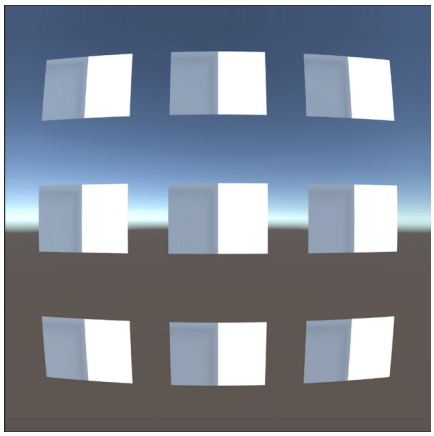
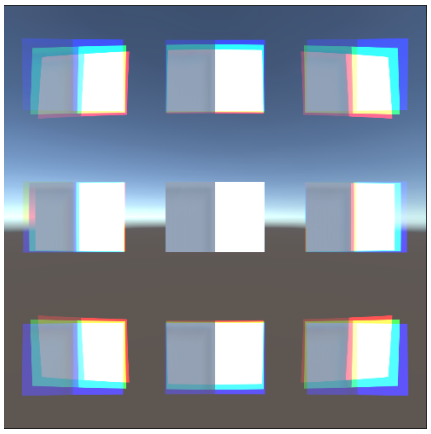
Virtual Reality – Research Report

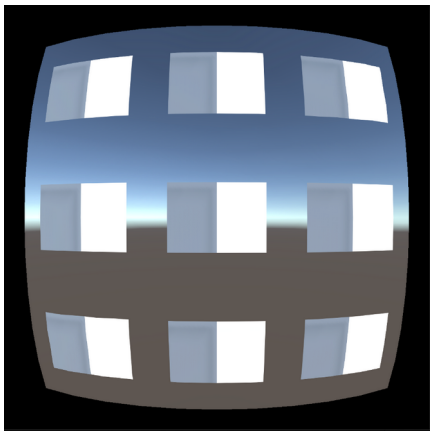
1. Static Renders



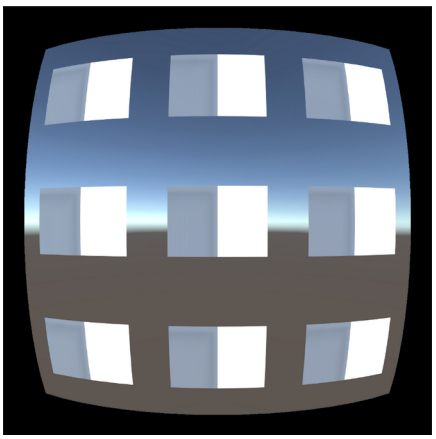
(a)



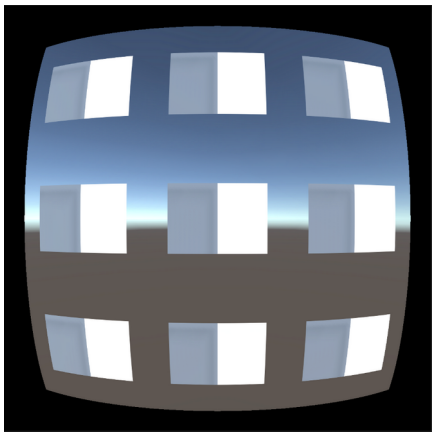
(b)



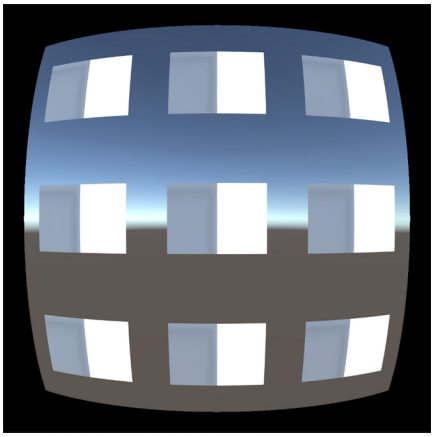
(c)



