Towards Perceptually Lossless Rendering: Latency Aware Foveated Rendering in Unreal Engine 4

Nicholas T. Swafford [1]
Darren Cosker [1]
Kenny Mitchell [2]

[1] Centre for Digital Entertainment
University of Bath

[2] Disney Research Zurich

Introduction

The human visual system is often assumed to be perfect despite limitations arising from a variety of different complexities and phenomena. Yet real-time rendering still operates on the assumption that a single render will be fully appreciated at any single point in time. With the increasing use of 4K-8K UHD displays and the push towards higher pixel densities for head-mounted displays, the industry is pressured to meet market demands for intensive real-time rendering. The adoption of perceptually lossless rendering methods has never seemed more appropriate.

A review of prior psychophysical and perceptual rendering literature suggests that perceptually lossless rendering, in which lossy renders are indistinguishable from their non-degraded counterparts, is a definite possibility. Typically, perceptual rendering methods employ foveation, in which the region corresponding to the central field of view is rendered with higher fidelity than the region corresponding to the periphery. In this study, we contribute a straightforward foveated rendering method in Unreal Engine 4 (UE4) in order to gather insights on what may be holding back the adoption of these methods at a commercial level. Additionally, we contribute a conservative metric for calculating the foveal region size in the presence of system latency.

Background

It is well established that peripheral vision is significantly worse than foveal vision in many ways, and that these differences cannot be explained solely by a loss of acuity [5]. However, acuity sensitivity still forms a significant portion of peripheral detail loss and can be one of the easiest phenomena to exploit.

Parkhurst and Niebur [4] previously implemented foveated geometric complexity decimation on a now decade old version of the Unreal Engine, but their study focused on task performance under aggressive foveation. Several of their subjects in their gaze-contingent experiments with a 60Hz eye tracker reported some form of visual anomaly when prompted about the fact. More recently, Guenter et al. [2] implemented a foveated rendering method with spatial and temporal property variation in which, at a certain level-of-detail (LOD), users reported foveated renders to be of equal or higher quality than non-foveated counterparts. They also noted the importance of system latency, as high latency systems produced a "pop" effect caused by the foveal region catching up to the gaze point.

Latency Aware Foveated Rendering

Our internal experimentation confirms the aforementioned latency criticality of perceptually lossless rendering. To compensate, the foveal region diameter can be increased such that the true foveal field of view is always contained within the rendered foveal region, for some estimated maximum saccadic speed (roughly $200\,^{\circ}\,s^{-1}$) [1] and overall system latency. In this way we avoid the "pop" effect described by Guenter et al. at the expense of computational gain. We propose Eq.1 to calculate the adjusted foveal diameter:

$$F_{\varnothing} = 2\,\rho_{pixel}\,d_u \tan\left(L_{tot} S_{max} + \frac{\alpha}{2}\right) + 2\,b_w + c \qquad (1)$$

Where L_{tot} is the worst-case total tracking latency in milliseconds, S_{max} is the maximum saccadic speed in $radians/ms$, d_u is the user's distance from the screen, α is the angle subtended by the fovea which is roughly 5° [3], b_w is the width of blending border, and ρ_{pixel} is the pixel density of the screen in $pixels/mm$.

The error constant c is added due to some simplifications. We assume the user remains at a constant distance from the screen between each tracking frame. As the user's maximal positioning speed is unknown, we either employ a conservative position prediction model or reduce the total tracking latency until the difference of distances is near zero. We also avoid calculating the change in radius when gaze is not perpendicular to the display surface, as this depends on the angle of incidence and the curvature of the display. Lastly, we assume the tracker's precision and accuracy errors are negligible.

Our foveated rendering method is a pragmatic, initial technique to be employed in perpetually lossless rendering. The peripheral render is rendered at a quarter of the intended display resolution and then upscaled with minor blurring. The foveal render, with diameter calculated by Eq.1, is layered on and blended against the peripheral layer (vignette mask) at the gaze point. We used a Tobii EyeX commercial eye tracker, an informed group of 4 subjects, a Radeon 290X GPU, and two scenes (simple and complex) for our study. We found that a foveal diameter corresponding roughly to 1000 pixels within a 4K UHD render of a simple scene, displayed on a 28" monitor at a typical viewing distance of 20", was sufficient to reasonably compensate for total system latency representing a saving of approximately 6.8 MP. Note that with a near-zero total latency, the foveal window would only have to be approximately a third of that diameter. There are no official statistics on the overall latency of the tracker, but as the simple scene ran at a relatively high frame rate (approximately 60 FPS) the tracker is the most likely source of total system latency.

Unfortunately, the total system latency when rendering at 4K UHD for our complex scene was too high to avoid the "pop" effect with a reasonably sized foveal diameter. Using the foveal diameter derived from our simple scene, our complex scene ran at approximately 24.3 FPS with foveation on compared to approximately 14.0 FPS with foveation off, an average saving of 20 ms per frame.

Figure 1: Section of a 1080p render of our complex scene, the Elemental tech demo (courtesy of Epic Games), showing simple foveation. Although the difference in quality between both regions is apparent here, it was not noticeable during experimental observations.

In conclusion, we believe that there is a renewed commercial push to find new ways to optimize real-time rendering, and that hardware demands have reached a critical point that necessitates the adoption of perceptually based rendering methods. We have contributed a pragmatic foveated method in a commercial game engine that exploits a single aspect of the human visual system, the loss of acuity in the peripheral field, and provides a substantial performance boost. Our findings indicate that current commercial eye tracking devices operate with unsatisfactorily high latency impeding the wider adoption of gaze-contingent methods. We also present a rule-of-thumb formula to adjust foveated rendering parameters to compensate for low-quality tracking. We would like to thank the Engineering and Physical Sciences Research Council for their help funding this research and Epic Games for making UE4 open source.

[1] Richard A Abrams, David E Meyer, and Sylvan Kornblum. Speed and accuracy of saccadic eye movements: characteristics of impulse variability in the oculomotor system. *Journal of Experimental Psychology: Human Perception and Performance*, 15(3):529, 1989.

[2] Brian Guenter, Mark Finch, Steven Drucker, Desney Tan, and John Snyder. Foveated 3d graphics. *ACM Transactions on Graphics (TOG)*, 31(6):164, 2012.

[3] Gustav Osterberg. *Topography of the layer of rods and cones in the human retina*. Nyt Nordisk Forlag, 1935.

[4] Derrick Parkhurst and Ernst Niebur. A feasibility test for perceptually adaptive level of detail rendering on desktop systems. In *Proceedings of the 1st Symposium on Applied perception in graphics and visualization*, pages 49–56. ACM, 2004.

[5] Hans Strasburger, Ingo Rentschler, and Martin Jüttner. Peripheral vision and pattern recognition: A review. *Journal of Vision*, 11(5):13, 2011.