Foveated 3D Graphics

Brian Guenter Mark Finch Steven Drucker Desney Tan John Snyder
Microsoft Research

Abstract

We exploit the falloff of acuity in the visual periphery to accelerate graphics computation by a factor of 5-6 on a desktop HD display (1920×1080). Our method tracks the user's gaze point and renders three image layers around it at progressively higher angular size but lower sampling rate. The three layers are then magnified to display resolution and smoothly composited. We develop a general and efficient antialiasing algorithm easily retrofitted into existing graphics code to minimize "twinkling" artifacts in the lower-resolution layers. A standard psychophysical model for acuity falloff assumes that minimum detectable angular size increases linearly as a function of eccentricity. Given the slope characterizing this falloff, we automatically compute layer sizes and sampling rates. The result looks like a full-resolution image but reduces the number of pixels shaded by a factor of 10-15.

We performed a user study to validate these results. It identifies two levels of foveation quality: a more conservative one in which users reported foveated rendering quality as equivalent to or better than non-foveated when directly shown both, and a more aggressive one in which users were unable to correctly label as increasing or decreasing a short quality progression relative to a high-quality foveated reference. Based on this user study, we obtain a slope value for the model of 1.32-1.65 arc minutes per degree of eccentricity. This allows us to predict two future advantages of foveated rendering: (1) bigger savings with larger, sharper displays than exist currently (e.g. 100 times speedup at a field of view of 70° and resolution matching foveal acuity), and (2) a roughly linear (rather than quadratic or worse) increase in rendering cost with increasing display field of view, for planar displays at a constant sharpness.

Keywords: antialiasing, eccentricity, minimum angle of resolution (MAR), multiresolution gaze-contingent display (MGCD).

Links: DL PDF WEB VIDEO

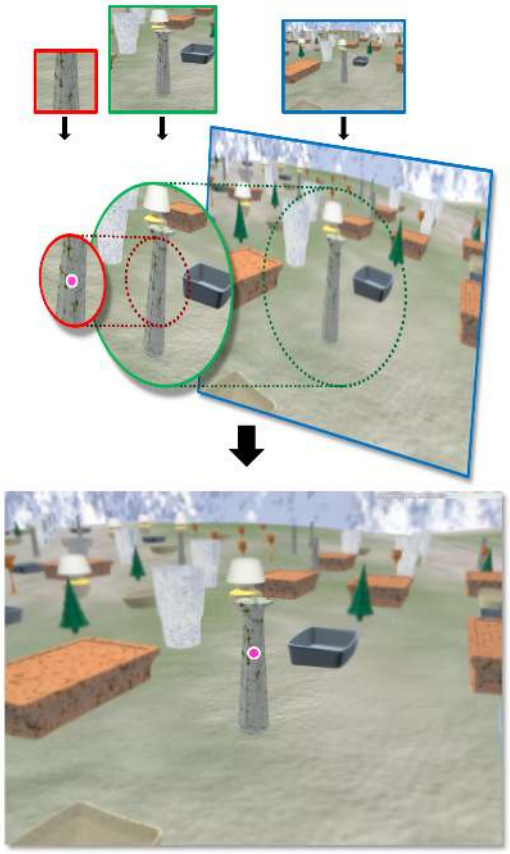


Figure 1: *Foveated rendering. We render three eccentricity layers (red border = inner layer, green = middle layer, blue = outer layer) around the tracked gaze point (pink dot), shown at their correct relative sizes in the top row. These are interpolated to native display resolution and smoothly composited to yield the final image at the bottom. Foveated rendering greatly reduces the number of pixels shaded and overall graphics computation.*

1 Introduction

We see 135° vertically and 160° horizontally, but sense fine detail only within a 5° central circle. This tiny portion of the visual field projects to the retinal region called the fovea, tightly packed with color cone receptors.[1] The angular distance away from the central gaze direction is called *eccentricity*. Acuity falls off rapidly as eccentricity increases due to reduced receptor and ganglion density in the retina, reduced optical nerve "bandwidth", and reduced "processing" devoted to the periphery in the visual cortex. A commonly-used psychophysical model, first discovered by Aubert and Foerster in 1857, asserts that the minimum discernible angular size (the reciprocal of visual acuity) increases roughly linearly with eccentricity. This model accurately predicts performance on many low-level vision tasks [Strasburger et al. 2011]. We use the term *foveation* as a shorthand for the decrease in acuity with eccentricity in the human visual system.

Current CG practice ignores user gaze and renders a high-resolution image over the whole display. This is tremendously wasteful of power and computing resources. The 5° foveal region fills a mere 0.8% of the solid angle of a 60° display. By tracking eye gaze and adapting image resolution and geometric level of detail (LOD) to eccentricity, we can omit unperceived detail and draw far fewer pixels and triangles.

Our system exploits foveation on existing graphics hardware by rendering three nested and overlapping render targets or *eccentricity layers* centered around the current gaze point. Refer to Figure 1. These layers are denoted the *inner/foveal layer*, *middle layer*,

[1] A smaller region of 1° diameter, called the foveola, is often considered the site of foveal vision.

and *outer layer*. The inner layer is smallest in angular diameter and rendered at the highest resolution (native display) and the finest LOD. The two peripheral layers cover a progressively larger angular diameter but are rendered at progressively lower resolution and coarser LOD. These outer layers are also updated at half the temporal rate of the inner layer. We then interpolate these layers up to native display resolution and smoothly blend between them.

Exploiting foveation in CG is an old idea [Levoy and Whitaker 1989; Ohshima et al. 1996; Luebke et al. 2000]. Successfully realizing a performance benefit requires managing two difficulties: aliasing in the periphery and system latency.

Hardware-accelerated 3D rendering relies on point sampling to rasterize polygons and shade pixels. Our system reduces sampling density in the peripheral layers, making antialiasing harder and leading to distracting strobing and crawling artifacts. Brute force supersampling back to native resolution reduces these artifacts but also eliminates the computational advantage of foveated rendering. We solve this problem by amortizing the cost of antialiasing for peripheral layers over multiple frames, using a combined approach of multisample antialiasing (MSAA), temporal reverse reprojection [Guenter 1994; Nehab et al. 2007], and temporal jitter of the spatial sampling grid [Cook et al. 1984].

System latency is the time elapsed between capturing the eye gaze position and displaying the corresponding foveated image. Too much latency makes visible the foveal region update from low to high resolution, causing a "pop" that destroys the illusion of a seamless display. Commodity graphics subsystems, especially eye trackers, that limit latency enough to enable significant savings from foveation have only recently become available. An earlier version of our system used a 60Hz monitor and the Tobii X50 eye tracker sampling at 50Hz with 35ms latency. Lagging foveal layer update on this system generated an obvious and unacceptable "pop". We obtain good results with our current system, which uses a 120Hz monitor and the Tobii TX300 with 300Hz update and 10ms latency.

By properly antialiasing, controlling latency, and choosing the right size and resolution for eccentricity layers, our system lets users perceive a seamless image of uniformly high resolution at a fraction of the cost of full-resolution rendering: 1/5 to 1/6 in our experiments.

This paper makes several contributions. First, we apply multiresolution gaze-contingent rendering to general, interactive 3D graphics and for the first time demonstrate a significant performance advantage. We exploit peripheral region subsampling but avoid artifacts using a fast yet effective antialiasing strategy. Other work [Ohshima et al. 1996; Luebke et al. 2000; Reddy 2001; Murphy and Duchowski 2001] adapts geometric LOD to eccentricity but not rendering resolution, and so cannot accelerate applications bottlenecked by per-pixel shading cost. Our use of a linear model for the acuity eccentricity function allows us to select size and sampling rate for each eccentricity layer using a precomputed optimization, based on a single slope value that we find through user study.

