Foveated Instant Preview for Progressive Rendering

Matias Koskela
Tampere University of Technology

Kalle Immonen
Tampere University of Technology

Timo Viitanen
Tampere University of Technology

Pekka Jääskeläinen
Tampere University of Technology

Joonas Multanen
Tampere University of Technology

Jarmo Takala
Tampere University of Technology

ABSTRACT

Progressive rendering, for example Monte Carlo rendering of 360° content for virtual reality headsets, is a time-consuming task. If the 3D artist notices an error while previewing the rendering, he or she must return to editing mode, do the required changes, and restart rendering. Restart is required because the rendering system cannot know which pixels are affected by the change. We propose the use of eye-tracking-based optimization to significantly speed up previewing the artist's points of interest. Moreover, we derive an optimized version of the visual acuity model, which follows the original model more accurately than previous work. The proposed optimization was tested with a comprehensive user study. The participants felt that preview with the proposed method converged instantly, and the recorded split times show that the preview is 10 times faster than conventional preview. In addition, the system does not have measurable drawbacks on computational performance.

CCS CONCEPTS

• **Computing methodologies** → **Rendering**; *Graphics systems and interfaces*; Ray tracing;

KEYWORDS

Progressive rendering, Preview, Foveated rendering, Eye tracking

ACM Reference Format:
Matias Koskela, Kalle Immonen, Timo Viitanen, Pekka Jääskeläinen, Joonas Multanen, and Jarmo Takala. 2017. Foveated Instant Preview for Progressive Rendering. In *SA '17 Technical Briefs: SIGGRAPH Asia 2017 Technical Briefs, November 27–30, 2017, Bangkok, Thailand*. ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3145749.3149423

1 INTRODUCTION

Virtual Reality (VR) devices are getting more and more common for both work and entertainment applications. However, one of the challenges of VR is how to easily generate 360° content, because of its high resolution, and the requirement of having meaningful interesting content in every direction. Rendering high resolution images with realistic-looking progressive rendering methods typically takes hours to complete. Noisy images of the rendering are produced quickly. However, for example in Monte Carlo rendering, reducing the error of the estimator to half at any point requires the number of samples to be quadrupled [Pharr and Humphreys 2010].

If the artist notices during the preview that something was wrong in the scene, he or she must cancel the rendering, make the required changes, and start the rendering all over again. Restart is required because the system cannot know what pixels are affected by the change. Typically, the artist can create a rough estimate of the scene with a faster rendering method, but the error-free version becomes visible only after the slow progressive rendering process, especially if the scene has reflections, transparency, or soft shadows. If the artist can preview the rendering faster, it directly transfers to the speed of the whole content generation process. Compared to conventional rendering, the high resolution of 360° content makes the preview even slower, which makes its optimization more important.

In this paper, we propose a method for optimizing preview of progressive rendering by applying *foveated rendering*, i.e., reducing the quality of rendering in the peripheral regions of vision. Quality can be reduced because the human visual system cannot detect fine detail outside the center of the vision. Moreover, it has been predicted that more than 90% of real-time path tracing samples can

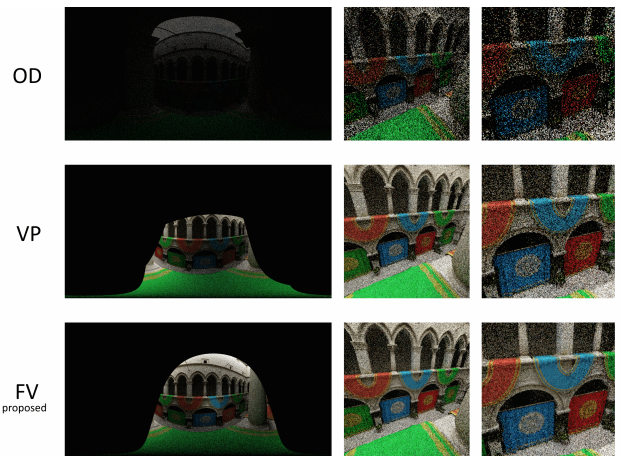

Figure 1: Results after two seconds of rendering with a static point of interest from left to right: rendering buffer, preview, and magnification of the point of interest. For the differences between the methods, see Sec. 4. Note how the point of interest already starts to converge in FV, but the edges of the preview have more noise than VP.

