Correcting Brightness Discontinuities in Fish Tank Virtual Reality Systems

Michael Treadgold, Kevin Novins, and Geoff Wyvill

Dept. Computer Science, University of Otago,
PO Box 56, Dunedin, New Zealand
miket5@cs.otago.ac.nz

Abstract

When a computer screen is viewed at different angles the amount of light reaching the viewer's eyes changes. This effect is particularly noticeable if two or more screens are aligned perpendicularly to each other. This situation occurs frequently in Fish Tank Virtual Reality (FTVR) systems with two or more screens. While physical restrictions make a complete solution impossible, a partial solution is presented that is easy to implement in hardware and requires only two simple experiments. The first finds the relationship between brightness and viewing angle, while the second finds the relationship between brightness and the value of the screen's pixels. The results show that we can improve the situation significantly at the cost of reduced overall brightness.

Keywords: *Fish Tank Virtual Reality, brightness correction, head-coupling, multiple screens*

1 Introduction

When a computer screen is viewed at an angle its observed brightness changes. Usually this is not particularly noticeable, but when two or more screens are placed perpendicularly, or indeed at any angle (excluding 180°), to each other the difference in brightness becomes apparent. The diagrams in Figure 1 illustrate this effect.

FTVR systems are a class of virtual reality systems that track the position of the viewer's eyes and draw the correct perspective view of the virtual scene for that position. It is beneficial to incorporate more than one screen into a FTVR system as this reduces the clipping of objects near the edge of the screen, and provides the user with a larger working volume. It is important to supply the viewer with correct visual information in order to maintain the illusion of three-dimensionality. Irregularities such as brightness discontinuities are distracting for the user and may even disrupt the illusion [2]. This problem occurs in such multi-screen FTVR systems as the CAVE [1], Cubby [3], and the Wedge [4].

We set up a new FTVR configuration which has come to be known as the Slice. It consists of two screens positioned at right angles to each other, as if they were two neighbouring faces of a cube. This configuration gives the user the freedom to move 270° around the outside of the setup, and allows them to view all sides of a virtual object placed inside the Slice (see Figure 2). Unfortunately this configuration is particularly susceptible to the brightness discontinuity effect, so we set out to correct it.

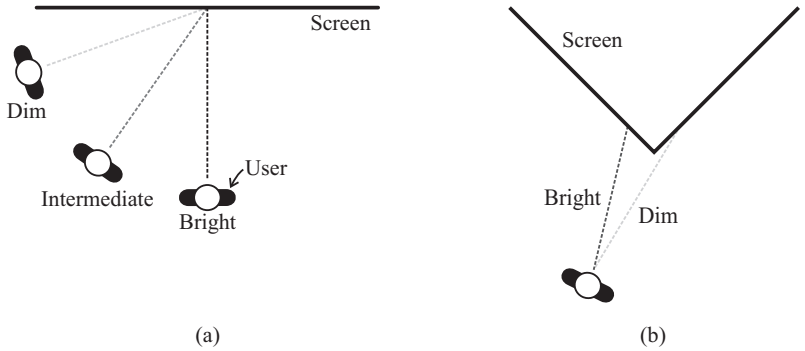


Figure 1: (a) Viewing a single screen at different angles. (b) Viewing two screens.

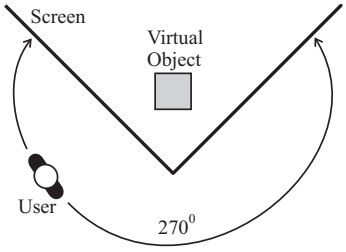


Figure 2: Viewing a virtual object inside the Slice.

2 Experiments

Before we could perform the correction we needed to find out the characteristics of the screen. We performed two brightness experiments, both of which used a luxmeter fitted to a collimator. The collimator, a long hollow tube, ensures that the luxmeter only measures light travelling directly towards it. The first experiment was designed to determine how the brightness of the screen changes as the viewing angle changes. The second was designed to determine how the brightness changes as the image on the screen is dimmed. This dimming is a digital adjustment; a per pixel operation that individually scales the red, green, and blue colour components.

The experimental setup for the first experiment is shown in Figure 3. A white image was displayed on the screen and measurements were made at several angles. The resulting distribution is shown in Figure 4(a), where the brightness is normalised to one when the viewing angle is zero.