Second, we carefully analyze asynchronous communication between our system components (eye tracking hardware, computer, graphics card, and display) to determine and document overall latency. Latency critically affects how foveated rendering is perceived and how much it can save.

Third, we experimentally verify through user studies that our foveated rendering method avoids objectionable artifacts and achieves quality comparable to non-foveated rendering. The study also reveals what slope for our linear model produces the illusion of a "high-resolution everywhere" display, and lets us estimate performance on systems with larger, sharper displays where foveated rendering saves even more.

2 Previous Work

Much previous work has examined the role of perception in image synthesis and applications of visual difference prediction (e.g., [Myszkowski 2002; Ramanarayanan et al. 2007]). We concentrate here on methods that specifically exploit foveation; i.e., acuity falloff in the visual periphery.

Some related work exploits foveation without eye tracking by assuming the user looks at the center of the screen [Funkhouser and Sequin 1993] or by using a content-based model of visual attention [Horvitz and Lengyel 1997; Yee et al. 2001]. Such models have statistical validity across time and different users, but cannot reliably predict where a user will look at every instant.

Eye tracking hardware and foveated displays have existed for at least four decades and much previous work has been published. Survey papers cover eye tracking applications [Duchowski 2002], eye tracking in 3D graphics [O'Sullivan et al. 2002], applications of multiresolution gaze-contingent displays [Reingold et al. 2003; Duchowski and Çöltekin 2007], and attention in display design [Baudisch et al. 2003].

Several papers have studied how degrading peripheral resolution affects visual search performance. [Watson et al. 1997] showed that peripheral resolution could be reduced by almost half without significantly degrading performance. [Watson et al. 2004] examined the effect of peripheral blurring at high eccentricity (20-30°) and found contrast a better indicator of search performance than size. Though this work indicates that supra-threshold acuity degradation can reduce search performance, the effect is complex and task-dependent. We believe this study operates well above the aliased zone in our Figure 2 where details should be preserved.[2]

Our approach most resembles the foveated volume rendering method in [Levoy and Whitaker 1989]. That work ray traces a volume dataset over three image regions sampled at pixel distances of 1, 1/2, and 1/4 of native resolution. It prefilters the (static) 3D volume using a 3D MIPMAP, taking fewer volume samples along more peripheral rays. Our method targets general 3D graphics[3] in which shading, silhouettes, and occlusions are harder to prefilter. We also generalize the method for adapting resolution to peripheral acuity. Most fundamentally, our work matures the ideas proposed in this early work by conducting user studies to confirm the quality of foveated rendering and by measuring a significant speedup. Based on our experience, effective foveation requires fast eye tracking and rendering, much faster than the 100-150ms latency Levoy and Whitaker estimated for their system. This accounts for its preliminary finding that "Users report [...] a perceptible delay in saccading mode. Users are generally aware of the variable-resolution structure of the image."

[Murphy and Duchowski 2007] propose gaze-contingent rendering based on ray tracing. A precomputed intermediate mesh specifies ray locations whose sampling density conforms to acuity falloff; sampling is further increased near object silhouettes. The method targets geometric LOD and ignores the important issue of shading antialiasing. It was also unable to accelerate rendering speed. We note that silhouette hyperacuity prevents our system from dramatically reducing geometric detail in the periphery. Efficiently retaining silhouette detail is an important topic for further research.

[2] The most extreme peripheral blurring we apply preserves much higher frequencies (3.5-7 cycles per degree in the outer layer) than was found in these tests to maximize performance (0.2 cycles per degree).

[3] Our method supports any content that can be rendered on 3D graphics hardware, including shaded meshes and 3D volumes. Its use of reverse reprojection requires the ability to project 3D points from their position in the current frame to their position in a previous one.

Foveation in 2D Applications Our eccentricity layers are similar to the concept of *multiresolution gaze-contingent display* (MGCD) proposed in previous work for medical/psychophysical science and video compression [Loschky and McConkie 2000; Geisler and Perry 2002; Reingold et al. 2003; Duchowski and Çöltekin 2007]. Unlike ours, this work assumes that high-resolution imagery is available to be pyramidally decimated and so could not yield any performance gain for real-time 3D graphics.

Foveated Geometric Detail Elision Geometric detail elision in the periphery has been proposed by many researchers [Cheng 2003; Luebke et al. 2000; Murphy and Duchowski 2001; Ohshima et al. 1996; Reddy 1998; Reddy 2001; Weaver 2007]. We use a simple technique for geometric LOD management described in Section 6, but another could be substituted. In fact, any LOD scheme works without change in our approach if it explicitly considers the pixel spacing of the render target. This simple method automatically ties LOD to eccentricity.

3 Human Visual Acuity in the Periphery

A large research literature documents how visual acuity falls off in the periphery. [Strasburger et al. 2011] provides an excellent survey and historical summary. When measured in terms of a *minimum angle of resolution* (MAR), rather than its reciprocal, *acuity*, a linear model matches both anatomical data (e.g. receptor density), as well as performance results on many low-level vision tasks [Weymouth 1958; Strasburger et al. 2011]. Acuity models form the basis for the well-known theory of *cortical magnification* or *M-scaling,* which posits that enlarging a visual stimulus minimizes performance variation as it becomes increasingly peripheral. A linear model is also typically used in this theory, based on the reciprocal of visual cortex tissue volume devoted to processing each slice of eccentricity.

Given a fixed contrast ratio supported by a particular display, we therefore assume the linear model:

$$\omega = m\, e + \omega_0, \tag{1}$$

where ω is the MAR in degrees per cycle, e is the eccentricity angle, ω_0 is smallest resolvable angle and represents the reciprocal of visual acuity at the fovea ($e = 0$), and m is the MAR slope. This model yields the expected hyperbolic falloff of acuity with eccentricity, where acuity is measured as a minimum resolvable angular frequency; i.e., $f = 1/\omega$. It is illustrated in Figure 2.

Equivalent models have been used before [Luebke et al. 2000; Murphy and Duchowski 2007], as well as models with an additional cubic term [Virsu and Romano 1979; Reddy 2001]. A simple linear model works well for "central" vision (angular radius <8°), after which MAR rises (i.e., acuity declines) more steeply. We assume a symmetric radial falloff ignoring differences between the horizontal and vertical facial axes. We also ignore retinal velocity effects.

In the fovea, receptor density is matched to the eye's optics. In the periphery, receptors become increasingly sparse relative to the eye's optical system Nyquist limit. As observed in [Thibos 1989], there is a "zone of aliasing" in peripheral vision where small angular sizes (high frequencies) are irregularly detectable but not uniformly resolvable. We seek a falloff slope m within this zone that provides a comfortable balance between suppressing aliasing and preserving visible but aliased frequencies in the periphery.[4] Targeting a higher

[4]Different users exhibit different sensitivity to the peripheral blurring that results when aliasing is more completely suppressed to the upper limit. Different graphical content and visual tasks also influence what may constitute an optimal acuity falloff.

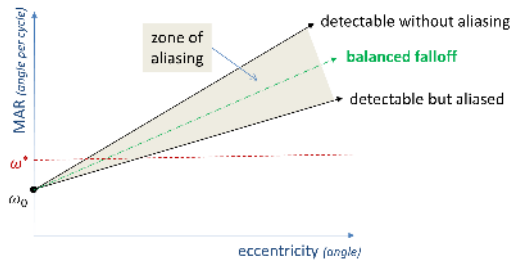


Figure 2: *Acuity falloff with eccentricity. A linear model in terms of the reciprocal of acuity, or MAR (minimum angular resolution), matches anatomical data and performance-based studies. The shaded area represents the "zone of aliasing" in the human visual system. Above its top limit, angular size is resolvable everywhere in the periphery (also termed "resolution acuity"). Below its bottom limit, details become invisible (also termed "detection acuity"). We seek a line between these two extremes that preserves some visible but aliased detail in the periphery, and also provides good savings in foveated rendering. ω^* denotes the display's sharpness angle: the angle subtended by two pixels given the display's width and the user's viewing distance. For typical desktop display configurations, it is larger (i.e., less detailed) than the smallest angle perceivable at the fovea, ω_0.*

slope improves savings from foveated rendering, because eccentricity layer resolution can fall off more rapidly.

