CADML Technical Report 05-0722, copyright 2005

Displaying readable object-space text in a head tracked, stereoscopic virtual environment

Eric Karasuda
Computer Science Department
University of California, Berkeley

Sara McMains*
Mechanical Engineering Department
University of California, Berkeley

Abstract

Object space text, although desirable for its correct occlusion behavior, often appears blurry or "shimmery" due to rapidly alternating text thickness when used with head tracked binocular stereo viewing. Text thickness tends to vary because it depends on scan conversion, which in turn depends on the user's location in a head tracked environment, and the user almost never stays perfectly still. This paper describes a simple method of eliminating such blurriness for object space text that need not have a fixed location in the virtual environment, such as menu system and annotation text. Our approach positions text relative to the user's view frustums (one frustum per eye), adjusting the 3D position of each piece of text as the user moves, so that the text occupies a constant place in each of the view frustums and projects to the same pixels regardless of the user's location.

1 INTRODUCTION

This paper describes a simple method for displaying readable object space text in a virtual reality environment with head tracked, binocular stereo viewing. We describe the method in the context of using stereo glasses (either shutter or polarized) in combination with a "fish tank VR" [Arthur et al. 1993] monitor or projection based environment.

Although image space text (e.g. bitmapped fonts) is appropriate for 2D applications and many non-stereo 3D applications, object space text is often more convenient for 3D stereo applications, even when the text does not have a fixed 3D position in the virtual world, such as menu system text or textual annotations. Object space text frees the programmer from manually calculating visible surfaces and the parallax appropriate for each piece of text's depth. Without stereo viewing, we can simply overwrite the graphics card's frame buffer with menus and annotations using bitmapped fonts. With stereo glasses, however, because the user's eyes have slightly different perspectives on the virtual world, 3D objects not located in the plane of the screen project to different pixels for each eye. If we write text directly to the frame buffer in the same spot for each eye, it appears to be in the plane of the screen and thus appears to cut through virtual objects, avatars, and 3D cursors currently located in front of the screen plane. We can make the text appear closer to the user by offsetting where we write the text for the two eyes. However, choosing the proper offset can be difficult. If the offset is not large enough, the text will still cut into virtual objects that parallax

*{mcmains | ekarasud}@kingkong.me.berkeley.edu

indicates are in front of the text. On the other hand, if the offset is too large, the resulting large parallax cues the users to cross their eyes even though the image is actually located on a screen far away, leading to poor readability and eye strain, and potentially double vision. Therefore, for proper occlusion consistent with stereo viewing, we need to use object space text, typically placed in a plane parallel to and in front of the plane of the screen.

Unfortunately, object-space text is difficult to read if the viewpoint is not perfectly stationary because of its blurry or "shimmery" appearance. The problem is that letter thickness — the thickness of the line used to draw text — varies when outline or texture-mapped fonts scan convert to different pixels in consecutive frames. Although letter thickness only varies by a pixel, it is often only one or two pixels thickness total to begin with, so a one pixel change is significant. When letter thickness changes with user movement in a head tracked environment, the resulting text is blurry because letter thickness may change as frequently as the screen refreshes — 120 times per second for real time stereoscopic imaging. The effect is particularly disconcerting for users who think they are standing still but who are actually moving slightly. If different pixels draw the text when the user moves slightly, users who mistakenly believe they are standing perfectly still tend to think that it is the text that is moving for no apparent reason; such text movement is particularly noticeable when text is close to the edge of the screen.

2 POSITIONING TEXT RELATIVE TO THE VIEW FRUSTUMS

To prevent letter thickness from changing as the user moves, we "attach" text to the intersection of the two eyes' view frustums by making text position a function of the frustum intersection's current location and continuously adjusting as the user moves (we are interested in the area visible to both eyes, and hence the intersection of the two frustums). This makes the text appear to float at a consistent location in each of the user's view frustums and effectively fixes the pixels drawing the text, thus eliminating blurriness.

Assuming the coordinate system of the virtual world is oriented so that the middle of the screen is the origin, the positive z axis comes directly out of the screen, and the positive x and y axes lie to the right and above the origin (in the plane of the screen), we specify the position of text in two parts to place it relative to the view frustum intersection. The first is a proportion p of the distance z_{user} between the user and the screen; the plane $z = z_{text} = p \cdot z_{user}$ contains the text. The second is horizontal and vertical offsets which specify the text's location in the cross section of the frustum intersection with $z = z_{text}$. Specifying text positions in this manner causes the set of pixels drawing each portion of text to stay the same regardless of the user's location.

3 PROOF

Assume that the position of each piece of text is specified as described above. Our goal is to show that, for each eye, the set of pixels which draws the given text stays the same regardless of the user's location.