The work has been financially supported by the TUT Graduate School, Nokia Foundation, Emil Aaltonen Foundation, Finnish Foundation for Technology Promotion, Academy of Finland (funding decision 297548), Finnish Funding Agency for Technology and Innovation (funding decision 40142/14, FiDiPro-StreamPro) and ARTEMIS joint undertaking under grant agreement no 621439 (ALMARVI) .

SA '17 Technical Briefs, November 27–30, 2017, Bangkok, Thailand

be omitted by employing foveated rendering [Koskela et al. 2016]. This paper's novel contributions are:

(1) We derive an optimized version of the human visual acuity model, which can be followed to accurately generate path tracing samples.
(2) We propose the use of foveated rendering to speed up preview of progressive rendering, and validate with a user study that the proposed method is 10 times faster than conventional preview.

2 RELATED WORK

There is a large body of work on real-time foveated rendering, which is summarized by a recent comprehensive literature review [Weier et al. 2017]. Foveated rendering is very appealing with *Head Mounted Displays* (HMD), which typically have a wider field of view than desktop monitors, and only a single observer per display [Shibata 2002].

Current hardware supports only a fixed, predefined resolution for rasterization. Therefore, foveated rendering can be implemented more easily with ray-tracing-based techniques, because they support arbitrary sampling patterns in screen space. Consequently, foveated ray tracing has gained academic interest [Murphy et al. 2009; Weier et al. 2016]. An intuitive idea would be to distribute samples according to a model of the human visual acuity's smallest detectable spatial frequency:

$$m(e) = \begin{cases} 1.0, & e \leq 5.79 \\ \frac{7.49}{(0.3e+1)^2}, & e > 5.79 \end{cases}, \tag{1}$$

where $e \geq 0$ and it is the eccentricity angle, i.e., the angle from the gaze direction [Reddy 2001]. This model is derived from various psychophysical studies. The equation describes just one radius of the visual acuity, and the actual 2D model is obtained by taking a solid of revolution of the equation.

Due to the complexity of the visual acuity model, linear denominator models can be used instead of the quadratic denominator model shown in Eq. 1. However, they are not as accurate on the peripheral parts of the vision [Guenter et al. 2012]. A further simplified version is to use a linear fall-off between full detail and the minimum sampling probability [Stengel et al. 2016; Vaidyanathan et al. 2014; Weier et al. 2016], or even a static probability with respect to eye tracking [Pohl et al. 2016].

The context of previewing is closely related to real-time rendering, since the preview needs to be updated in real-time. Moreover, preview of a region of interest needs to converge as quickly as possible, because then the artist can cancel the rendering earlier, and make the required changes faster. One way to vary the convergence rates is to apply so-called *guided preview*, and have more samples in an area chosen by the artist with a pointing device [Roth et al. 2015]. Another idea is to select an area of the image where the sample computation is concentrated [Pixar 2017]. Importance masking [LuxRender 2013] is an advanced version of area selection.

In this paper, we utilize the idea of guided preview, and use one of the most intuitive ways for guiding, i.e., the point where the user is looking at. Moreover, we use the quadratic denominator visual acuity model instead of the significant simplifications of it.

3 PROPOSED METHOD

The idea of our preview method is to render images for VR and to give the artist an instant preview. The method tracks the eye of the user and generates more samples around the gaze direction. Sampling according to the visual acuity model does not decrease the user experience of previewing, because resolution can be reduced significantly on the peripheral parts of vision without affecting search task performance [Duchowski et al. 2009].