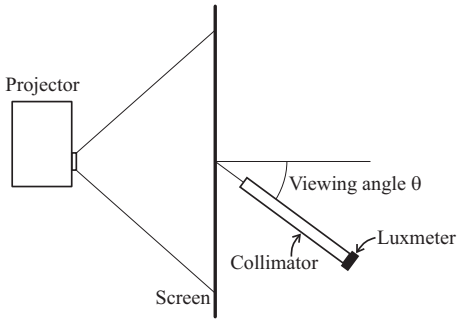
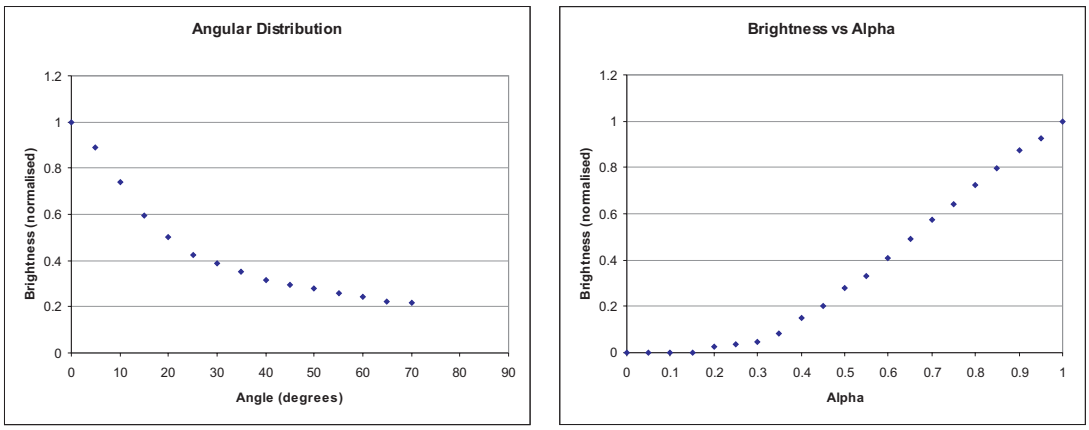


Figure 3: Experimental setup to measure the angular distribution of light.

The brightness of the screen can be changed simply and quickly[1] by using the hardware on the 3D graphics card in conjunction with OpenGL's glBlendFunc() function [6]. We introduce a parameter called α which is used to scale the red, blue, and green components of each pixel.

The second experiment found out the relationship between α and the brightness of the displayed pixels. With the luxmeter pointing directly at the screen α was varied and brightness measurements were taken. The results of this experiment are shown in Figure 4(b), where the brightness has been normalised to one when α is one.



(a) (b)

Figure 4: Experimental results. (a) Brightness vs. viewing angle. (b) Brightness vs. alpha.

3 Implementation

From the first experiment described above we define the function $B_\theta(\theta)$, which gives the ratio of the brightness of a pixel when viewed at angle θ to its brightness when viewed at an angle of zero degrees. From the second experiment we define $B_\alpha(\alpha)$, which is the ratio of the brightness of a pixel scaled by α to the brightness of a pixel with no scaling. In both cases, values in between the measured sample points were linearly interpolated. Values of $B_\theta(\theta)$ were extrapolated past 70° by assuming that $B_\theta(90) = 0$.

We chose the pivotal point for measuring the viewing angle as the front corner of the two screens because this is where the discontinuity occurs. It is a simple matter to calculate the viewing angle because, in our system, the position of the user's head is being tracked. As the system is symmetric, for the rest of this discussion we will consider only one of the Slice's two screens.

Accounting for both the effect of the scale factor α and the viewing angle θ, the observed brightness, b, of a pixel is determined by the equation:

$$b = B_\alpha(\alpha) B_\theta(\theta). \tag{1}$$

Our goal is to control the α parameter to keep b constant, regardless of θ. In order to do this, we solve the above equation for α:

$$\alpha = B_\alpha^{-1}\left(\frac{b}{B_\theta(\theta)}\right). \tag{2}$$

[1] it takes approximately 3ms to process a 1024×768 pixel, 32 bit image with our GeForce256 graphics card.