Foveal acuity in healthy, non-elderly adults with corrected vision substantially exceeds the 20/20 Snellen chart standard, which equals 30 cycles per degree. It generally does not exceed 20/10 = 60 cycles per degree. A figure of 48 cycles per degree (20/12.5) is a good estimate of average foveal acuity for adults below 50 years of age [Colenbrander 2001]. We therefore choose the representative value $\omega_0 = 1/48°$.

Unlike previous work, we assume there is uncertainty in the choice of an "optimal" MAR slope m, because of the aliasing that occurs in the periphery of the human visual system. We therefore fix ω_0 and search for an m that provides sufficient quality in actual experiments with users.

4 Foveated Rendering System

Ideally, the graphics card would render at full display resolution where the gaze is centered and continuously decrease resolution outward from there. A more efficient method on current graphics hardware is to approximate this ideal by rendering several overlapped rectangular regions (see Figure 1), called *eccentricity lay-*

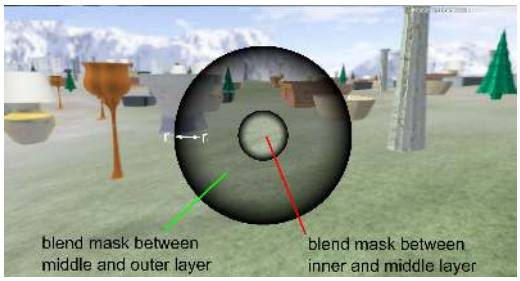


Figure 3: *Circular masks for blending eccentricity layers. Mask darkness corresponds to blend weight: black is 0 and clear is 1. At radius r_1, the blend weight is 1 for the middle layer and 0 for the outer layer. At r_2, it is 0 for the middle layer and 1 for the outer layer. Between these radii, the blend weight follows the "smoothstep" function: a cubic with zero derivatives at the endpoints. Beyond r_2 only the outer layer is visible. The inner layer is similarly blended with the middle layer using the inner blend mask. The size of the smoothstep transition region is set at 40% of the layer's radius.*

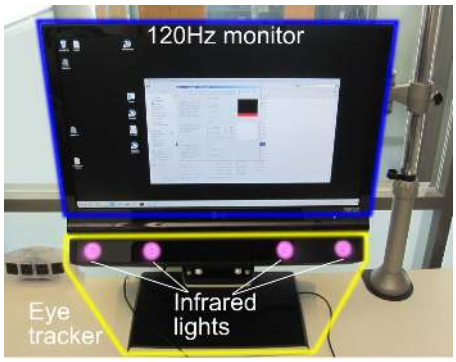


Figure 4: *Tobii TX300 Eye tracker and high speed monitor. The eye tracker has a 300Hz update rate with <10ms latency. The monitor updates at 120Hz with a measured pixel switching time of 6ms. It also has a low latency input mode which eliminates image buffering internal to the monitor. Monitors without this feature can have internal buffering latencies as high as 34ms.*

ers. All except the outer eccentricity layer cover only part of the total display area. After all layers are rendered they are interpolated to final display resolution and blended together (see Figure 3). Our current system uses three layers.

If screen update lags eye movement too much, the transition in the inner layer from blurry to sharp becomes visible after rapid eye movement [Thibos 1989]. Our system updates the inner layer at 120Hz to combat this, and all other layers at 60 Hz. We stagger the update of the middle and outer regions, updating one on even frames and the other on odd. This levels out the computation and the frame rate.

4.1 System Latency

Previous attempts to demonstrate the effectiveness of foveated rendering have been hampered by latency. We carefully chose our system hardware to minimize it. Our system comprises a Tobii TX300 Eye Tracker (Figure 4), an LG W2363D 23" 1920x1080 LCD monitor, an HPZ800 PC with Intel Xeon CPU (E5640 at 2.67GHz), and an NVidia GeForce GTX 580 GPU. The Tobii connects via Ethernet to the Z800 PC. Eye tracking, rendering, and monitor scanout from the GPU framebuffer all run asynchronously.

Measuring latency in graphics systems with eye tracking is difficult because multiple devices are communicating asynchronously. We measure each component's latency and analyze asynchronous device interactions to find the best- and worst-case overall latency. Four main sources contribute: the TX300 eye tracker, scene render time, monitor scanout delay, and pixel switching time.[5]

The TX300 eye tracker takes from 1.3ms to 3.3ms to process data and 3.3 ms to capture a frame. Best case latency can be as low as 6.6ms, which we will round to 7ms for our latency calculations. Worst case latency for the TX300 is less than 10ms.

The LG monitor updates at 120Hz so scan out latency is 8.3ms which we round to 8ms. This monitor has a low latency input mode which eliminates internal image buffering.[6] The manufacturer claims a 3ms pixel switching time but we measured 6ms, using a Phantom Miro M110 2000fps camera.

[5]Network latency between the Tobii and the CPU and PCI bus latency between the CPU and the GPU are negligible by comparison.

[6]Most LCD computer monitors buffer a small number of frames internally, some as many as four. At a 120Hz refresh rate this adds 34ms of latency.

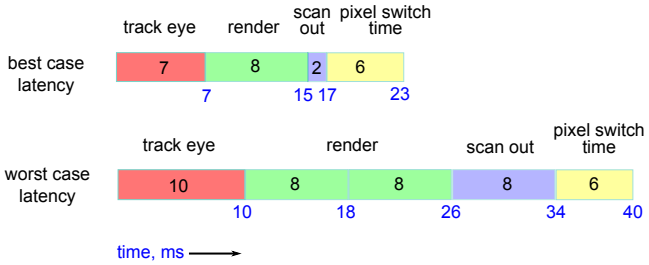


Figure 5: *System latency. In the* best *case, tracker latency is 7ms. The eye track event arrives just before the render phase begins so scene parameters are updated immediately before rendering. The monitor scans out the part of the screen the user is looking at so the first scanline of the inner layer is undelayed and the time to scan the ~200 scanlines it contains is about 2ms. In the* worst *case, tracker latency is 10ms. The eye track event arrives just after a rendering has already begun. The first render period uses old scene parameters; new parameters are unavailable until the next render period 8ms later, adding two render periods to latency. The monitor scans out the line just following where the user is looking so the inner layer is not updated for an additional 8ms. Pixel switch time adds 6ms in best and worst cases.*

Scene rendering takes 8ms at a 120Hz refresh rate. It is possible for an eye tracking event to arrive at the CPU just after scene parameters have been set for the current frame. The updated scene parameters are not available until this obsolete frame finishes rendering, adding an additional frame of latency.

Based on these individual latencies, we analyze the possible asynchronous interleavings of events and derive best and worst case system latencies for updating the inner layer where delays are most critical. The results are shown in Figure 5.

4.2 Antialiasing

Antialiasing already challenges traditional, hardware-accelerated 3D rendering because of its point sampling. The challenge increases with foveated rendering, where peripheral eccentricity layers are sub-sampled by as much as a factor of 6 in each dimension. Spatial aliasing in the peripheral layers generates distracting temporal artifacts during scene motion. A coherent artifact, such as crawling along a sharp edge or thin highlight, is especially bad because the eye is drawn to it. The obvious solution of increasing sampling density negates any performance improvement from foveation.[7]

In addition to being efficient, the antialiasing method should be generic, requiring little or no modification to shaders or geometry and making it easy to retrofit existing graphics applications for foveation. We used three techniques to improve antialiasing: hardware multi-sample antialiasing (MSAA), temporal reprojection [Guenter 1994; Nehab et al. 2007], and whole frame jitter sampling [Cook et al. 1984]. These add little extra computational cost and can easily be incorporated into existing applications.