Sampling the world according to the visual acuity model requires random positions to be generated with probability density equal to Eq. 1. Note that the equation from Reddy [2001] is not consistent with the definition of the probability density function because its integral over the entire space is not equal to one. However, we will fix the equation so that it follows the definition in Eq. 5.

As we show below, generating random numbers according to the solid of revolution of Eq. 1 would have been too complicated for the targeted real-time preview method. Therefore, we simplify the generation by producing polar coordinates: one uniformly distributed for the angular coordinate ϕ, and another for the radial coordinate r, which is the distance from the center of the vision. To achieve correct distribution for r, the probability density of Eq. 1 must be slightly modified based on the circumference of circle $2\pi R$ (where R is the radius):

$$g(e) = 2\pi e\, m(e) = \begin{cases} 2\pi e, & e \leq 5.79 \\ \frac{14.98\pi e}{(0.3e+1)^2}, & e > 5.79 \end{cases}. \tag{2}$$

The angle can be generated by one of the many algorithms available for quickly generating uniformly distributed random numbers. In addition, uniform distribution can be transformed to any other distribution with the so-called inversion method [Devroye 1986]:

$$r = f^{-1}(u), \tag{3}$$

where u is a uniformly distributed random number in interval $[0, 1]$, f is the desired cumulative distribution function, and r is a random number that has a cumulative distribution f. The inversion method requires us to derive the cumulative distribution function from the probability density defined in Eq. 2 by taking the integral of $g(e)$ in interval $[0, x]$:

$$h(x) = \int_0^x g(e)\, de = \begin{cases} \pi x^2, & x \leq 5.79 \\ \frac{14.98}{0.09}\pi\big(\frac{1}{0.3x+1} + \ln(0.3x+1)\big) - 612.256, & x > 5.79 \end{cases}. \tag{4}$$

We chose the upper limit of the function at 80°, because the model starts to reach zero around 80°. Finally, the integral needs to be modified to be consistent with the cumulative distribution function definition that runs from 0 to 1 in y-axis:

$$f(x) = \frac{h(x)}{G(80)}, \tag{5}$$

where $G(e) = \int g(e)\, de$.

Eq. 3 requires the inverse of f in Eq. 5. However, it cannot be expressed in terms of standard mathematical functions and Lambert W-function [Weisstein 2002] would be needed. We simplified the function by approximating it with a fitted fourth-order polynomial

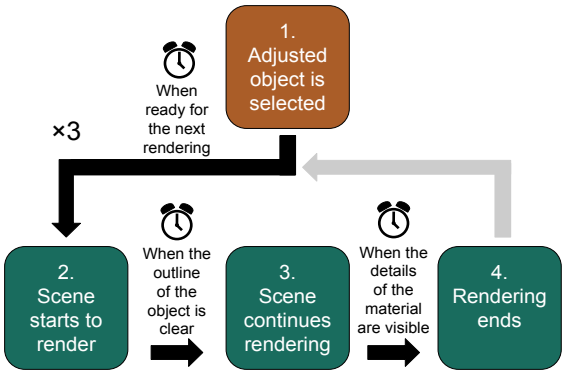

Figure 2: Illustration of the single scene procedure in the user study. Boxes are the states and arrows are the participant's actions for transitioning to other states. Clocks represent points where the system saved split times.

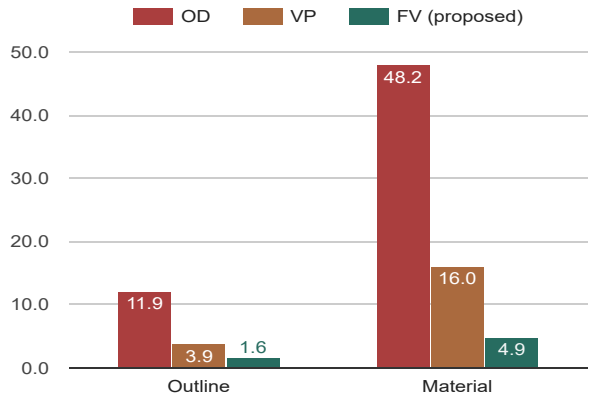

Figure 3: Geometric mean split times (less is better) over all 5 test scenes of the *visible outline* and *visible material* criteria for each of the three preview methods.