The function B_α^{-1} is defined within $[0, 1]$ but the argument in Equation 2 can become greater than one, and hence this equation cannot always be solved. No matter what value of b we choose, there will be some viewing angles that we cannot correct for completely. The set of these angles, the smallest of which we define as θ_c, forms what we call the *dead zone*. The size of this zone will be reduced if we lower b, however this will also dim the overall brightness of the display. We therefore set the size of the dead zone by making b equal to $b_{\min}$, which is the minimum brightness level that is acceptable to the user. This value is determined by viewing a completely white screen head-on (viewing angle is zero) and choosing an appropriate value for α. This value of α is called $\alpha_{\min}$ and is related to the minimum brightness by:

$$b_{\min} = B_\alpha(\alpha_{\min}).$$

Furthermore, the critical angle θ_c, is found by substituting $b = b_{\min}$ and $\alpha = 1$ into Equation 1. This gives us:

$$\theta_c = B_\theta^{-1}(b_{\min}).$$

To implement the brightness correction we divide the viewing area into the three regions shown in Figure 5. In Region 1, $\theta \in [-90, 0]$, only one screen is visible, so brightness correction is not important. We chose to set α to the same level used for $\theta = 0°$, which is $\alpha_{\min}$. In Region 2, $\theta \in (0, \theta_c)$, perfect brightness correction is used and α is set by Equation 2. Region 3 is the dead zone, $\theta \in [\theta_c, 90]$. Perfect brightness correction is not possible, so we set α to the maximum value of one.

4 Results

The photographs in Figure 6 show the improvement that this method makes to the brightness discontinuity. A colour copy of this document can be found on our web page [5]. Even within the dead zone the discontinuity is improved. What is not so apparent is the fact that the overall brightness of the system is less after the correction is applied. Also, while the white background appears much smoother across the boundary, the close-up photos in Figure 7 show that the correction is not complete. The left side of the teapot lid is darker than the right side, and the contrast on the right side of the teapot is less than on the left.

One factor that affects the brightness correction is the natural difference between projectors. Different projectors have slightly different colour characteristics, and may need adjusting prior to applying the correction. Also, the brightness of the image changes over

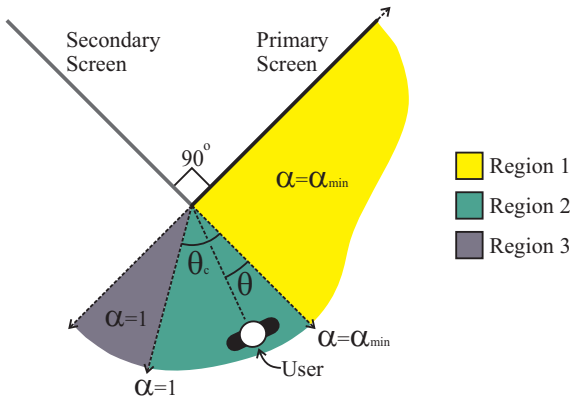


Figure 5: The viewing area divided into three regions.