Figure 6 describes our antialiasing method. Each component reduces artifact coherence, aliasing energy, or both. MSAA increases effective resolution along silhouette edges, significantly reducing aliasing energy there. Whole frame jitter combined with temporal reprojection reduces the coherence and energy of aliasing throughout the image.

[7]The test scenes used in our experiments contain run-time, procedurally-generated surface textures for which conventional MIPMAP texture prefiltering cannot be used. This complicates antialiasing. Regardless of the scene representation, and even with conventional image-based textures, geometric and shading aliasing remain problems that are worse with foveated rendering.

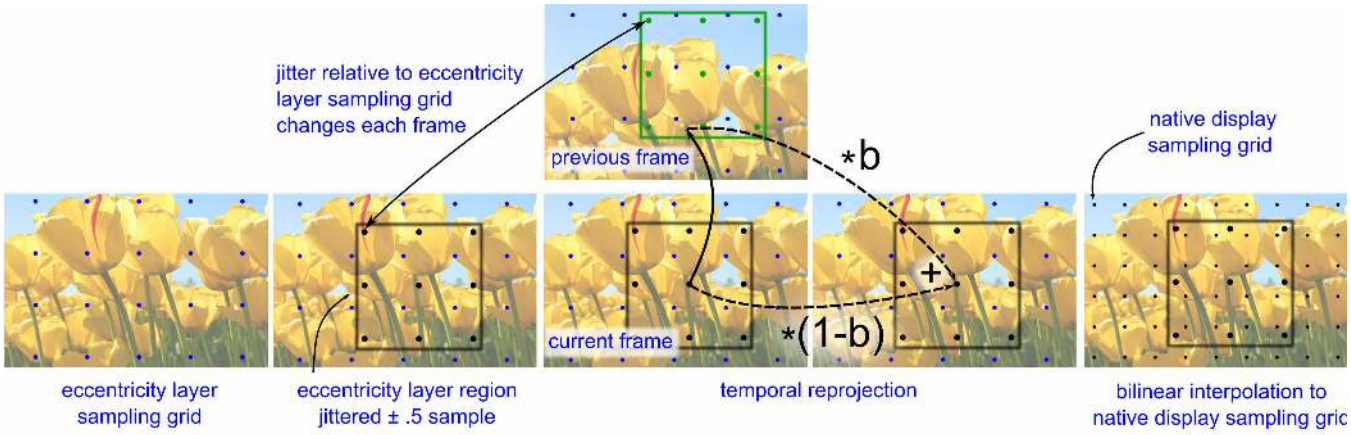


Figure 6: *Antialiasing. Each eccentricity layer has its own sampling grid. The inner layer grid coincides with the native display grid but the other layers do not. Every frame, each eccentricity layer is jittered by ± .5 of its pixel pitch. The layer's current pixel values are blended with their temporally reprojected value from the previous frame to increase the effective sampling. Finally, the layer is resampled to the native display sampling grid using bilinear interpolation.*

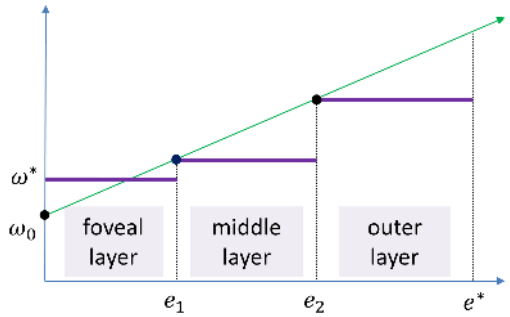


Figure 7: *Eccentricity layer parameter selection for $n = 3$ layers. We are given a particular acuity falloff to target in terms of an MAR line with slope m, drawn in green. We optimize over layer angular radii, e_1 and e_2, to minimize the sum of the pixels in all three layers. The inner layer is always sampled at the smallest MAR the display supports, ω^*. e^* denotes the angular radius of the entire display: half its field of view.*

Combining all three techniques dramatically reduces aliasing artifacts, as can be seen in the accompanying video.

4.3 Eccentricity Layer Parameter Optimization

Eccentricity layers are nested and overlap around the gaze point, as shown in Figure 1. A few discrete layers approximate the smooth falloff of frequency sensitivity with angular eccentricity in the human visual system. More layers improve the approximation but also add rendering and memory overhead. Three layers is close to optimal in our current system, but more layers increase efficiency at higher display resolution. We optimize to find layer parameters based on a simple but fundamental idea: for a given acuity falloff line, find the eccentricity layer sizes which support at least that much resolution at every eccentricity, while minimizing the total number of pixels across all layers.

Given an acuity falloff based on a slope m from Eq. 1, and the display's configuration parameters (horizontal resolution D^*, width W^*, distance from viewer V^*, and aspect ratio[8] $\alpha^* \leq 1$), a simple optimization procedure is invoked to select the size and sampling rate for each eccentricity layer. We outline the general procedure for n layers. Denote a layer by L_i, where $i = 1$ indexes the inner layer and $i = n$ indexes the outermost, with corresponding angular radius e_i, and sampling factor (pixel size) $s_i \geq 1$ represented as a multiple of the (unit) pixel size of the native display.

The innermost and outermost layers are special. The innermost layer always uses the native display's sampling rate (i.e., $s_1 = 1$). We cannot exceed native display resolution, even though it may not match human foveal acuity at the desired viewing distance, V^*. The outermost layer is not a square image centered about the gaze point like the other layers: it subsamples the entire screen image and preserves the display's aspect ratio.

The layer parameters are chosen to provide a piecewise-constant approximation to the MAR line targeted. This is conservative in the sense of always lying below the desired line; it preserves at least as much frequency content in the periphery as the target.

Under these assumptions, as shown in Figure 7, choosing any set of progressively increasing layer radii e_i determines the corresponding layer sampling factors s_i. Each layer must be sampled at the smallest MAR at which it appears on the screen, corresponding to the eccentricity just outside the previous layer's angular extent. The appropriate computation is given by $s_1 = 1$ and

$$s_{i+1} = \frac{\omega_i}{\omega^*} = \frac{m\, e_i + \omega_0}{\omega^*} \tag{2}$$

for $i \in [1,\, n-1]$. As before, ω^* represents the display's sharpness angle in terms of degrees per cycle (one cycle equals two pixels). The horizontal size/diameter D_i of each layer in pixels can be computed from its angular radius e_i and sampling factor s_i via

$$D_i = \begin{cases} 2\,\dfrac{D^*}{s_i}\,\tan(e_i)\,\dfrac{V^*}{W^*}, & 1 \leq i < n \\ \dfrac{D^*}{s_i}, & i = n \end{cases} . \tag{3}$$

The rendering work done in each layer can then be estimated by the number of pixels it contains, given by

$$P_i = \begin{cases} (D_i)^2, & 1 \leq i < n \\ \alpha^*\,(D_i)^2, & i = n \end{cases} . \tag{4}$$

Finally, we minimize the total work

$$P = \sum_{i=1}^{n} w_i\, P_i \tag{5}$$

[8] Our analysis assumes a "landscape" display aspect. For "portrait" displays, $\alpha^* > 1$; width and resolution should be measured with respect to the longer vertical axis.

as a function of the $n-1$ optimization variables e_1, $e_2, \cdots$, e_{n-1}. We use brute force minimization by discretizing the display's angular extent, e^*, and testing the objective at each point $0 < e_1 < e_2 < \cdots < e_{n-1} < e^*$. We set $w_i = 1$.

