Selecting Invisible Objects

Junwei Sun, Wolfgang Stuerzlinger*
SIAT, Simon Fraser University

ABSTRACT

We augment 3D user interfaces with a new technique that enables users to select objects that are invisible from the current viewpoint. We present a layer-based method for selecting invisible objects, which works for arbitrary objects and scenes. The user study shows that with our new techniques users can easily select hidden objects.

Keywords: 3D selection, transparency.

Index Terms: H.5.2 [Information Interfaces and Presentation]: User Interfaces— Graphical User Interfaces (GUI)

1 INTRODUCTION

Selecting a 3D rigid object is a basic task in 3D user interfaces. This task can be time-consuming, because 3D selection can involve controlling all 3 degrees of freedom (DOFs). Direct selection of 3D objects limits the user to the objects that are within their reach. Unless wireframe visualization in used, ray-based selection techniques limit selection to all visible objects [2], which typically requires only 2D input. In such systems, the only option to select invisible objects is to move the camera so that the desired object becomes visible. Here, we present a new layer-based technique that enables users to select occluded objects with ray-based selection, even if the objects are placed within non-convex parts of a scene.

We performed a user study to evaluate our extended selection technique. For this we used a task that requires the user to select a 3D object (initially) invisible in the main perspective view.

2 RELATED WORK

Transparency has been widely used in 2D and 3D user interfaces. Bier et al. [3] introduced a 2D transparent lens. Gutwin et al. [6] studied the effects of dynamic transparency on targeting performance.

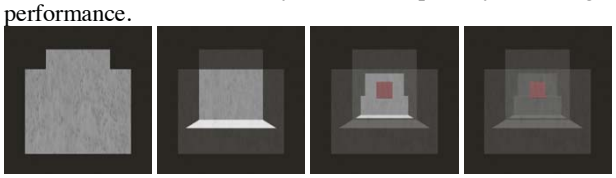

Figure 1: (a) A scene with standard opaque rendering. (b) The first layer rendered semi-transparent. (c) The first two layers rendered semi-transparent, revealing the red cube. (d) The entire object rendered semi-transparent, which corresponds to the visualization used by XPointer. It is hard to see if the cube is in front of the small ledge or behind it – the cube is in front of the ledge at position D, see Figure 2.

Agustina et al. proposed XPointer [1], an X-ray telepointer technique for collaborative 3D selection, which enables users to select initially invisible objects. Their selection technique is object-based, as users can specify which of the objects intersected by ray-casting should be selected. Their X-ray manipulator [1], is a technique for adjusting the XPointer's penetration depth, but their method only works correctly in scenes with convex objects. If there is a concave object in the scene, when the selection ray hits that concave object, the entire object is made semi-transparent by XPointer. Thus, all objects within or occluded by the concave object are then visible simultaneously. Relative to revealing the scene layer by layer, this decreases the number of depth cues available for object selection (and later manipulation).

Figure 1 and 2 show the same 3D scene with a concave object, with Figure 2 showing a side view. The position of the red cube in Figure 1 corresponds to position D in Figure 2. XPointer would make the whole base object semi-transparent, as illustrated in Figure 1(d), which makes it difficult to judge the position of the red cube, unless the user has the strong prior knowledge of the scene.

3 CONTROL-DEPTH SELECTION

With all 2D input-based interaction methods that rely on a single view, users can only select visible objects. Invisible objects cannot be selected without moving the camera or transitioning to other view types, such as wireframe views. In Figure 2 and with such techniques, if there is an object at position C, D or E in the scene, the object cannot be selected without camera navigation. We introduce our new "Control-Depth" selection technique that enables users to select completely invisible objects. The feature involves the Ctrl-key and mouse wheel actions.

To better deal with scenarios with concave scene objects in comparison to XPointer [1], we propose a new layer-based selection technique. We use the Ctrl-key to activate the depth selection mode. While users are holding the Ctrl-key pressed down (without having clicked any mouse buttons), they can push the mouse wheel forward to reveal the first, previously hidden, perspective depth layer of the scene, making the initially visible surfaces in front of them semi-transparent. Every additional mouse wheel push reveals the next layer behind the previous one. If the user pulls the mouse wheel backwards, i.e., towards them, we transition the next closer semi-transparent layer back to opaque. An occluded object will then become visible after all layers in front of it are made semi-transparent. This permits the user to simply "scroll" among all visible layers. The users can then simply select the desired object by clicking "through" the transparent layers of the scene. We use depth peeling [5] to identify the visible layers for Control-Depth selection.

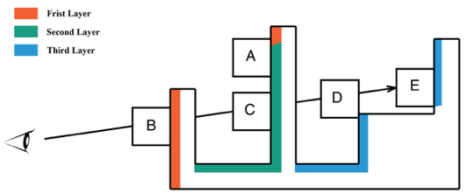

Figure 2: Illustration of Control-Depth selection. Objects C, D, and E are fully occluded, but can be revealed with Control-Depth selection (and then selected). The first, second, and third (front-facing) layer of the concave object is shown in orange, green, and blue, respectively.

* {junweis, w.s}@sfu.ca

2018 IEEE Conference on Virtual Reality and 3D User Interfaces
18-22 March, Reutlingen, Germany
978-1-5386-3365-6/18/$31.00 ©2018 IEEE

Figure 1 illustrates the process of Control-Depth selection. Figure 1(a) shows the original view of the scene. In Figure 1(b), the first layer of the scene was made semi-transparent by moving the mouse wheel forward once while holding the Ctrl-key down. In Figure 1(c), the second layer was made semi-transparent, revealing the red cube. The user can then simply select that cube by moving the mouse cursor over it and selecting it through a click. For selection, we select the first non-transparent object below the cursor.