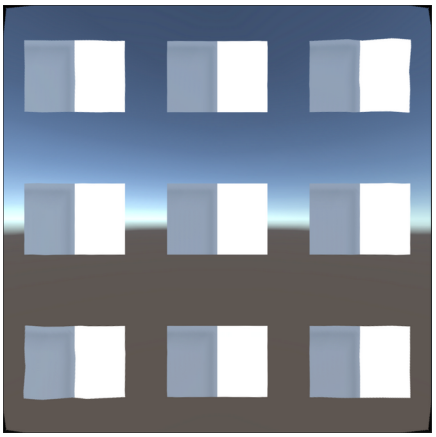
(d)



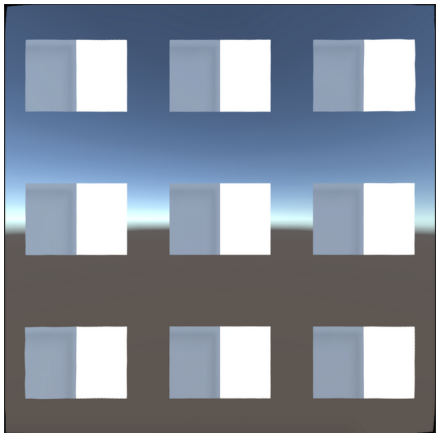
(e)



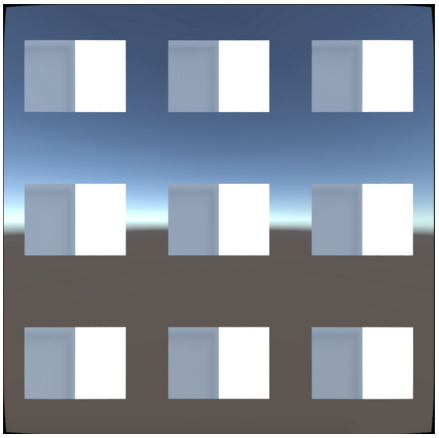
(f)



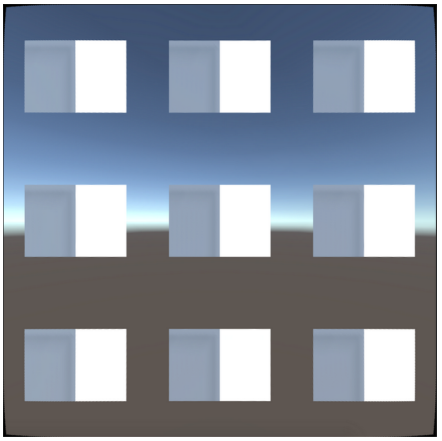
(g)



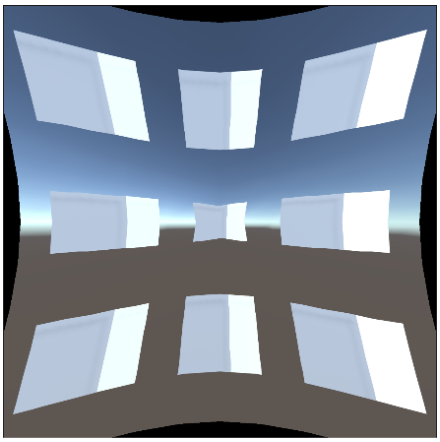
(h)



(i)



(j)



(k)

(a) Pre-correction using fragment shader
(b) Radial chromatic aberration correction
(c) Pre-correction using vertex shader – 10 vertices
(d) Pre-correction using vertex shader – 64 vertices
(e) Pre-correction using vertex shader – 100 vertices
(f) Pre-correction using vertex shader – 512 vertices
(g) Simulated distortion after correction using vertex shader – 10 vertices
(h) Simulated distortion after correction using vertex shader – 64 vertices
(i) Simulated distortion after correction using vertex shader – 100 vertices
(j) Simulated distortion after correction using vertex shader – 512 vertices
(k) Vertex shader using only the inverse given by LaValle – 10 vertices

2. Pixel Vs Mesh based Distortion Correction

In pixel-based methods, computations and transformations are performed in the fragment shader at a pixel level. This means that with a large enough render texture and quad size errors are difficult to see, only presenting via aliasing. However, functioning at such detail comes at a cost. Given that every transformation is unique within the Cartesian coordinate system, it is required that calculations are performed for each pixel. Render textures of size 1,200 x 1,200 pixels are used for all images seen above. This means both 1,200 x 1,200 = 1,440,000 sets of calculations and texture lookups must be performed per frame in the case of pixel-based methods.

