allow saliency queries of 3D model points.

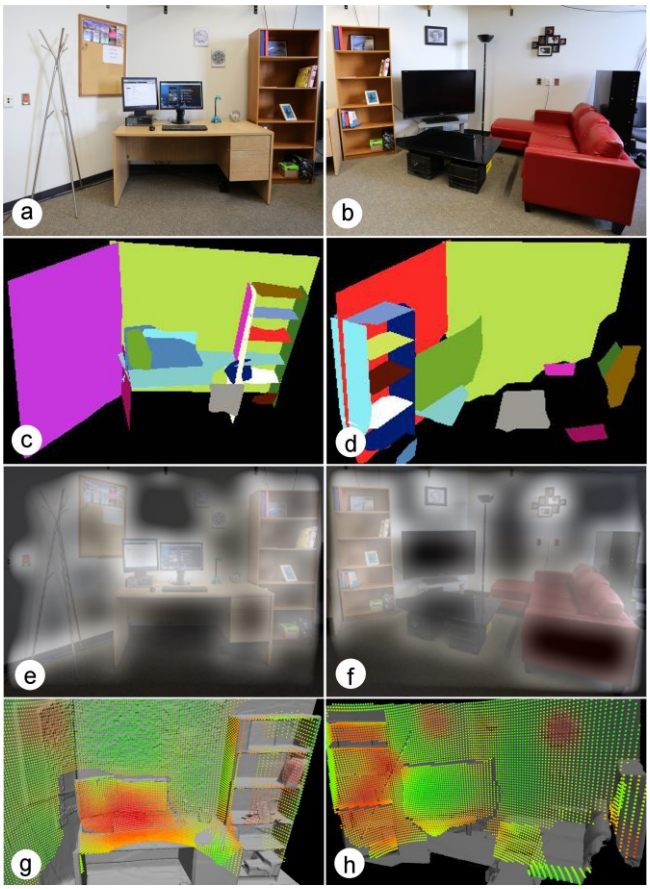

Figure 2. Office (a) and living room test environments (b). Surface polygons (c, d) generated from the mesh models (g, h). Saliency maps using AIM [4] (light regions are high salience, contrast increased for demonstration; e, f). Saliency maps projected on mesh models (red nodes are high salience; g, h).

The layout solution space is a set of all possible assignments of a set of application windows W to unique points in a discretized set P_E. We define a 'goodness' function $\boldsymbol{Goodness(L) \coloneqq \sum_i \alpha_i \cdot r_i(L_i)}$ where α_i is an optional weight, $r_i: (Oi \subseteq O) \rightarrow \mathbb{R}$ is a constraint operating on the parameters O, L is a proposed layout solution, and L_i is a layout subset containing windows with constraints O_i.

The algorithm iteratively evaluates the goodness function on layouts of randomly positioned windows. Layouts are confined to a region 90° wide × 45° high, centered on the forward view, discretized into points at increments of 5°. Windows are resized to maintain apparent angular width. We update the solution if improvement is found or with probability $p < 0.005$. This factor allows the algorithm to escape local maxima to find better solutions. We run 2000 iterations of this algorithm to generate an initial solution, then an additional 500 iterations for a 'fine-tuning' phase, where the pool of positions for each window is restricted to within 0.2m of the previous iteration. The primary phase finds a 'good' layout from the whole available space and the fine-tuning phase optimizes that layout within the local maxima. Mean run-time of the procedure is 3.26 s.

Our current implementation uses the following constraints:

Adherence enforces spatial constancy by minimizing the angular distance of a window's location from its default body-centric position (Figure 3a). The score is calculated as $1 - d^2$, where d *is* the absolute angular displacement normalized by a maximum angle of 30°.

Nonocclusion uses visual saliency to minimize the occlusion of important scene objects. The score $1 - s^4$, where s is the salience of the occupied region normalized by the scene's maximal salience value. High scores are given to windows in regions with low salience.

We apply several local window constraints: *View Direction* (to align windows closely to the user's forward view), *Surface Fit* (whether a window lies fully in a polygon), and *Line-of-Sight* (window corners are unoccluded). Additional global layout constraints are *Relative Order* of windows (whether windows maintain their spatial relations e.g. left-of), and *Overlap* (whether windows overlap others).