that was determined numerically by least squares regression:

$$f^{-1}(u) \approx 80 \times \begin{cases} 0.2330\sqrt{u}, & u \le 0.0965 \\ 0.3136u^4 + 0.0021u^3 + \\ \quad 0.3451u^2 + 0.2984u + 0.0404, & u > 0.0965 \end{cases}. \quad (6)$$

The maximum error of Eq. 6 to Eq. 1 is 1.8% and integral of their difference is less than 0.04%, which are very small especially if compared to approximations from previous work.

In the proposed method the users preview the results with a VR HMD that has eye tracking capability, but also a desktop setup could be used. We chose the HMD because a VR headset gives better spatial awareness and enjoyment [MacQuarrie and Steed 2017] and, therefore, it is likely that the artist wants to preview the scene with a device similar to what the end users will use.

4 USER STUDY

To measure the subjective performance of our proposed instant preview method, we conducted a user study. The study used five different scenes and three different preview methods in random order. We chose the scenes to represent different 360° rendering scenarios. The preview methods were:

- *Omnidirectional preview* (OD): In this method samples were distributed uniformly to every possible point in an equirectangular image. This method represents conventional baseline rendering without any preview optimizations.
- *Viewport preview* (VP): This method distributed samples uniformly to the area currently viewable with the HMD. The idea of this method was to simulate sampling similar to rectangle area selection tool used in some rendering engines.
- *Foveated preview* (FV): This is the proposed method which distributed samples according to the visual acuity model centered on the gaze point of the eye-tracked user.

Procedure for each 3D scene can be seen in Fig. 2. In every scene, we asked the participants to play the role of a 3D artist, and to choose an object in the 3D world. They were told that the object represents an object that they have just adjusted. Adjustment could have been, for example, changing the orientation of the object or changing its material parameters.

After the selection, the rendering started, and the participants recorded split times. The first split was recorded at the point where the participants thought that they could determine if transform or rotation of the object was successful. The second split represented the time when the participant was able to determine if the material adjustment was successful. The idea was that at these points the artist could cancel the rendering and go back to editing mode.

We chose unidirectional path tracing with importance sampling as the progressive rendering method. AMD RadeonRays [AMD 2017] was used for ray traversal and the path tracer ran on an AMD Fury X GPU. The FOVE 0 VR headset was used as a viewing device in the study due to its eye tracking capability. The system generated equirectangular images and the previewing used trilinear filtering to cancel flickering near poles.

5 RESULTS

We conducted the user study with 16 participants, of whom 11 were male and 5 were female. The ages of the participants varied between 22 and 37. Two of the participants knew details about the test set-up beforehand, but their results were close to the average of the other results.

The geometric means of all timings are shown in Fig. 3 and arithmetic means of each scene over all participants are listed in Table 1. The results show that the proposed method required only around 10% time compared to the baseline method of OD. The time saving translates directly to the speed of the artist's feedback loop, since he or she can quit the rendering and start making the required changes 10 times faster. Equivalent comparison states that previewing with VP takes around 30% time compared to OD.

In an open discussion after the test, many participants reported that FV was so fast that it was hard to record the first split at the right time. They also stated that it felt that the FV method converged instantly. On the other hand, several participants stated that slowness of OD might have caused them to get bored, inducing them to mark a split time at a lower quality than with the other methods. Most of the participants also stated that they did not

Table 1: Arithmetic mean (μ) split times and their standard deviations (σ) from the user study for each scene. The results of the FV (proposed) and VP are compared to the OD and pp stands for percent point. Big σ values in OD are caused by the participants selecting different kind of objects.

