

REGION WARPING IN A VIRTUAL REALITY SYSTEM WITH PRIORITY RENDERING

Yang-Wai Chow, Ronald Pose and Matthew Regan
School of Computer Science and Software Engineering, Monash University
Clayton, 3800 Victoria, Australia
yang.wai.chow@csse.monash.edu.au
ronald.pose@csse.monash.edu.au

ABSTRACT

The ultimate aim of virtual reality is to present the user with an illusion of reality within a virtual environment. Visual artefacts that appear in a virtual reality system's computer generated graphics can easily destroy this artificial sense of reality. Therefore in order to maintain an illusion of reality, it is essential to eliminate or hide such visual artefacts from the user. This paper investigates the implementation of region warping, a methodology devised for the purpose of masking scene tearing artefacts. These scene tearing artefacts are a side effect from implementing large object segmentation together with a prioritized rendering technique, which was developed for use in conjunction with a specialized virtual reality graphics display controller, known as the Address Recalculation Pipeline. Region warping introduces slight distortions in a scene as a result of compensating for such tearing artefacts. This paper describes the basis for region warping and discusses the results of experiments that were conducted using this warping technique.

KEYWORDS

Address recalculation pipeline, distortion, priority rendering, region warping, visual artefacts.

1. INTRODUCTION

Immersive virtual reality systems attempt to present the user with an illusion of reality within a virtual world. One of the major aspects involved in portraying a believable artificial environment, is in the presenting of computer generated 3-dimensional imagery to the user. However certain graphics artefacts in a scene, such as aliasing artefacts or scene tearing artefacts, can completely destroy the illusion of reality that such virtual reality systems attempt to create.

In light of this, it is therefore necessary to compensate for such unwanted artefacts before a virtual reality system can place a user in a realistic believable virtual environment. There are often tradeoffs involved in such compensation techniques, such as blurring effects or distortions in the display. Nevertheless the human visual system is generally more sensitive to glaringly obvious artefacts, as compared to noise [Cook 1986] or slight distortions. As a result, the user often perceives higher quality in scenes rendered using techniques that mask or hide artefacts, as compared to scenes that do not compensate for these artefacts.

This paper is concerned with addressing scene tearing artefacts that surfaces as a consequence of implementing a particular rendering technique in conjunction with an Address Recalculation Pipeline (ARP) virtual reality system. The ARP virtual reality system was designed in order to reduce the latency experienced by a user during user head rotations. A technique known as priority rendering was developed for use together with this system for the purpose of reducing the overall rendering load, as this would allow for the possibility of rendering more complex and realistic scenes [Regan and Pose 1993; 1994].

Additional rendering load reductions have been achieved by implementing large object segmentation with priority rendering [Chow et al. 2005a]. However this segmentation introduced potential discrepancies between the shared vertices of segmented objects, resulting in the user perceiving scene tearing artefacts. Region warping was devised as a method to compensate for such tearing artefacts, by perturbing vertices in order to hide the tearing. As a byproduct of these perturbations, slight distortions in the displayed images were introduced. The human visual system has a finite resolving ability, it is therefore important to

understand the level of distortion and identify threshold conditions whereby a user cannot distinguish between the compensated image and the ideal image [Naiman 1998]. This paper presents the results of initial region warping experiments and examines the distortions introduced by the implementation of this approach.

2. PREVIOUS WORK

In order to describe the basis for region warping, this section first provides a background of the Address Recalculation Pipeline virtual reality system, as well as the prioritized rendering technique designed for use in conjunction with this system.

The Address Recalculation Pipeline (ARP) is a specialized graphics display hardware architecture designed for use in immersive Head Mounted Display (HMD) virtual reality systems. The purpose of the ARP is to reduce the latency experienced by a user during user head rotations, and succeeds in doing so by delaying viewport orientation mapping until after the rendering process. In order for the ARP to perform viewport mapping post rendering, the scene that surrounds the user's head has to be rendered to display memory, an example of this is shown in figure 1. The ARP then maps relevant sections of the image already residing in display memory based on the most up-to-date user head orientation information. In doing so, user head orientation information only has to be accurately known just before the first pixel has to be sent to the HMD, thus effectively removing the usually lengthy rendering time from user head rotational latency.