4 User Study

We performed a user study to evaluate the performance of Control-Depth selection. We initially considered a comparison with the X-Ray manipulator in XPointer. Yet, we chose to not to do this, as the XPointer technique [1] does not provide adequate depth cues about the positioning of an object in a scene with concave parts appropriately, as discussed above. As our interaction technique is targeted at novices, we decided to disable camera navigation in our experiment, as navigation would dominate the timings and thus pose a confound.

4.1 Experiment Design

We recruited 12 (7 female) unpaid undergrad students from the local university population. Our participants had varying game expertise, with 42% playing games regularly. We built our system in the Unity game engine. We used a mouse and a keyboard as input devices.

We designed a 3D object selection experiment and asked participants to select an object from a hidden position in various scenes. The experiment compared Control-Depth selection and click-based selection in a within-subjects design. Each participant performed 20 trials, ten trials for Control-Depth selection and ten for click-based selection. We used two different levels of depth complexity for the trials. In all conditions, the scene was shown in a 4-view display, with one perspective view and three orthogonal views. In the click-based condition, users have to find the object in one of the orthogonal views and click it. Figure 3 shows two sample tasks for the Control-Depth condition. The object position is invisible in perspective views, yet it is marked with a red cross.

We asked the participants to perform the tasks as quickly as possible. We measured the time from when the users see the task to completion of selection. For the Control-Depth condition, we measured the total time, but also the individual times from pressing the Ctrl key till completion of selection.

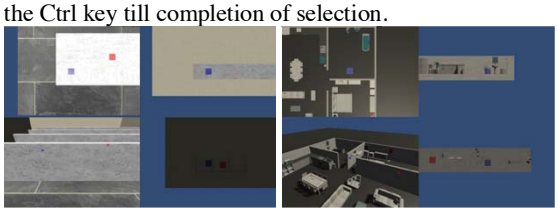

Figure 3: Two sample tasks. The object position is invisible in the perspective view, marked by red markers.

4.2 Results

The time for click-based selection is 3.07 seconds, while the Control-Depth selection takes 4.65 s on average. For Control-Depth selection, it took users only 2.31 s from pressing the Ctrl key to completion of selection, which includes the mouse movement time.

5 Discussion

The results showed that clicked-based selection is 34% faster than Control-Depth selection. This is not surprising, as Control-Depth selection requires more actions from the users. However, the difference in time is only 1.58 s. According to Brown et al.'s results [4], an average mouse movement takes about 0.8 s. Therefore, the visual search time is (3.07 - 0.8 =) 2.27 s for click-based selection, and (4.65 - 2.31 =) 2.34 s for Control-Depth selection, which are unlikely to be significantly different.

As we targeted our interaction design at novices, we only recruited novice users unfamiliar with 3D editing software. Yet, we believe that expert users will still benefit from our new technique. Their stronger mental model of the geometry of a scene together with Control-Depth selection will permit them to avoid camera navigation altogether in many instances, which will make their workflow more efficient.

We use mouse wheel operations together with a modifier key, Ctrl. This is very similar to how 3D packages use mouse operations together with modifier keys, e.g., to move objects in three dimensions. After being exposed to our new technique once, all participants had no difficulties using it during the study. Depending on the application scenario, other activation methods can be used, including through side buttons on a mouse (or controller) instead of the Ctrl key, which then also obviates the need for a keyboard.

6 Conclusion

For applications where users are selecting objects with a mouse in a 3D scene, we presented our new Control-Depth selection method, which enables users to select invisible objects even in non-convex scenes, by iteratively "peeling away" layers in front of a target object and then simply selecting it.

We plan to combine our Control-Depth selection with the sliding technique [8] for positioning. We plan to perform a user study to compare the performance of our combined technique with common 3D manipulation widgets [7]. We hypothesize that even though it takes two steps to complete the task in the sliding condition (Control-Depth selection and positioning), the combined time would still be significantly faster than with common 3D widgets. Despite the slight disadvantage in selection phase, the object will be "ready for sliding" after the Control-Depth selection phase. With our generalized techniques, users would be able to perform object selection and manipulation continuously within a single perspective view.

References

[1] Agustina and C. Sun, 2013, February. Xpointer: an X-ray telepointer for relaxed-space-time Wysiwis and unconstrained collaborative 3D design systems. In *Proceedings of the 2013 Conference on Computer Supported Cooperative Work* (pp. 729-740). ACM.

[2] F. Argelaguet and C. Andujar, 2013. A survey of 3D object selection techniques for virtual environments. *Computers & Graphics*, *37*(3), pp.121-136.

[3] E.A. Bier, M.C. Stone, K. Pier, W. Buxton and T.D. DeRose, 1993, September. Toolglasses and magic lenses: the see-through interface. In *Proceedings of the 20th Annual Conference on Computer Graphics and Interactive Techniques* (pp. 73-80). ACM.

[4] M.A. Brown and W. Stuerzlinger, 2014, May. The performance of un-instrumented in-air pointing. In *Proceedings of Graphics Interface 2014* (pp. 59-66). Canadian Information Processing Society.

[5] C. Everitt, 2001. Interactive order-independent transparency. *White paper, nVIDIA*, *2*(6), p.7.

[6] C. Gutwin, J. Dyck and C. Fedak, 2003. The effects of dynamic transparency on targeting performance. In *Graphics Interface* (pp. 105-112).

[7] P.S. Strauss and R. Carey, 1992, July. An object-oriented 3D graphics toolkit. In *ACM SIGGRAPH Computer Graphics* (Vol. 26, No. 2, pp. 341-349). ACM.

[8] J. Sun, W. Stuerzlinger and D. Shuralyov, 2016, October. Shift-sliding and Depth-pop for 3D positioning. In *Proceedings of the 2016 Symposium on Spatial User Interaction* (pp. 69-78). ACM.