For mesh-based methods, however, computations and transformations are performed within the vertex shader. Therefore, transformation can be seen to be only unique to each fragment rather than each pixel. While this leaves the number of texture lookups unchanged, it has a significant impact on the number of computations required, only being performed once per vertex. Figures (c) to (f), use 10, 64, 100 and 512 vertices respectively, meaning almost 1,000 times less computations than what is required for pixel-based methods. However, using too few vertices can then lead to unwanted visual effects and reduced quality as can be seen in Figures (c), (d), (g) and (h) - most notably in the corner cubes. This is due to straight edges being preserved within fragments themselves. Therefore, curved edges are approximated poorly if too few vertices are used. For the given scene, once at 100 vertices such issues are seen to disappear. Despite extensive efforts to profile each shader exactly using Xcode frame capture on a mac mini and Intel Graphics Performance Analyser on a windows laptop, both were unsuccessful for different reasons. Instead the Unity Profiler was used alone to give average GPU render times of 0.12 ms, 0.13 ms, 0.13 ms and 0.15 ms for the 10, 64, 100 and 512 vertex based methods respectively. This correlation between the number of vertices and the render time was expected due to the increased number of computations. However, given the significant amount of variance seen for each measurement and the small difference in averages, the impact is largely negligible and likely not of significant concern when compared to pixel-based methods.

3. Eye Tracking

Equations used for distortion correction work on the assumption that the eye is looking directly at the centre of the screen. Therefore, distortion pre-correction originates from the centre of the image and works outwards. However, this is rarely true with eyes frequently focussing at different points around the screen. This means that the the optical paths used to view the screen are changed, adding additional asymmetric distortion. [1] find that such distortion can lead to errors spanning up to 50 pixels. To correct for this requires first being able to track the positioning of a users eye. In combination with this, Martschinke et al. model the image as a 3x3 grid and calculate the required correction distortions at each of the 9 locations. Depending on where the eye is focussed they are

then able to bi-linearly interpolate to get a distortion that approximates that which is required. This is seen to reduce error to an average of 6 pixels.

In addition to this, eye tracking has also shown to be beneficial when used for foveated rendering. This can be achieved in several ways, one method relies on sampling maps to divide the image into 10 concentric regions [2]. Each of these bands renders at a different resolution, the highest at the centre and gradually decreasing outwards. Eye tracking is then used to specify the centre around which these concentric circles should originate. As a result, only the region that the user is looking at is rendered at full resolution. While most obviously applicable for pixel-based methods, foveated rendering can also be used for the render textures used in mesh based methods. In addition to this, mesh based methods could also be modified such that vertex density varies in the same way as resolution. This would keep the vertex count low while also ensuring no poor visual quality in the regions where the user is looking.

Although good in theory, both of these methods are highly dependent on the latency present in eye tracking techniques. In the case that such tracking is too slow, this could lead to an increase in visual artefacts and reduced quality. [3] measure such latency for a variety of commercially available headsets. Most notably they find the Fove-0 and Vive Pro Eye to have latencies of 15 ms and 50 ms respectively. Although the human eye functions at a maximum fps of 60, [4] and many others suggest using an fps of at least 90 to avoid inducing motion sickness. This requires frames to refresh every 11 ms, not enough time for eye tracking to take place in either of the devices mentioned. Although the Fove-0 makes use of a fps of 70 to account for this, arguably most commercial headsets do not perform eye tracking fast enough to avoid certain negative effects.