It suffices to show that, regardless of the user's position, the two eyes' frustum cross sections corresponding to $z = z_{text}$ have equal, constant dimensions and that the overlap of the two cross

sections is constant. If each frustum's cross section has constant dimensions, a pair of offsets for a given frustum cross section will always map to the same pixels. If, in addition, the cross sections have equal area and overlap a constant amount, the cross sections will remain horizontally offset by a fixed amount and the pixels drawing each portion of text for each eye will remain constant regardless of the user's position.

Figure 1 shows that the widths $|GD|$ and $|CH|$ of the two frustum cross sections are equal since $\triangle EAB \sim \triangle EGD$, $\triangle FCH \sim \triangle FAB$, and $\frac{base(\triangle EAB)}{height(\triangle EAB)} = \frac{m}{z_{user}} = \frac{base(\triangle FAB)}{height(\triangle FAB)}$ implies $\frac{base(\triangle FCH)}{height(\triangle FCH)} = \frac{base(\triangle FAB)}{height(\triangle FAB)} = \frac{base(\triangle EAB)}{height(\triangle EAB)} = \frac{base(\triangle EGD)}{height(\triangle EGD)}$. Similarly, the heights of the two frustum cross sections are equal. To show that the intersection of the two cross sections is constant, it suffices to show that $|CD|$ is constant since the two cross sections are the same height.

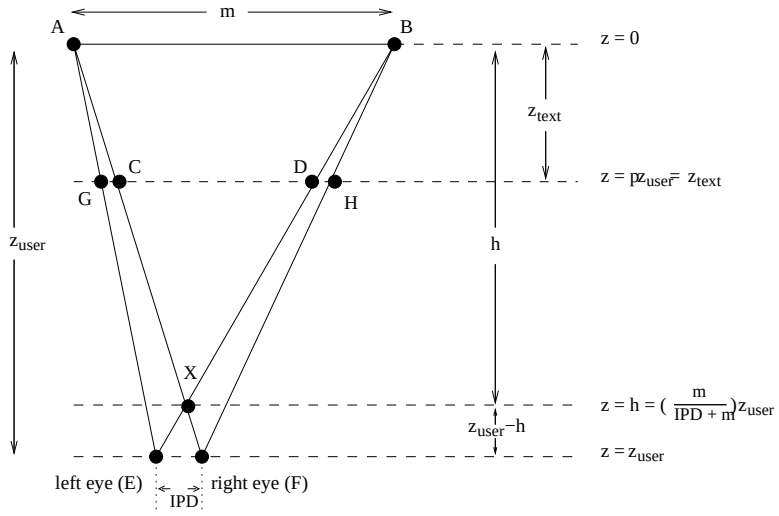

Figure 1: *Top view looking down at the user and screen. The screen extends from A to B. The constant p determines the plane $z = p \cdot z_{user} = z_{text}$ in which the text is placed. The width of the intersection of the two view frustums, given by the length of CD, is constant regardless of the user's location. Also note that $|GD| = |CH|$ by similar triangles.*

To see that $|CD|$ is constant, see Fig. 1 and verify planes $z = z_{user}$, $z = \frac{z_{user} \cdot m}{IPD+m}$, and $z = z_{text}$ are parallel to the plane of the screen, so triangles ABX, CDX, and FEX are similar. By the similarity of $\triangle ABX$ and $\triangle FEX$, we have $\frac{height(\triangle ABX)}{height(\triangle FEX)} = \frac{base(\triangle ABX)}{base(\triangle FEX)}$, or $\frac{h}{z_{user}-h} = \frac{m}{IPD}$, which gives $h = \frac{z_{user} \cdot m}{IPD+m}$. By the similarity of ABX and CDX, we have $\frac{height(\triangle ABX)}{height(\triangle CDX)} = \frac{base(\triangle ABX)}{base(\triangle CDX)}$, which gives $|CD| = m\frac{m \cdot z_{user}/(IPD+m) - z_{text}}{m \cdot z_{user}/(IPD+m)} = m - p(IPD+m)$. Thus $|CD|$ is constant.

4 EVALUATION

The above method of attaching text to the view frustum intersection provides a way to display readable, object space text when pixel space bitmapped fonts are inappropriate or impractical. We have implemented it using GLF outline fonts [Podobedov accessed 2004] for both annotations

and a menuing system in a head tracked, stereoscopic virtual reality environment. We use Stereographics liquid crystal shutter glasses and a Fakespace ImmersaDesk with a Phantom haptic arm mounted on top to control the stylus. For annotations, we have found using $p \approx \frac{1}{4}$ as the proportion factor, placing text in a plane a quarter of the distance from the user to the screen, works well with our setup. This puts the annotations close enough to the user to allow easy interaction with the stylus, but far enough away that they don't occlude more of the viewing volume than necessary. For menus, we set p based on the stylus position when the user requests the menu.