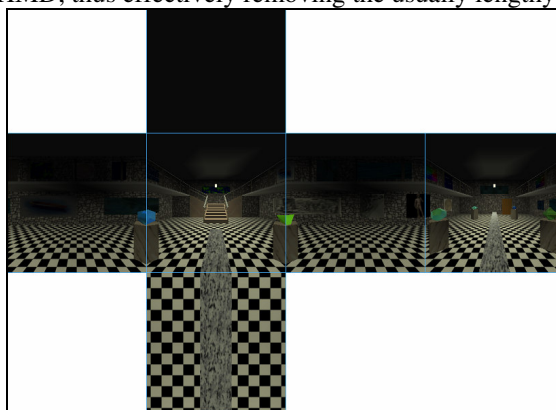


Figure 1: Scene surrounding the user's head.

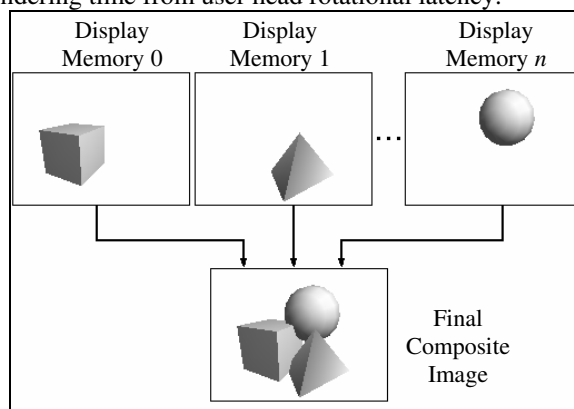


Figure 2: Image Composition.

Priority rendering was a technique developed in conjunction with the ARP, with the aim of reducing the overall rendering load. Using multiple display memories, and possibly multiple rendering engines, different virtual world objects can be rendered onto different display memories before being composed to form an image of the final scene. This is illustrated in figure 2. In this way, each of these separate display memories can be updated individually at different update rates. In the scene that surrounds the user's head, when the user's head orientation changes, sections of the scene closer to the user will appear to change more rapidly compared to sections further away from the user. Priority rendering therefore allows different sections of the scene to be updated at different update rates. For further details refer to Regan and Pose [1993; 1994].

Initial implementation of priority rendering required the computation of individual object validity periods. Validity periods were estimates regarding the length of time an object's image in display memory would remain valid, before requiring an update. Upon acquiring these estimates, virtual world objects would then be sorted according to these time validity periods and be assigned individually to the separate display memories with the different update rates. In this manner rendering power could be concentrated on parts of a scene that were changing the most, effectively reducing the virtual reality system's overall rendering load.

A methodology to ease the management of virtual world objects among the different display memories was devised based on observations and analysis of previous priority rendering efforts. This approach, known as region priority rendering, grouped non-dynamic virtual world objects into regions based on spatial locality [Chow et al. 2005a]. Figure 3 shows an example of virtual world regions. By doing this, entire regions of objects could be assigned to the different display memories, based on the user's location in the virtual world, instead of having to compute individual validity estimates, then sort and assign each individual object to a

display memory periodically. Regions closer to the user would be assigned to display memories with higher update rates, while regions further away would be assigned to display memories with lower update rates.

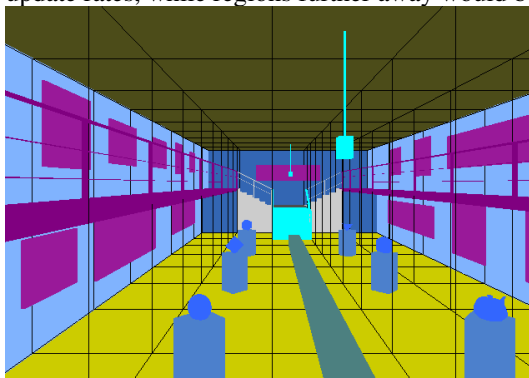


Figure 3: Division of the virtual world into regions.