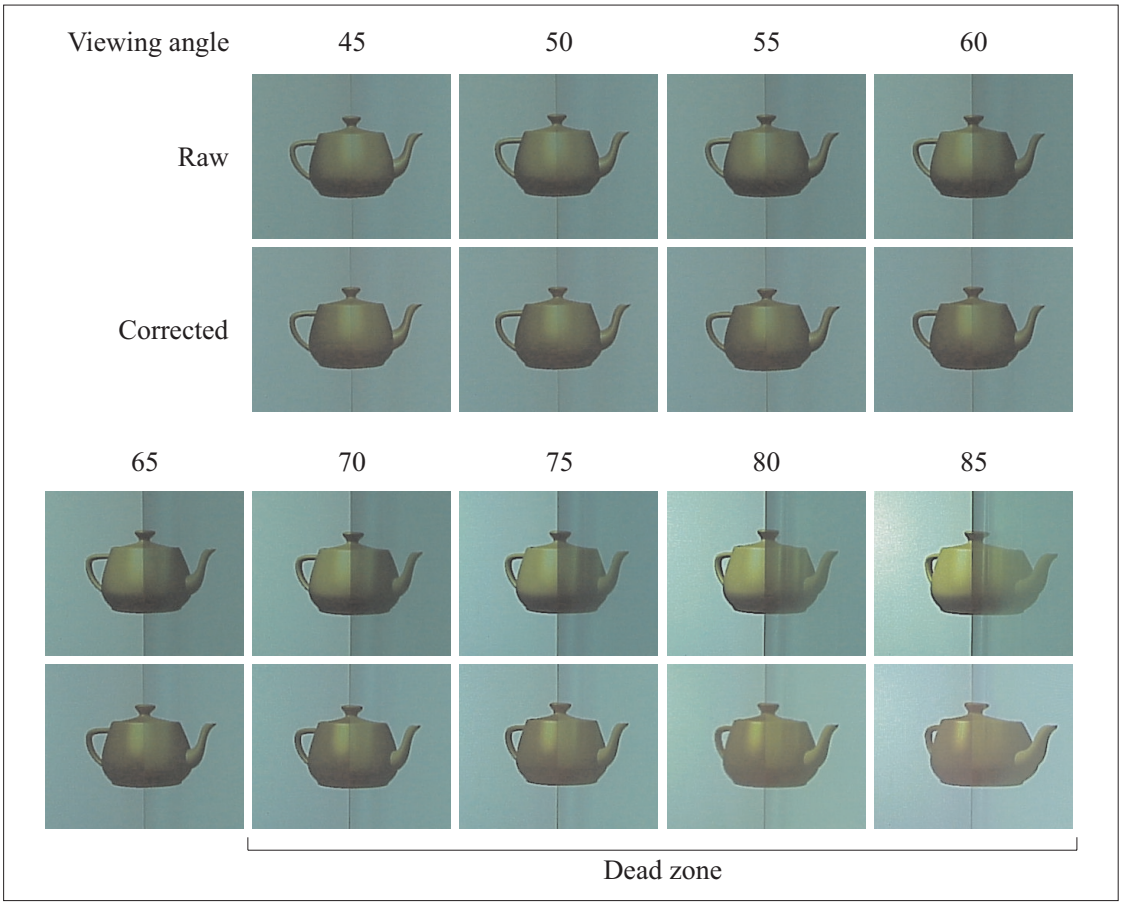


Figure 6: The effect of the brightness correction ($\alpha_{min} = 0.46$, $\theta_c = 68°$).

Figure 7: Imperfections in the brightness correction. A close-up of the teapot viewed at 65°.

the projected surface. While we did not find this to be a problem, it may be necessary to consider in some systems.

5 Future Work

The imperfections in the brightness correction are believed to be caused by different levels of brightness reacting differently to the change in viewing angle. Research could be done to determine these relationships and devise an improved correction method. Such a method

would require adjusting individual pixels differently. This would make the whole process more expensive, primarily because it would not be possible to use the hardware on the graphics card. Alternatively, it may be possible to refine the current method by considering an 'average' relationship between brightness and viewing angle.

The fact that the colour of the virtual scene may not be completely saturated could be exploited to reduce the size of the dead zone. This could be done by allowing the scale factor α to become greater than one. In this situation objects that are not saturated would appear correct, but highly saturated objects could appear 'washed out', with a reduction in contrast.

6 Conclusions

The described method provides a starting point for systems that suffer from brightness discontinuities. Users of the system have to be prepared to tolerate a certain amount of brightness loss and some discontinuity, however, the experiments required are modest and the method is simple and cheap to implement. Even though the dead zone is noticeable, the result is an improvement over an uncorrected system.

References

[1] C. Cruz-Neira, D. J. Sandin, T. A. DeFanti, R. V. Kenyon and J. C. Hart: The CAVE: audio visual experience automatic virtual environment in: *Communications of the ACM* vol. 35 (Jun. 1992) pp. 64–72.

[2] J. Djajadiningrat: *Cubby: What You See Is Where You Act* Ph.D. thesis (1998).

[3] J. P. Djajadiningrat, G. J. F. Smets and C. J. Overbeeke: Cubby: a multiscreen movement parallax display for direct manual manipulation in: *Displays* vol. 17 (1997) pp. 191–197.

[4] H. J. Gardner, R. W. Boswell and D. Whitehouse: The WEDGE Emmersive [*sic*] Projection Theatre in: *The Simulation Technology and Training (SimTecT99) Conference* (Mar. 1999) .
`http://wedge.anu.edu.au/`

[5] M. Treadgold, K. Novins and G. Wyvill: Correcting Brightness Discontinuities in Fish Tank Virtual Reality Systems (2000)
`http://www.cs.otago.ac.nz/gpxpriv/public_html/homepage.html`
Last accessed: October 2000 .

[6] M. Woo, J. Neider and T. Davis: *OpenGL Programming Guide* Addison-Wesley Developers Press second edition edn. (1997).