The frustum-attached text is far easier to focus on than text placed at fixed positions in the virtual world. The fact that the text follows the user as the user moves is not at all distracting because the motion is smooth and predictable, though annotation text that moves away from the object being annotated does require a method for visually associating the annotations with specific locations, which might not be necessary otherwise. Typical solutions include using small icons in the virtual world, possibly color coded [Craig and Zimring 2002; Jung et al. 2002], or lines connecting the annotation to the feature being annotated [SolidWorks Corp. 2004; Par 2003; Bell et al. 2001]; we describe a method for 3D routing of the lines for maximum clarity in [Karasuda and McMains 2004]. In addition to readability, our method possesses a few other potentially desirable attributes:

- Text never falls outside the view frustum intersection as a result of user movement – typically a desirable attribute in menu systems and annotations. (Instead we let the user control removing menus, and remove annotations when the object being annotated is no longer in view.) In comparison, Figure 2 shows how easily text with a fixed position in the virtual world falls outside the view frustum intersection when the user moves.

- When the user approaches the screen, text rarely comes close enough to the user to cause double vision and the associated discomfort, and never ends up behind the user. Text only gets close enough to the user to cause discomfort if the user is extremely close to the screen since text is placed at a proportion of the distance between the user and screen. In contrast, text fixed in the virtual world causes discomfort more readily when the user approaches the screen.

- Users may want text placed in the periphery of their visual field so as not to obscure more important objects in the center. For example, textual annotations too large to overlay on the objects they annotate may be best placed toward the edges of the view frustum intersection so as not to obscure important objects in the center. With our placement scheme, text automatically updates its position as the user moves, remaining at the periphery if that is where it was originally placed.

ACKNOWLEDGMENTS

This work was supported in part by Ford Motor Company and UC Micro.

References

ARTHUR, K., BOOTH, K., AND WARE, C. 1993. Evaluating 3D task performance for fish tank virtual worlds. *ACM Transactions on Information Systems, 11,* 3 (July), 239–65.

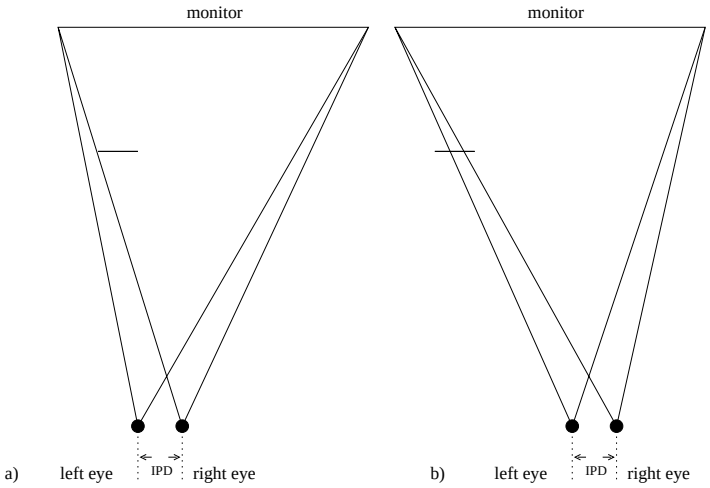

Figure 2: *Top view showing the effect of user movement and head tracking on text visibility when text is at a fixed location in the world and not attached to the view frustums. The horizontal line represents text at a fixed location in the world. In (a), the text is fully visible, but at the periphery of the user's vision. When the user moves to the right, as shown in (b), the text is no longer fully visible. If the user were to move to the left instead, the text would occlude the centers of the view frustums, including any object the text might annotate.*

BELL, B., FEINER, S., AND HOLLERER, T. 2001. View management for virtual and augmented reality. In *UIST*, 101–110.

CRAIG, D. L., AND ZIMRING, C. 2002. Support for collaborative design reasoning in shared virtual spaces. In *Automation in Construction*, vol. 11, 249–259.

JUNG, T., GROSS, M., AND DO, E. Y.-L. 2002. Annotating and sketching on 3d web models. In *International Conference on Intelligent User Interfaces (IUI)*, 95–102.

KARASUDA, E., AND MCMAINS, S. 2004. Textual annotation in a head tracked, stereoscopic virtual design environment. In *Proceedings of the ASME Design Engineering Technical Conferences*, vol. 4, 527–536.

PARAMETRIC TECHNOLOGY CORPORATION. 2003. *Pro/ENGINEER Wildfire*.

PODOBEDOV, R., accessed 2004. GLF. http://astronomy.swin.edu.au/˜pbourke/opengl/glf/.

SOLIDWORKS CORP., 2004. SolidWorks SDK.