Figure 4: Magnification of scene tearing artefact.

Further rendering load reductions have also been achieved through the segmenting of large objects for priority rendering. This was because by segmenting a large object, each of the smaller object segments could be treated individually and be rendered at particular update rates, independent of the other object segments. In this respect, different sections of an object could potentially be updated on separate display memories at different update rates, rather than having to update the whole object even if only a small section of that object required updating. Large virtual world objects were segmented along the region boundaries.

The implementation of large object segmentation with region priority rendering for the ARP virtual reality system, introduced a scene tearing/overlapping problem [Chow et al. 2005a]. Scene tearing artefacts resulted in a user perceiving gaps or discrepancies in the virtual world, thus destroying the illusion of reality that the virtual reality system attempted to create. Scene tearing can be seen in figure 4 (tearing appears as white because the frame buffer was originally cleared to this colour), where the closer section of the scene was updated as the user translated through the scene but the further section had yet to be updated.

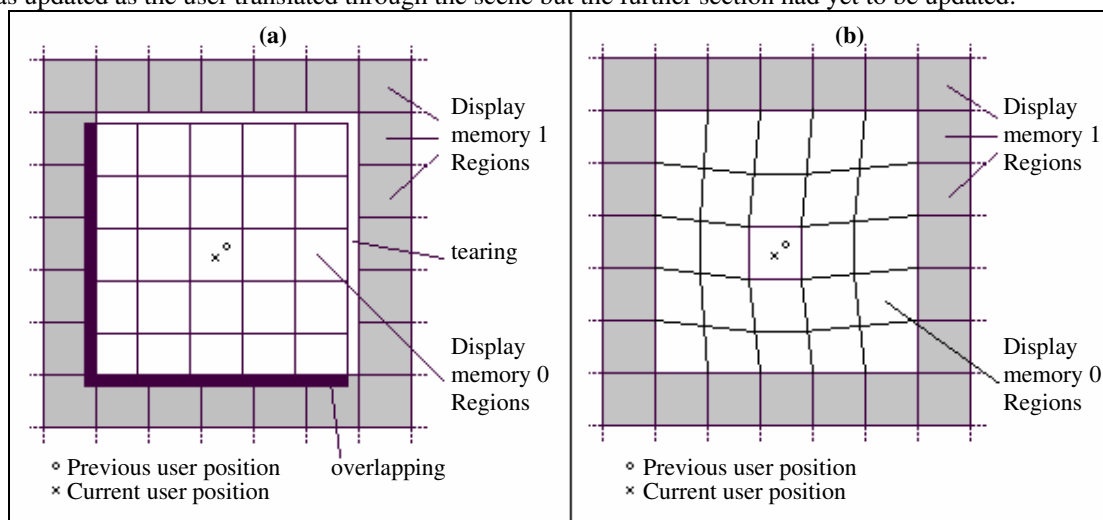


Figure 5: Greatly exaggerated top-down view of (a) tearing/overlapping (b) region warping.

These tearing artefacts would surface when an object's segments were allocated to different display memories and were thus updated at different update rates, whilst the user was translating through the scene. A top-down view of the tearing/overlapping of regions is depicted in figure 5a, where display memory 0 is updated at a higher update rate compared to display memory 1. Even though the user has moved to a new location, display memory 1 has yet to be updated. This meant that when a user translated through the scene, the display memories were updated at different times with the user's viewpoint located at different locations. Therefore the shared polygon vertices (at the edges between display memory 0 and display memory 1) were computed at slightly different locations even though they were the exact same vertices.