4. Frame Resolution

The compression of parts of a render texture into a smaller real estate is the equivalent to a large render texture being mapped to a smaller surface. The latter is a well known problem and results in aliasing due to sampling taking place at a frequency too low compared to that of the image itself. As a result, jagged textures are seen within the output image. This is solved using anti-aliasing techniques the most common being MSAA, based on the principles of super-sampling. Within this the image is first rendered to a higher resolution, values are then calculated for a lower resolution image using this. Each pixel in the lower resolution image can be thought to have multiple corresponding pixel values in the higher resolution image. An average of these are taken and the output is used for the pixel value of the smaller image. In the case of distortion correction the latter could be applied by: re-rendering the texture to a higher resolution, applying the distortion and then outputting to a render texture the same size as the original (according to MSAA). On the other hand, similarly to that in section 3, concentric bands of the image could be rendered to lower resolutions prior to distortion correction taking place. The fixed degree of distortion from the lens would mean that suitable bands could be easily precomputed.

For these same reasons, using a rendering resolution higher than that of the display resolution would result in aliasing and poor image quality. As a result, rendering resolution must be less than or equal to that of the device resolution. Given this, along with the extremely small distance between the users eyes and the display, if the devices hardware resolution is too low, the image will appear blurry and lacking in detail. To ensure this is not the case requires for modern headsets to have displays with extremely high resolutions, such that even frames rendered at slightly lower resolutions appear sharp. This results in VR headsets suffering from very high power consumption along with the transmission of data over cables being problematic [5].

5. Equation modifications

The inverse equation given by LaValle [6] was seen only to lead to distortion being possible at a limited number of coefficient values and gave poor quality, see Figure (k), especially when used in lens simulation. Therefore, the equation given in [6] was instead used similarly to as in [7], being modified to work in polar coordinates:

$$f^{-1}(r) \approx r - r\left(\frac{c_1 r^2 + c_2 r^4 + c_1^2 r^4 + c_2^2 r^8 + 2 c_1 c_2 r^6}{1 + 4 c_1 r^2 + 6 c_2 r^4}\right)$$

References

[1] *J. Martschinke, J. Martschinke, M. Stamminger and F. Bauer, "Gaze-Dependent Distortion Correction for Thick Lenses in HMDs," 2019 IEEE Conference on Virtual Reality and 3D User Interfaces (VR), 2019, pp. 1848-1851, doi: 10.1109/VR.2019.8798107.*

[2] *D. Pohl, T. Bolkart, S. Nickels and O. Grau, "Using astigmatism in wide angle HMDs to improve rendering," 2015 IEEE Virtual Reality (VR), 2015, pp. 263-264, doi: 10.1109/VR.2015.7223396.*

[3] *S. Niklas, N. Diederick, W. Tamara, S. Frank, R. Katharina, W. Siegfried, L. Markus, “A Comparison of Eye Tracking Latencies Among Several Commercial Head-Mounted Displays.” i-Perception, 2021, 12. 204166952098333. 10.1177/2041669520983338.*

[4] *"Guidelines for VR Performance Optimization | Oculus Developers", Developer.oculus.com. [Online]. Available: https://developer.oculus.com/documentation/native/pc/dg-performance-guidelines/. [Accessed: 19- May- 2021].*

[5] *G. Denes, K. Maruszczyk, G. Ash and R. K. Mantiuk, "Temporal Resolution Multiplexing: Exploiting the limitations of spatio-temporal vision for more efficient VR rendering," in IEEE Transactions on Visualization and Computer Graphics, vol. 25, no. 5, pp. 2072-2082, May 2019, doi: 10.1109/TVCG.2019.2898741.*

[6] *S. M. LaValle, Virtual Reality, 1st ed. Cambridge University Press, pp. 198-200.*

[7] *J. Mallon and P. F. Whelan, "Precise radial un-distortion of images," Proceedings of the 17th International Conference on Pattern Recognition, 2004. ICPR 2004., 2004, pp. 18-21 Vol.1, doi: 10.1109/ICPR.2004.1333995.*