The display's angular radius may be calculated from the basic configuration parameters via

$$e^* = \tan^{-1}\left(\frac{W^*}{2\,V^*}\right), \tag{6}$$

and its sharpness angle[9] via

$$\omega^* = \tan^{-1}\left(\frac{2\,W^*}{V^*\,D^*}\right). \tag{7}$$

To evaluate potential speedup, estimated rendering work P may be compared to the total number of display pixels

$$P^* = \alpha^*\,(D^*)^2\,. \tag{8}$$

Weights in Eq. 5 can also be adjusted; e.g. $w_1 = 1$ and $w_i = 1/2$, $i > 1$. This accounts for the fact that the inner layer is updated twice as often as the outer layers, and is thus twice as expensive. We used the unweighted estimate of total work in our user study design and for predicting performance in our analysis, to separate spatial from temporal effects in peripheral vision. This obviously provides a more conservative estimate of savings from foveated rendering.

5 User Studies

We conducted three different experiments with the same 15 subjects to test the conditions for which foveated rendering produces both acceptable quality and quality equivalent to non-foveated rendering: a pair test, a ramp test, and a slider test.

All tests were based on a 1D space of foveation quality whose design is detailed in a supplement. The foveation quality index j is inversely related to the MAR slope m (see Eq. 2 in the supplement). A small index j denotes an aggressively foveated rendering which assumes a rapid falloff of peripheral acuity. Larger indices denote a less aggressive peripheral falloff which saves less.

The pair test presented each user with pairs of short animated sequences, each 8 seconds long and separated by a short interval (0.5s) of black. The reference element of the pair used non-foveated rendering; the other used foveated rendering at quality levels from $j = 8$ (low quality) to $j = 22$ (high quality). Pairs at all quality levels in this range were presented twice, in both orders (non-foveated then foveated, and foveated then non-foveated). After seeing each pair, users reported whether the first rendering was better, the second was better, or the two were the same quality. The experiment was designed to interrogate what foveation quality level was comparable to non-foveated rendering.

The ramp test presented each user with a set of short sequences, in which the foveation quality incrementally ramped either up to or down from a reference quality of $j_0 = 22$ to a varying foveation quality in the set $j_1 \in \{4, 5, 7, 10, 12, 15, 17, 22\}$. Users were then asked whether the quality had increased, decreased, or remained the same over each sequence. Each ramp was presented in both directions ($j_0 \rightarrow j_1$ and $j_1 \rightarrow j_0$), sampled using 5 discrete steps, each 5 seconds long and separated by a short interval of black. The study was designed to find the lowest foveation quality perceived to be equivalent to a high quality setting in the absence of abrupt transitions.

[9] For a fixed head position, sharpness angle varies over a planar display. Our definition assumes the head is aligned with the display center and measures sharpness there. This represents a worst-case (maximum) displayable angular size.

Figure 8: *Objects in our user study.*

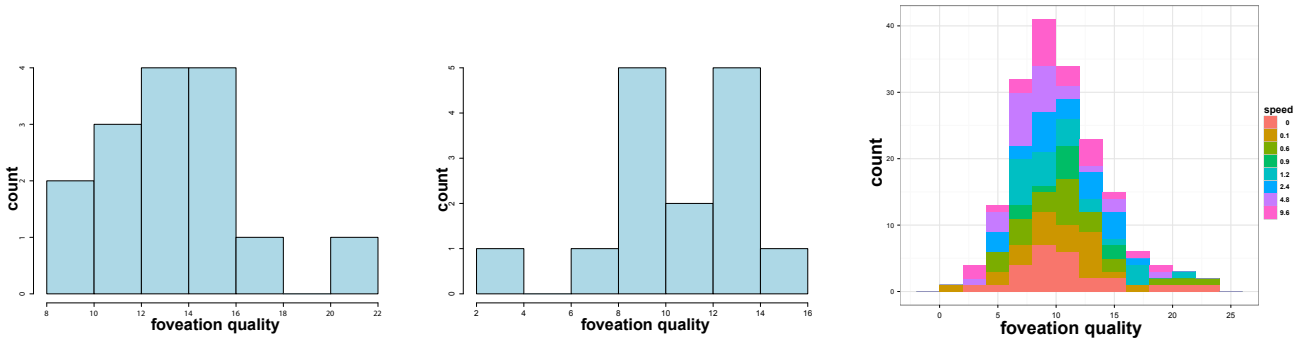
Finally, the slider test let users navigate the foveation quality space themselves. Users were first presented with a non-foveated animation as a reference. Then starting at a low level of foveation quality ($j = 4$), users could increase the level, show the non-foveated reference again, or decrease the level, with the stated task of finding a quality level equivalent to the non-foveated reference. We recorded the first quality level index at which users stopped increasing the level and instead compared it to the reference. This test also explored the effect of animation speed on the demand for foveation quality, by running the slider test across six different speeds of the moving camera, a stopped but panning camera, and a completely static camera, yielding a total of 8 different camera motions. Each camera motion was presented to each subject twice, for a total of 16 separate slider tests.

Graphical content for the study involved a moving camera through a static 3D scene, composed of a terrain, a grid of various objects positioned above it, and an environment map showing mountains and clouds. The objects range from diffuse to glossy and were rendered with various types of procedural shaders, including texture and environment mapping; refer to Figure 8. The non-foveated reference renders at about 40Hz, as high as our system supports. Representative animations are contained in the accompanying video.

We used the LG W2363D 120Hz LCD monitor of resolution 1920×1080 for our user study. Display configuration parameters were $V^* = 59\text{cm}$, $W^* = 51\text{cm}$, $D^* = 1920$, and $\alpha^* = 9/16 = 0.5625$. This yields an angular display radius of $e^* = 23.4°$ (Eq. 6), and an angular display sharpness of $\omega^* = 0.0516°$ (Eq. 7), which represents only a fraction of human foveal acuity, $\omega_0/\omega^* \approx 40\%$.

Study Results For the pair test, we identified a foveation quality threshold for each subject as the lowest variable index j he or she reported as equal to or better in quality than the non-foveated reference. For the ramp test, we identified this threshold as the lowest quality index for which each subject incorrectly labeled the ramp direction or reported that quality did not change over the ramp. The resulting extracted thresholds for the pair test had a mean of 14.9 and standard deviation of 3.07. For the ramp test, the mean was 11.9 and standard deviation was 3.48. Histograms for these thresholds are shown in Figure 9. For the slider test, the mean threshold was 14.5 with standard deviation 4.1, across all subjects and speeds.

Raw data for the user study is included in the supplement, as well as further information about study participants. The supplement also documents an informal user study which provides evidence that a



(a) *pair test* **(b)** *ramp test* **(c)** *slider test*

Figure 9: *Foveation quality histograms for the three tests in our formal user study.*

single setting for foveation quality works well for most users. Informally, we have also tried the system with multiple, rigidly moving objects as well as camera motion. We notice no difference in foveation quality between the two types of motion.

Analysis and Discussion The difference in means between the pair and ramp tests was statistically significant (see the supplement for the analysis). We found no statistically significant dependence of foveation quality demand on animation speed in the slider test.

We therefore distinguish two quality targets, A and B. From the ramp test, we obtain the more aggressive mean threshold $j_A \approx 12$, where quality is neither perceptibly degraded or enhanced over a short progression compared to a high-quality foveated reference ($j_0 = 22$). From the pair test, we obtain the more conservative mean threshold $j_B \approx 15$, where foveation quality is equivalent to a non-foveated reference. Slider test results generally confirm the pair test's, probably because we showed users the non-foveated rendering as a quality reference at the start of each trial.