Region warping was formulated as a solution to the scene tearing problem. This methodology was based on a surface region warping technique previously suggested for use in computer animation, as a way intended to force vertices on different regions to align [Milliron et al. 2002]. Warping techniques have also been successfully used in many other virtual reality applications, as well as in computer graphics [Mark et al. 1997; Simon et al. 2004]. The effects of region warping can be seen in figure 5b, where the object vertices of the regions located on the display memory with the higher update rate are perturbed in order to force the shared vertices to align. The implementation of this approach effectively hid tearing artefacts caused by the updating of different object segments at different update rates.

3. EXPERIMENTAL METHOD

This section gives a description of the region warping procedure used in the experiments, along with an account of the experiments that were performed.

Region warping was designed to solve the tearing problem which occurred due to different segments of segmented objects being updated at different update rates. This technique essentially involves perturbing vertices across regions for the purpose of aligning the shared vertices. In order to avoid the potential problem of certain objects looking slightly displaced due to the warping, it is necessary to perturb all the vertices in the regions and not just vertices of the segmented objects. This potential problem is portrayed in figure 6. Figure 6a shows the original scene rendered normally. In figure 6b only the floor segment is warped (the rear end of the floor has shifted slightly to the right), resulting in the object looking slightly out of place (highlighted in the ovals). In figure 6c all the vertices are perturbed, the object no longer looks out of place.

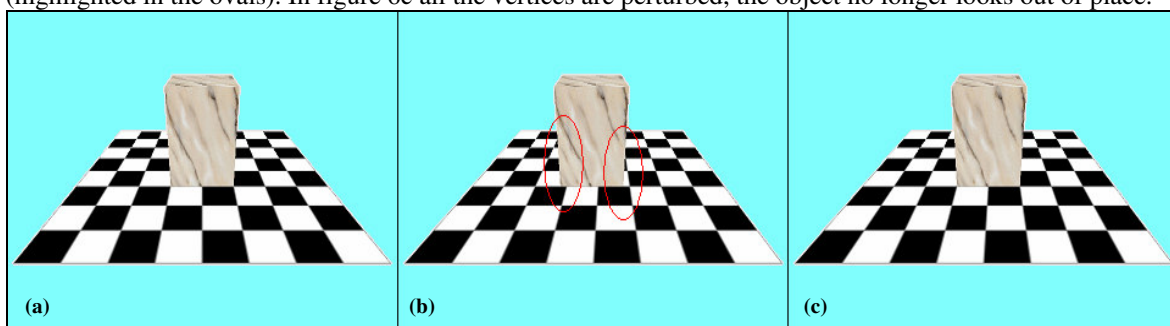


Figure 6: (a) Normal image. (b) Only floor segment warped, object looks out of place. (c) All vertices are perturbed.

Before warping could be done, the amount of perturbation required to force the boundary vertices to align had to be known. This was done by keeping track of the user's previous positions, whenever a particular display memory was last updated. From this, a vector of maximum perturbation could be calculated from the difference between these user positions.

For example (refer to figure 5a), in order for the vertices at the edge of display memory 0 to align with the shared vertices on display memory 1, vertices at the edge of display memory 0 have to be perturbed by the Maximum Perturbation Vector (MPV), which is simply the difference between the previous user position and the current user position. Note that this is a 3-dimensional vector, as figure 5a merely shows the top-down view. Only the vertices located at the edges of display memory 0 have to be perturbed by the MPV, vertices in the region occupied by the user are not perturbed at all, whilst other vertices located in between these edges are interpolated to a value somewhere in between no perturbation and maximum perturbation.

Therefore, normalization of the vertices had to be done in order to compute the exact perturbation required for each vertex, before interpolation could be performed. Normalization was performed using what can be seen as concentric square rings, centered on the circumference of the square base region the user was located in, as shown in figure 7. All vertices located on or inside the circumference of the center, user occupied region, were normalized to 0.0 signifying no perturbation. All vertices along the largest circumference (the edges between display memory 0 and display memory 1), were normalized to 1.0, meaning maximum perturbation. All other vertices were normalized between these values, proportional to their respective locations. For example, vertices on the square ring exactly in the middle between 0.0 and 1.0, was assigned a value of 0.5.