DISCUSSION

In preliminary trials we found the nondeterministic algorithm to be relatively consistent. However the number of iterations can be increased to improve consistency between trials or decreased to reduce run time. One advantage of our approach is that a finer discretization of space will have negligible effect on run time, whereas greedy search [9] complexity would increase with P_E.

Layout	*Adherence*	*Nonocclusion*	*View-direction*
Balanced	1	2	0
Constancy	1	0	0
Saliency	0	2	1

Table 1. Three possible constraint weighting schemas promoting different mixtures of spatial constancy and visual saliency. All other weights are set to their default value of 1.

Figure 3 shows outputs of our layout algorithm with the constraint weighting schemas defined in Table 1, which vary the balance of *Adherence* and *Nonocclusion*. The Balanced schema (Figure 3b) is ideally tuned to balance these contrasting factors in our test environments. Through trial and error, we found that the *Nonocclusion* constraint requires a higher weight than *Adherence* to prevent windows from often overlapping high salience regions, such as the area surrounding the desktop monitors in the office setting (Figure 2g). The Constancy schema (Figure 3c) has a *Nonocclusion* weight of zero. This theoretically causes each window to be projected onto the nearest surface in line with its default position (similar to Figure 1a), however the other constraints cause some deviation. Conversely, the Saliency schema (Figure 3d) has an *Adherence* weight of zero. This causes windows to congregate in low salience basins of the environment's saliency map, regardless of their distance from the default location. We provide the *View-direction* constraint in place of constancy to help prevent windows from moving to extreme distances from the user's forward view.

LIMITATIONS AND FUTURE WORK

In this work we use a body-centric reference frame for allowing windows to be found quickly given a limited FoV. However, there are other possible interpretations of spatial constancy, for instance placement of objects relative to semantically meaningful objects. We also note that applying a body-centric layout on a world-fixed frame assumes a 'primary' user pose within the room. There are many cases where this holds true, for instance in a typical office or in one's favourite cozy chair. Many interesting research questions are presented with more complex situations. For instance, how should a layout behave if a user frequently rotates between two different orientations?

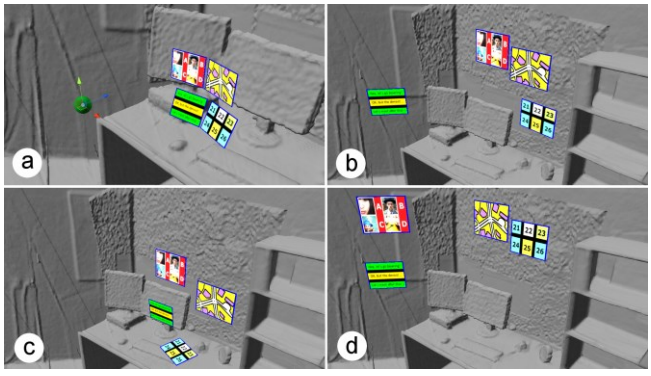

Figure 3. a) Default window locations set in 'floating' array 50 cm from viewing position (green sphere). Results of weighting schemas b) Balanced, c) Constancy, and d) Saliency.

In future, we plan to explore the benefits and trade-offs of body-centric vs world-based approaches to spatial constancy and combine these in a single layout manager.

Dynamic environments pose additional questions, for instance whether users would prefer windows to dynamically change position when someone enters the room, or to be temporarily occluded. Planned improvements include real-time extraction of the environment model and layout optimization, for instance by eliminating the mesh model or cropping to reduce raycasting operations used to detect occluded surface regions. This will allow us to explore additional design challenges, such as predicting and reacting to stimuli from people or other moving objects in the environment.

CONCLUSION

We introduce a HWD layout manager that integrates applications into the built environment. Our implementation focuses on providing spatial constancy for consistency between environments while observing local features such as surface structure and visual saliency. We apply these and some additional constraints on window layouts in two test environments with varying visual information density.

ACKNOWLEDGMENTS

We thank NSERC for funding this project.