We finally obtain the following estimates for model slope m:

$$\begin{aligned} j_A = 12 \quad &\Rightarrow \quad m_A = 0.0275 = 1.65' \text{ per eccentricity } ^\circ \\ j_B = 15 \quad &\Rightarrow \quad m_B = 0.0220 = 1.32' \text{ per eccentricity } ^\circ \end{aligned}$$

We can use these two estimated slopes to predict foveation savings with displays that are bigger and sharper than our current system's. Figure 12 graphs this speedup in terms of a varying field of view, assuming a display of aspect ratio $\alpha^* = 3/4$ whose resolution supports 100% of average human foveal acuity; i.e., $\omega^* = \omega_0$. Our current desktop display configuration supports only 40% of this level, and would require 6.25 times as many pixels to do so at 100%. Figure 13 highlights the slower growth in rendering cost that foveated rendering achieves compared to non-foveated, as field of view increases at a constant sharpness.

Graphs of predicted performance calculate horizontal display resolution for a given field of view $\theta^* = 2\,e^*$ using Eqs. 6 and 7, via

$$D^*(\theta^*) = \frac{4\,\tan(e^*)}{\tan(\omega_0)} = \frac{4\,\tan(\theta^*/2)}{\tan(\omega_0)}. \qquad (9)$$

Total number of display pixels, all of which are processed in non-foveated rendering, is P^* from Eq. 8, yielding the graph in Figure 12b. Cost increases as the tangent of e^* squared.

6 System Performance

Measured performance is shown in Figure 10 and Table 1. MSAA was set to 4 samples per pixel for both foveated and non-foveated

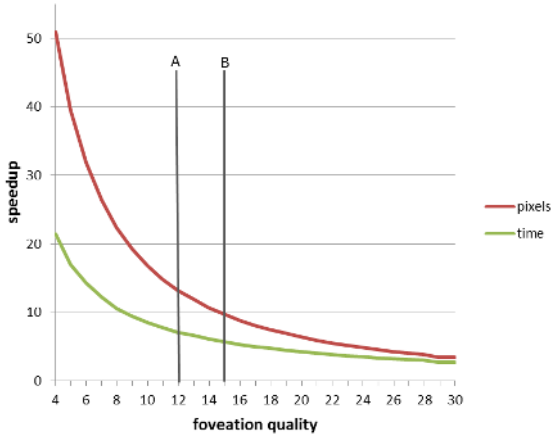


Figure 10: *Measured foveation speedup as a function of foveation quality. For each quality j, the upper curve ("pixels") plots the ratio of the number of display pixels to the sum of pixels in all eccentricity layers. The lower curve ("time") plots the measured time to render the full resolution display over the time to render and composite the eccentricity layers. Quality level A is based on a ramped comparison to a high-quality foveated reference. Quality level B is based on a direct, pair-wise comparison to a non-foveated reference. Both levels are based on means across all study subjects.*

rendering, for the user study and in Figures 10 and 11.[10] Performance measurement averaged over a scene fly-through of 5377 frames, with the gaze centered on the closest approaching object.[11]

We disabled synchronization to the monitor refresh (v-sync) and simply recorded the time needed to complete the flythrough, using the respective foveated or non-foveated rendering method.

Foveated time measurements include all computation: the time to render all three eccentricity layers and to perform bilinear reconstruction and compositing to the final display. In fact, layer reconstruction and compositing form only a small part of the computation (0.2-0.3ms), as can be seen by comparing the "No SkyBox" and "No Render" rows for the three foveated rendering methods in Table 1. The same is true for temporal reprojection, as can be seen

[10] Table 1 compares non-foveated performance with and without MSAA so its cost can be isolated: compare column N_1 with N_0.

[11] This gaze control is conservative. We obtain larger measured speedups when the gaze is fixed at the center of the screen, where an expensively shaded object appears only haphazardly. We think it is fairer to assume the user will focus on expensively-rendered objects on the screen, as is done in our measurement. Gaze control was performed by recording the gaze position when constantly looking at the nearest approaching object and then using those recorded positions in the performance test.

Rendering	I	Q_A	Q_B	N_0	N_1	N_2
Full	4.5	4.9	5.9	23.6	27.7	27.9
Cheap Terrain	2.6	2.8	3.2	6.6	9.2	9.6
No Objects	0.8	0.8	0.9	1.0	1.4	1.6
No Terrain	0.8	0.8	0.8	0.4	0.5	0.7
No SkyBox	0.6	0.5	0.6	0.3	0.4	0.5
No Render	0.3	0.3	0.3	0.3	0.4	0.3

Table 1: *Measured system performance at selected settings. Times are in milliseconds. Columns represent level of foveation: "I" denotes foveated rendering at the "informal study" settings described in the supplement; "Q_A" and "Q_B" refer to foveation at quality levels A and B from our formal user study. The rightmost three columns denote non-foveated rendering. "N_0" is non-foveated with no MSAA, jitter, or reprojection. "N_1" is non-foveated with MSAA. "N_2" is non-foveated with MSAA, jitter, and reprojection, as actually used in our user study as a reference. Rows show graphical content, which progressively degrades from top to bottom. "Full" includes all content as presented in our user study. "Cheap Terrain" replaces the procedural shading on the terrain with a simpler version. "No Objects" further removes all objects above the terrain. "No Terrain" removes the terrain surface entirely. "No SkyBox" removes the environment map. Finally, "No Render" eliminates all graphical content and just executes a framebuffer clear and a presentation buffer swap.*

by comparing columns N_1 and N_2.

Unlike the predictions in Section 5 (Figures 12 and 13), measured pixel counts for foveated rendering account for the fact that the middle and outer layers are temporally subsampled. We simply weight their pixel counts, P_2 and P_3, by 1/2. Measured timings obviously also reflect any temporal subsampling.

The speedup factor in terms of number of pixels rendered compared to non-foveated was 13.3 for quality level A and 9.8 for quality level B. Overall measured speedup over non-foveated rendering was a factor of 5.7 at quality level A and 4.8 at level B (see Section 5 for a discussion of the two quality levels).

At the "informal study" foveation settings (see supplement), we obtained a speedup factor of about 15 in terms of number of pixels rendered, and 6.2 in overall graphics performance. Refer to the "I" column in Table 1.

Figure 11 decomposes rendering cost for each of the three layers and shows how render time relates to each layer's pixel area, over the space of foveation quality.

Geometric Complexity Our system represents objects as parametric surfaces tessellated using a continuous LOD algorithm. Tessellation is based on the squared distance between the camera and the object's center. As squared distance goes from MinDistSq to MaxDistSq, each object's tessellation ranges from MaxTriCount to MinTriCount. A separate tessellation curve can be specified for each layer. We set the per-object MaxTriCount to 20,000 for the inner and middle layers, and to 8,000 in the outer layer.

Our test scenario contains 400 such parametric objects arranged in a 20×20 grid plus a terrain consisting of about 20k triangles. When non-foveated, our system renders roughly 250k triangles per frame visible in the view frustum. Geometric LOD for the inner and middle layers is identical to non-foveated rendering. The outer layer is rendered with roughly half as many triangles as full LOD. This saves only 0.5ms. We included it to show feasibility: an alpha-blend between layers, sufficiently far out in the periphery, masks LOD transitions.

Our system's simple LOD approach could easily be replaced by a more sophisticated one. Our results show that a geometric LOD scheme can reduce geometry, at least in the outer layer, without introducing objectionable artifacts. Because of its larger view frustum, a modest reduction in the peripheral layer's geometry more than compensates for increased load from the inner layers.