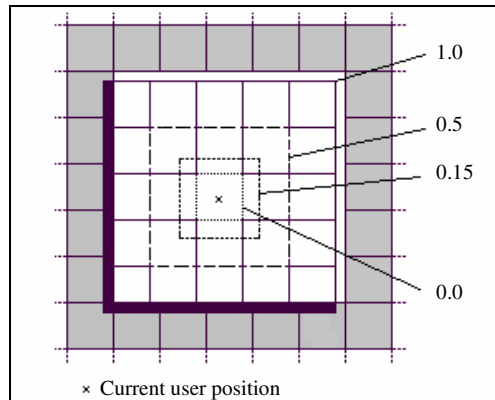


Figure 7: Normalization.

Interpolation of vertex perturbations could then be conducted with these normalized values. Experimental simulations simulating the display of an ARP virtual reality system, used in conjunction with region priority rendering and region warping were performed. The aim of the experiment was to see the distortion effects that would result from the perturbing of region vertices. Therefore simulations on region warping were carried out using two methods of interpolation, namely linear interpolation and squared interpolation. In linear interpolation, the normalized values were multiplied by the MPV to obtain the perturbation vector for a particular vertex. Whereas in squared interpolation, the square of the normalized values was used instead. The choice of these interpolation methods could be seen as evenly spreading the distortions across the regions, for the case of linear interpolation and in the other case, concentrating distortions further away from the user.

By rendering the vertices with the newly perturbed vertex coordinates, entire regions were warped slightly in order to hide the tearing. Figure 5b illustrates a greatly exaggerated top-down view of the vertex perturbation effects. In reality a practical update threshold for priority rendering, or in other words the maximum amount tolerable before an object's display becomes invalid and requires updating, should either be the inter-pixel spacing of display memory or the inter-pixel spacing of the HMD. This would mean that the tearing artefacts, and therefore the amount of perturbations required, should only be in the order of one or two pixels. Nevertheless even at this pixel level, the experiments were intended to ascertain whether there would be any significant differences between the two interpolation approaches.

The virtual world scenario used for the experiments was that of an art gallery. This was in anticipation that a user would perceive more clearly the perturbing of vertices on a rigid, well-structured scene, as oppose to a non-structured scene. Also this indoor scene was designed to be well within the human depth perception range. In order to be able to compare the differences between linear interpolation and squared interpolation, the experiment had to be repeatable. Therefore a fixed path, regarded as the worst case scenario for the tearing from the user's point of view was chosen for the experiments. Experiments were then repeated, with the scene rendered in a number of different ways, namely; normal rendering (without priority rendering), priority rendering with large object segmentation (showing tearing), priority rendering with linearly interpolated region warping and priority rendering with squared interpolation region warping. Individual displayed frames were saved as complete sequences of screenshots for mathematical analysis, as well as for the construction of short video clips used for comparison purposes.

4. RESULTS AND DISCUSSIONS

The Mean Squared Error (MSE) and Peak Signal-to-Noise Ratio (PSNR) error metrics (defined in eq.1 and eq.2 below) were used to analyze the two methods of warping mathematically. These error metrics are often used to quantify the level of distortion in video and image compression techniques. For experiments using the different rendering methods described above, the absolute values of the MSE and PSNR results were not of any interest. However an indication regarding the amount of distortion found between the differently rendered frames could be inferred by comparing these frame-by-frame error metrics values.

$$MSE = \frac{1}{MN} \sum_{y=1}^M \sum_{x=1}^N [I(x, y) - I'(x, y)]^2 \quad (\text{eq.1}) \quad PSNR = 20 \log_{10} \frac{255}{\sqrt{MSE}} \quad (\text{eq.2})$$

where M and N are the dimensions of the frames and I(x,y) is the reference frame.