REFERENCES

1. Agarawala, A. and Balakrishnan, R. Keepin' it real: Pushing the desktop metaphor with physics, piles and the pen. Proc. *CHI '06*, ACM (2006), 1283-1292.
2. Bell, B., Feiner, S. and Höllerer, T. View management for virtual and augmented reality. Proc. *UIST '01*, ACM (2001), 101-110.
3. Billinghurst, M., Bowskill, J., Jessop, M. and Morphett, J. A wearable spatial conferencing space. Proc. *ISWC '98*, IEEE (1998), 76-83.
4. Bruce, N. and Tsotsos, J. Saliency based on information maximization. Proc. *NIPS '05* (2005), 155, 162.
5. Cao, X. and Balakrishnan, R. Interacting with dynamically defined information spaces using a handheld projector and a pen. Proc. *UIST '06*, ACM (2006), 225-234.
6. Ens, B., Finnegan, R. and Irani, P. The Personal Cockpit: A spatial interface for effective task switching on head-worn displays. Proc. *CHI '14*, ACM (2014), 3171-3180.
7. Feiner, S. MacIntyre, B., Haupt, M. and Solomon, E. Windows on the world: 2D windows for 3D augmented reality. Proc. *UIST '93*, ACM (1993), 145-155.
8. Gal, R., Shapira, L. Ofek, E., and Kohli, P., FLARE: Fast Layout for Augmented Reality Applications, Proc. *ISMAR '14*, ACM (2014), 207-212.
9. Grasset, R., Langlotz, T., Kalkofen, D., Tatzgern, M. and Schmalstieg, D. 2012. Image-driven view management for augmented reality browsers. Proc. *ISMAR '12*, IEEE (2012), 177-186.
10. Hartmann, K., Ali, K. and Strothotte, T. Floating labels: Applying dynamic potential fields for label layout. In *Smart Graphics*, Butz, A., Krüger, A. and Oliver, P. (eds.). Springer, 101-113.
11. Hastings, W. K., Monte Carlo sampling methods using markov chains and their applications. *Biometrika* 57, 1 (1970), 97-109.
12. Izadi, S., Kim, D., Hilliges, O., Molyneaux, D., Newcombe, R., Kohli, P., Shotton, J., Hodges, S., Freeman, D., Davison, A., and Fitzgibbon, A. KinectFusion: Real-time 3D reconstruction and interaction using a moving depth camera. Proc. *UIST '11*, ACM (2011), 559-568.
13. Lee, W., Park, Y., Lepetit, V. and Woo, W. Video-based in situ tagging on mobile phones. *TCSVT 21*, 10, IEEE (2011), 1487-1496.
14. Merrell, P., Schkufza, E.,Li, Z., Agrawala, M., and Koltun, V. Interactive furniture layout using interior design guidelines. *TOG 30*, 4, ACM (2011).
15. Raskar, R., van Baar, J., Beardsley, P., Willwacher, T., Rao, S. and Forlines, C. iLamps: Geometrically aware and self-configuring projectors. Proc. *SIGGRAPH '03*, ACM (2003), 809-818.
16. Raskar, R., Welch, G., Cutts, M., Lake, A., Stesin, L. and Fuchs, H. The office of the future: A unified approach to image-based modelling and spatially immersive displays. Proc. *SIGGRAPH '98*, ACM (1998), 179-188.
17. Rekimoto, J. and Saitoh, M. Augmented surfaces: A spatially continuous work space for hybrid computing environments. Proc. *CHI '99*, ACM (1999), 378-385.
18. Scarr, J., Cockburn, A., Gutwin, C. and Bunt, A. Improving command selection with CommandMaps. Proc. *CHI '12*, ACM (2012), 257-266.
19. Silberman, N., Shapira, L., Gal, R. and Kohli, P. A contour completion model for augmenting surface reconstructions. Proc. *ECCV '14*, ACM (2014), 488-503.
20. Tak, S., Cockburn, A., Humm, K., Ahlström, D., Gutwin, G. and Scarr, J. Improving window switching interfaces. Proc. *INTERACT '09*, Springer (2009), 187-200.
21. Tatzgern, M., Kalkofen, D., Grasset, R. and Schmalstieg, D. Hedgehog labeling: View management techniques for external labels in 3D space. Proc. *VR '14*, IEEE, 27-32.
22. Waldner, M., Grasset, R., Steinberger, M. and Schmalstieg, D. Display adaptive window management for irregular surfaces. Proc. *ITS '11*, ACM (2011), 222-231.