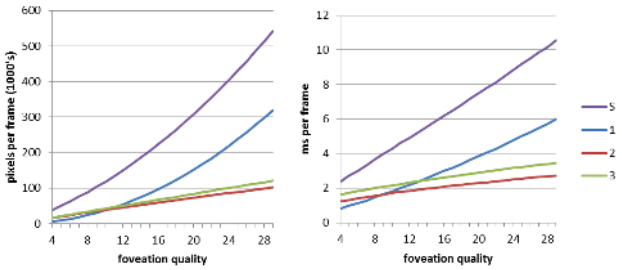


Figure 11: *Performance breakdown by eccentricity layer. "S" represents the total. The left graph plots number of pixels shaded; the right plots total render time per frame. At the lowest quality settings, render time is dominated by the outer, lowest-resolution layer, probably due to its higher geometric complexity. As quality increases in our 1D space, the foveal layer's cost rises faster than the cost of the other layers. At the quality settings indicated by our user study (j=12-15), rendering work is well balanced across the three layers.*

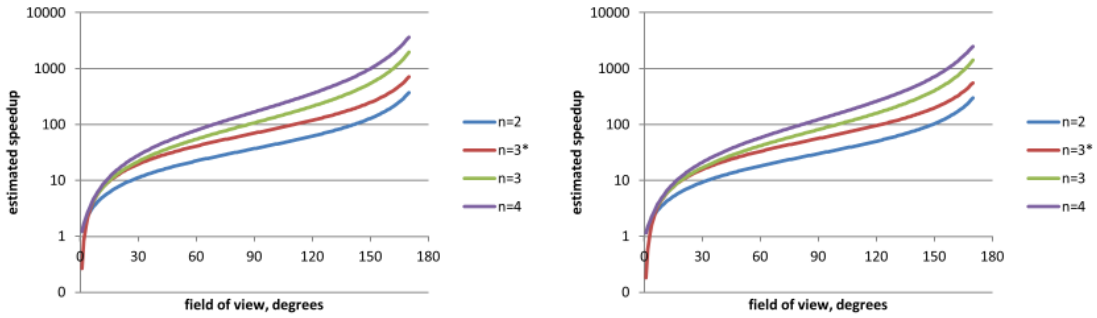
Foveation Retrofitting With quick modifications, we also "foveated" an existing application which allows 3D navigation in the dynamic, planet-wide world shown in Figure 14. This application was not written by us and was designed without considering foveation. Its patch-based geometric LOD system was used unchanged for all three eccentricity layers, with no attempt made to accomodate LOD to eccentricity. Nevertheless, we obtained speedups similar to those documented for our test system: a worst-case speedup of about a factor of 5 at the "informal study" foveation settings, and significantly higher speedups in many views. Measured speedup would have been higher still had we enabled MSAA in the non-foveated rendering.

7 Conclusion

Our experiments show that foveated rendering improves graphics performance by a factor of 5-6 on current desktop displays at HD resolution, achieving quality comparable to standard rendering in a user study. We introduce several new ideas, including the application of eccentricity layers to accelerate general 3D graphics, a new method for peripheral layer antialiasing, and a precomputed optimization for eccentricity layer parameter selection. Our user studies establish a specific range of MAR slopes (0.022-0.034) that can be targeted to obtain a quality range from acceptable through equivalent to traditional rendering, and used in a simple model to predict savings from foveation in any display configuration. Though our system uses an external desktop display, its approach and experimental results should apply to a head-mounted one too.

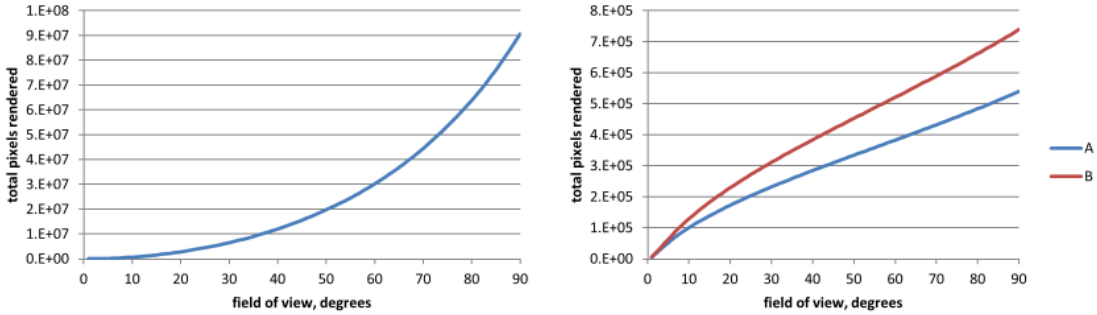
Our findings suggest much greater savings at higher display resolution and field of view. We expect foveated rendering to become increasingly important as display field of view enlarges, resolution increases to match human acuity, and users demand more mobile and lower-power graphics performance. Our work also suggests the likely reduction in bandwidth to be had by "foveating" the video signal transmitted between GPU and display [Bergström 2003].

When foveating applications that target a 60Hz frame rate, only half our speedup is available to improve overall graphics quality, e.g. to increase the complexity of geometry or shaders. The other half is devoted to doubling the frame rate of the inner layer, with obvious benefits for smoothing the animation's appearance and improving interactivity. Lower latency eye tracking may ease the urgency of a



(a) *foveation quality* Q_A **(b)** *foveation quality* Q_B

Figure 12: *Predicted speedup from foveated rendering. Speedup is estimated by dividing the total number of display pixels by the number of pixels rendered in all eccentricity layers (i.e., the ratio P^*/P from Section 4.3). Foveated rendering uses layer parameters from optimizing Eq. 5. We assume a display aspect ratio of $\alpha^* = 0.75$ and sufficient resolution at each field of view so that display sharpness exactly equals average human foveal acuity; i.e. $\omega^* = \omega_0$. The four different curves are based on optimization with different numbers of layers, $n = 2, 3\,4$. The starred curve ($n = 3^*$) uses three layers but restricts the middle layer to a sampling factor of $s_2 = 2$, as is documented in the supplement and applied in our user study. Desktop planar displays matched to human acuity at large fields of view ($> 90 - 120°$) are impractically large both in physical size and resolution but are included here for completeness.*



(a) *non-foveated (fast cost growth)* **(b)** *foveated (slow cost growth)*

Figure 13: *Predicted rendering cost. Our assumptions are the same as in Figure 12: the display's field of view varies while retaining resolution sufficient to match 100% of human foveal acuity. Non-foveated rendering cost increases rapidly as field of view increases, shown in (a). Foveated rendering cost increases much more slowly and is in fact roughly linear up to an angular diameter of 90°, as shown in (b). Foveated rendering cost is plotted at both slopes estimated from our user study, using $n = 4$ optimized eccentricity layers. Note that the vertical scale in (a) is 125 times higher than in (b).*

120Hz update for the foveal layer. On new platforms, our speedup can be translated into an improvement of graphics quality or a reduction in power usage and heat dissipation.

In future work, we would like to investigate what constitutes acceptable foveation quality without any ground truth referent. More work is needed to validate our model for displays at higher resolution and field of view, and to investigate how much latency is tolerable. Further study could confirm the conjecture that antialiasing based on our limited peripheral acuity actually enhances performance on visual tasks [Murphy and Duchowski 2007]. Finally, foveation encourages new tradeoffs that deserve study, in which different high-level parameters or even entirely different algorithms are applied to render the different eccentricity layers.