The MSE and PSNR results were calculated by using the normally rendered experiment frames as the original, undistorted reference frames. Figure 8 shows the results of the MSE for the sequence of frames interpolated using the two different interpolation methods. It can be seen from figure 8 that while the shape of the graphs is roughly similar, the squared interpolation frames have significantly less error as opposed to frames rendered using linear interpolation. The PSNR results also indicated that the squared interpolation method produced frames that had less distortion, and in other words were closer to normal rendering. Table 1 shows a short sequence of PSNR results.

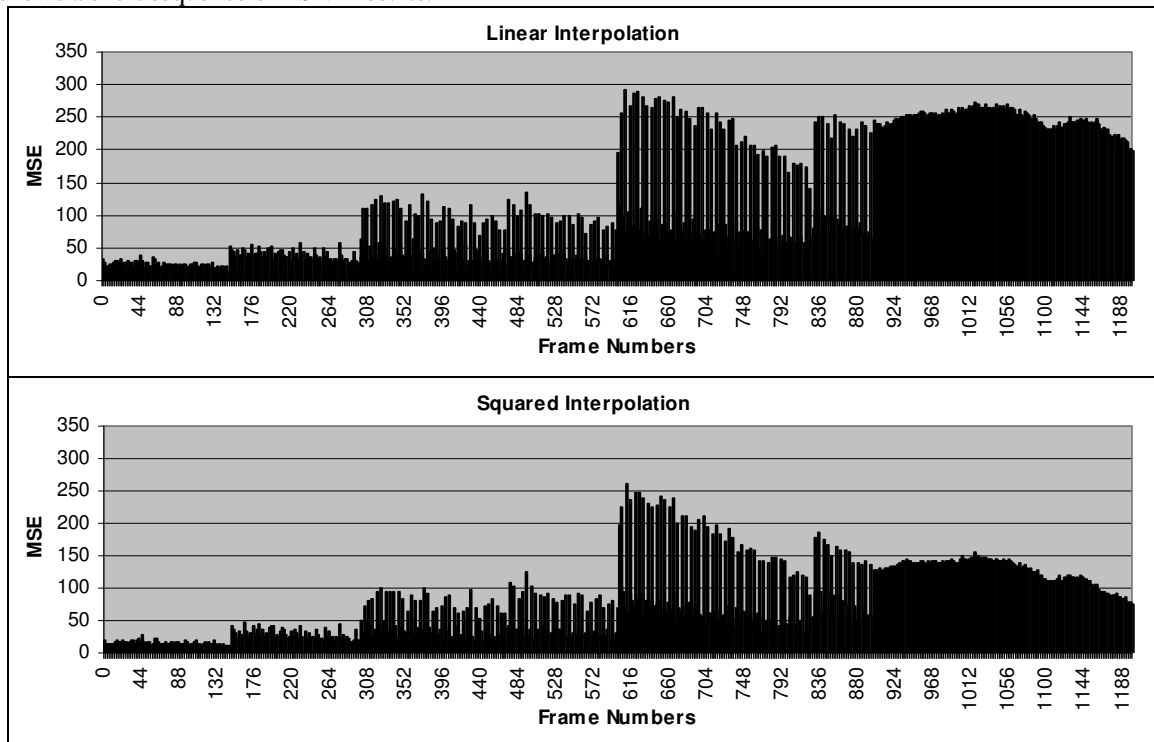


Figure 8: Mean Square Error of the individual frames, for the two methods of interpolation.

Table 1: PSNR for a sequence of frames, comparing linear interpolation and squared interpolation

Frame Numbers	PSNR (dB)	
	Linear Interpolation	Squared Interpolation
890	29.91794	30.44601
891	30.28992	32.71463
892	80.82784	80.82784
893	29.41706	31.64415
894	24.58903	26.82375
895	30.09526	30.47705
896	30.52503	33.17345
897	85.69346	85.69346
898	29.78135	31.54816
899	24.23855	27.04248

Further evidence of this could be seen from observations of error images. Figure 9 shows an error image (grey indicates no difference) between a frame rendered with linear interpolation and the original reference frame. To the unaided human eye, differences between the frames were more or less negligible. However, it

can be seen from the error image that region warping stretches pixels slightly across the scene to compensate for the tearing. Figure 10 is an image of the exact same error frame this time showing differences between normal rendering, linear interpolation and squared interpolation. Grey in figure 10 indicates where all three images were the same, white signifies errors in both interpolation methods from the original, and black shows no errors in squared interpolation but errors in linear interpolation. This shows that sections of the scene closer to the user's viewpoint contain less error in squared interpolation. Another way of looking at this is that the error in the squared interpolation is pushed further back from the user's viewpoint. These observations were consistent with previous mathematical analysis.

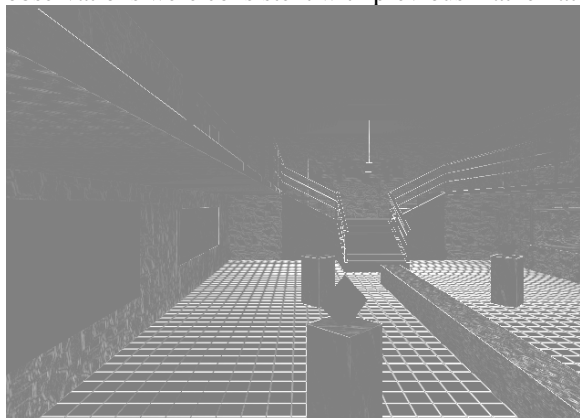


Figure 9: Difference image between linearly interpolated region warping and normally rendered frames.

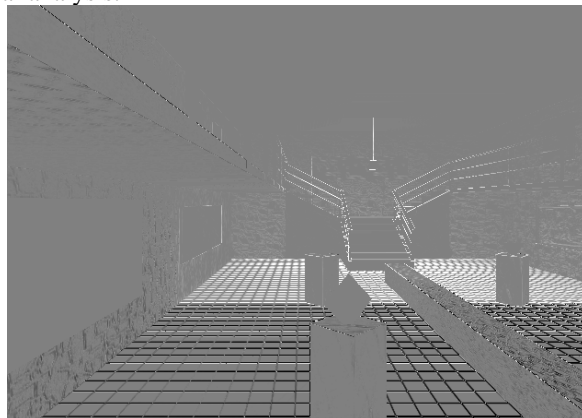


Figure 10: Differences between normal rendering, squared interpolation and linear interpolation region warping.

Figure 4 (previously shown above) is an image of an uncompensated frame, showing clearly the tearing (tearing appears as white because the frame buffer was clear to this color, therefore any discrepancies between the vertices showed up as white). Figure 11 shows the region warping results on the exact same frame as figure 4, tearing is no longer apparent. Magnification of the tearing in figure 4 proved that previous assumptions of the tearing only being one or two pixels wide were indeed correct. In spite of this, the experimental results still gave an indication that by pushing the bulk of errors further away from the user's viewpoint, region warping with squared interpolation resulted in significantly less distorted frames and is therefore the more favorable of the two interpolation methods.



Figure 11: Frame rendered with linear interpolated region warping, tearing is no longer apparent.

However it is unclear as to how these distortions, no matter how small, might affect a user's perception of realism within a virtual environment. In particular, it is important to determine whether these distortions will destroy the illusion of reality in stereoscopic virtual reality. Researchers have observed that even in perfect head tracked stereoscopic systems, a user still perceives virtual object warping or shifting during head movements, and have suggested pre-distortion methods to counteract such distortions [Wartell et al. 1999]. It is therefore conceivable that the small distortions caused by priority rendering with region warping might not appear that differently from normal rendering. Moreover, the distortions due to region warping only occur

when a user's viewpoint is translating through the scene, and the human eye is less sensitive to detail that moves rapidly across the retina [Reddy 2001].