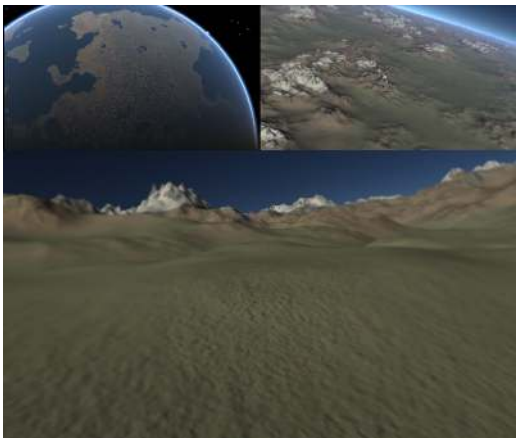


Figure 14: *Planet demo. Images are shown foveated at the image center.*

Several changes to hardware and software elements of our system would make foveation still more effective. Two of these are eye tracking with lower latency and a bigger tracking volume, and graphics hardware that allows per-pixel jitter. Our system does not access the high-resolution MSAA "supersamples" when performing temporal reprojection; only the final resolved buffer. We note that peripheral subsampling increases the need for prefiltered shaders [Han et al. 2007; Olano and Baker 2010], an active area of research. Graphics architectures based on ray tracing may efficiently permit a more continuous falloff of sampling density with eccentricity [Murphy and Duchowski 2007].

Acknowledgments Thanks to Arthur Shek for the high-resolution environment map used in our test scenario.

References

BAUDISCH, P., DECARLO, D., DUCHOWSKI, A. T., AND GEISLER, W. S. 2003. Focusing on the essential: considering attention in display design. *Commun. ACM 46* (March), 60–66.

BERGSTRÖM, P. 2003. *Eye-movement Controlled Image Coding*. PhD thesis, Linköping University, Linköping, Sweden.

CHENG, I. 2003. Foveated 3D model simplification. In *Proceedings, Signal Processing and its Applications*, 241–244.

COLENBRANDER, A. 2001. Vision and vision loss. In *Duane's Clinical Opthalmology*. Lippincott, Williams and Wilkins, ch. 51.

COOK, R. L., PORTER, T., AND CARPENTER, L. 1984. Distributed ray tracing. *SIGGRAPH Comput. Graph. 18* (January), 137–145.

DUCHOWSKI, A. T., AND ÇÖLTEKIN, A. 2007. Foveated gaze-coningent displays for peripheral LOD management, 3D visualization, and stereo imaging. *ACM Trans. on Multimedia, Communications and Applications 3*, 4 (Dec.).

DUCHOWSKI, A. T. 2002. A breadth-first survey of eye tracking applications. *Behavior Research Methods, Instruments, and Computers 34*, 455–70.

FUNKHOUSER, T., AND SEQUIN, C. 1993. Adaptive display algorithm for interactive frame rates during visualization of complex virtual environments. In *Computer Graphics (SIGGRAPH '93 Proceedings)*, vol. 27, 247–254.

GEISLER, W. S., AND PERRY, J. S. 2002. Real-time simulation of arbitrary visual fields. In *Proceedings of the symposum on eye tracking research applications ETRA*.

GUENTER, B. 1994. Motion compensated noise reduction. Tech. rep., Microsoft Research.

HAN, C., SUN, B., RAMAMOORTHI, R., AND GRINSPUN, E. 2007. Frequency domain normal map filtering. *ACM Trans. Graph. 26* (July).

HORVITZ, E., AND LENGYEL, J. 1997. Perception, attention, and resources: a decision-theoretic approach to graphics rendering. In *Proceedings of the Thirteenth conference on Uncertainty in artificial intelligence*, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, UAI'97, 238–249.

LEVOY, M., AND WHITAKER, R. 1989. Gaze-directed volume rendering. Tech. rep., University of North Carolina.

LOSCHKY, L. C., AND MCCONKIE, G. W. 2000. User performance with gaze contingent multiresolutional displays. In *Proceedings of the 2000 symposium on Eye tracking research & applications*, ETRA '00, 97–103.

LUEBKE, D., HALEN, B., NEWFIELD, D., AND WATSON, B. 2000. Perceptually driven simplification using gaze-directed rendering. Tech. rep., University of Virginia.

MURPHY, H., AND DUCHOWSKI, A. T. 2001. Gaze-contingent level of detail rendering. In *Eurographics 2001 (Short Presentations)*.

MURPHY, H. A., AND DUCHOWSKI, A. T. 2007. Hybrid image/model based gaze-contingent rendering. *Applied Perception in Graphics and Visualization*.

MYSZKOWSKI, K. 2002. Perception-based global illumination, rendering, and animation techniques. In *SCCG '02: Proceedings of the 18th Spring Conference on Computer Graphics*, 13–24.

NEHAB, D., SANDER, P. V., LAWRENCE, J. D., TATARCHUK, N., AND ISIDORO, J. R. 2007. Accelerating real-time shading with reverse reprojection caching. In *ACM SIGGRAPH/Eurographics Symposium on Graphics Hardware*, 25–35.

OHSHIMA, T., YAMAMOTO, H., AND TAMURA, H. 1996. Gaze-directed adaptive rendering for interacting with virtual space. In *Proceedings of the 1996 Virtual Reality Annual International Symposium (VRAIS 96)*, IEEE Computer Society, Washington, DC, USA, VRAIS '96, 103–110.

OLANO, M., AND BAKER, D. 2010. LEAN mapping. In *Proceedings of the 2010 ACM SIGGRAPH symposium on Interactive 3D Graphics and Games*, ACM, New York, NY, USA, I3D '10, 181–188.

O'SULLIVAN, C., DINGLIANA, J., AND HOWLETT, S. 2002. Gaze-contingent algorithms for interactive graphics. In *The Mind's Eyes: Cognitive and Applied Aspects of Eye Movement Research*, J. Hyönä, R. Radach, and H. Deubel, Eds. Elsevier Science, Oxford.

RAMANARAYANAN, G., FERWERDA, J., WALTER, B., AND BALA, K. 2007. Visual equivalence: towards a new standard for image fidelity. *ACM Trans. Graph. 26* (July).

REDDY, M. 1998. Specification and evaluation of level of detail selection criteria. *Virtual Reality 3*, 2, 132–143.

REDDY, M. 2001. Perceptually optimized 3D graphics. *Computer Graphics and Applications, IEEE 21*, 5 (sep/oct), 68 –75.

REINGOLD, E., LOSCHKY, L. C., MCCONKIE, G. W., AND STAMPE, D. M. 2003. Gaze-contingent multiresolution displays: an integrative review. *Human Factors 45*, 2, 307–28.

STRASBURGER, H., RENTSCHLER, I., AND JÜTTNER, M. 2011. Peripheral vision and pattern recognition: a review. *Journal of Vision 11*, 5, 1–82.

THIBOS, L. N. 1989. Image processing by the human eye. In *Visual Communications and Image Processing IV*, W. A. Pearlman, Ed., Proc. SPIE 1199, 1148–1153.

VIRSU, V., AND ROMANO, J. 1979. Visual resolution, contrast sensitivity, and the cortical magnification factor. *Experimental Brain Research 37*, 475–494.

WATSON, B., WALKER, N., HODGES, L. F., AND WORDEN, A. 1997. Managing level of detail through peripheral degradation: effects on search performance with a head-mounted display. *ACM Trans. Comput.-Hum. Interact. 4* (December), 323–346.

WATSON, B., WALKER, N., AND HODGES, L. F. 2004. Supra-threshold control of peripheral LOD. *ACM Trans. Graph. 23* (Aug.), 750–759.

WEAVER, K. A. 2007. *Design and evaluation of a perceptually adaptive rendering system for immersive virtual reality environments*. Master's thesis, Iowa State University.

WEYMOUTH, F. W. 1958. Visual sensory units and the minimum angle of resolution. *American Journal of Opthalmology 46*, 102–113.

YEE, H., PATTANAIK, S., AND GREENBERG, D. P. 2001. Spatiotemporal sensitivity and visual attention for efficient rendering of dynamic environments. *ACM Trans. Graph. 20* (January), 39–65.