This gives rise to the possibility that a user might not perceive region warping distortions that were concentrated at a certain distance away from the user's vantage point. Similar arguments have been used in computer graphics level of detail (LOD) modulation techniques, which seek to reduce the LOD in a scene without the user perceiving any differences [Reddy 2001]. The experimental results presented in this paper have established that region warping with squared interpolation manages to push distortions further back from a user's perception of detail range. Increasing region sizes might also add to this effect.

5. CONCLUSION AND FUTUREWORK

This paper has described region warping, used to hide scene tearing artefacts, and has presented experimental results comparing two interpolation methods for region warping. Analysis of the experimental results have shown that rendering using the squared interpolation region warping method produces displayed images containing less error from the original ideally rendered images. At the same time, squared interpolation pushes the distortions caused by the warping further away from the user's viewpoint. Given this, squared interpolation was therefore deemed to be the more attractive of the two region warping interpolation methods.

These results have thus provided a framework for further experiments dealing with human perception to region warping. These future experiments will require involving a number of human subjects and will focus on determining an acceptable distortion threshold for region warping, from a human perception point of view, issues faced in designing human visual perception experiments are outlined in Chow et al. [2005b].

6. ACKNOWLEDGEMENTS

Textures used in the scene were taken from Paul Bourke's texture library at: (<http://astronomy.swin.edu.au/~pbourke/texture/>).

REFERENCES

- Chow, Y.W., Pose, R. and Regan, M., 2005a. Large Object Segmentation with Region Priority Rendering. *Accepted to The 28th Australasian Computer Science Conference 2005 (ACSC '05)*, Newcastle, NSW, Australia.
- Chow, Y.W., Pose, R. and Regan, M., 2005b. Design Issues in Human Visual Perception Experiments on Region Warping. *Submitted to IADIS International Conference on Applied Computing 2005*.
- Cook, R.L., 1986. Stochastic Sampling in Computer Graphics. *ACM Transactions on Graphics*, Vol. 5, No. 1, pp. 51-72.
- Mark, W., Bishop, G. and McMillan, L., 1997. Post-Rendering 3D Warping. *Proceedings of 1997 Symposium on Interactive 3D Graphics*, Providence, Rhode Island, pp. 7-16.
- Milliron, T., Jensen, R.J., Barzel, R. and Finkelstein, A., 2002. A Framework for Geometric Warps and Deformations. *ACM Transactions on Graphics*, Vol. 21, No. 1, pp. 20-51.
- Naiman, A.C., 1998. Jagged Edges: When is Filtering Needed? *ACM Transactions on Graphics*, Vol. 21, No. 1, pp. 238-258.
- Reddy, M., 2001. Perceptually Optimized 3D Graphics. *IEEE Computer Graphics and Applications*, Vol. 21, No. 5, pp. 68-75.
- Regan, M. and Pose, R., 1993. An Interactive Graphics Display Architecture. *Proceedings of the Virtual Reality Annual International Symposium (IEEE VRAIS '93)*, Seattle, United States, pp. 293-299.
- Regan, M. and Pose, R., 1994. Priority Rendering with a Virtual Reality Address Recalculation Pipeline. *Proceedings of SIGGRAPH '94, in Computer Graphics, Annual Conference Series*, pp. 155-162.
- Simon, A., Smith, R.C. and Pawlicki, R.R., 2004. OmniStereo for Panoramic Virtual Environment Display Systems. *Proceedings of IEEE Virtual Reality (IEEE VR '04)*, Chicago, Illinois, pp. 67-73.
- Wartell, Z., Hodges, L.F., and Ribarsky, W., 1999. Balancing Fusion, Image Depth and Distortion in Stereoscopic Head-Tracked Displays. *Proceedings of SIGGRAPH '99, in Computer Graphics, Annual Conference Series*, pp. 351-358.