Split type	Outline visible						Material visible					
Preview method	OD		VP		FV		OD		VP		FV	
Value type	μ (s)	σ (s)	μ (%)	σ (pp)	μ (%)	σ (pp)	μ (s)	σ (s)	μ (%)	σ (pp)	μ (%)	σ (pp)
BMW	6.1	7.6	41.8	9.8	20.8	1.9	29.9	16.9	33.6	15.1	11.8	5.3
Classroom	15.6	7.7	27.1	3.1	11.6	2.1	67.9	79.2	30.8	35.7	7.9	4.6
Conference	41.9	59.7	29.7	8.6	7.6	2.3	137.2	197.4	33.9	46.0	8.2	8.6
Sibenik	24.7	20.8	29.5	9.2	11.4	3.9	76.7	55.2	34.3	29.0	10.9	10.1
Sponza	16.1	13.8	25.9	4.9	9.1	2.1	60.4	46.8	30.1	18.9	7.5	4.0

realize that eye tracking was used, and instead thought that the actual rendering was somehow faster. Not noticing the eye tracking means that the visual acuity model is a good way to distribute the samples.

Measurement of the computational performance of the three different methods states that they are computationally equally good. On the target machine, according to AMD CodeXL, it takes around 0.17 ms to launch 65,536 primary rays with all of the preview methods. The launch includes generating random pixel coordinates for the rays and calculating the ray origin and direction based on them. In the case of the proposed FV extra work is done to change the random number distribution with the inversion method (Eq. 6). The timing implies that the extra work is hidden by the latencies of memory accesses and the kernel launch.

The ray tracing performance is dependent on the user's gaze or head direction with the FV and VP methods, respectively. In contrast, OD has the same ray tracing performance no matter where the user is looking at. While OD yields a larger number of samples per second than the other methods, this result can be misleading because many of the rays are sent to directions that are easier to ray trace, e.g., straight to a skybox.

6 CONCLUSIONS

In this paper we presented a foveation-based preview system for progressive rendering. The system tracks the user's gaze and distributes samples according to a visual acuity model.

Thanks to our optimized visual acuity model, the image converges at the user's point of interest 10 times faster than with conventional uniform sampling over the whole 360° image area. Quick convergence enables the 3D artist to cancel the rendering 10 times earlier, reducing the length of the feedback loop significantly. We recorded these timings in a user study with 16 participants. The study measured when the users could detect both if a change in the transformation of an object was successful, and if a change in the material parameters was successful. Generating uniform random numbers according to the visual acuity model did not have a measurable difference on the computation performance.

The proposed system uses a head mounted display, but it could be extended to support a desktop display with eye tracking. In the future, we are interested in exploring more ways to make content generation for virtual reality devices easier and faster.

ACKNOWLEDGMENTS

The authors would like to thank the creators of the 3D models used in the user study especially Frank Meinl for the Crytek Sponza model seen in Fig. 1. In addition, the authors would like to thank Heli Väätäjä, Chelsea Kelling, and Otto Kauhanen for helpful discussions.

REFERENCES

AMD. 2017. RadeonRays SDK. Online. (2017). Available: https://github.com/GPUOpen-LibrariesAndSDKs/RadeonRays_SDK, Referenced: 24th of May 2017.

Luc Devroye. 1986. *Non-Uniform Random Variate Generation.* Springer.

Andrew Duchowski, David Bate, Paris Stringfellow, Kaveri Thakur, Brian Melloy, and Anand Gramopadhye. 2009. On Spatiochromatic Visual Sensitivity and Peripheral Color LOD Management. *ACM Transactions on Applied Perception* 6, 2 (2009).

Brian Guenter, Mark Finch, Steven Drucker, Desney Tan, and John Snyder. 2012. Foveated 3D Graphics. *ACM Transactions on Graphics* 31, 6 (2012).

Matias Koskela, Timo Viitanen, Pekka Jääskeläinen, and Jarmo Takala. 2016. Foveated Path Tracing. In *Proceedings of the International Symposium on Visual Computing.*

The community of LuxRender. 2013. *LuxRender Documentation: Refine Brush.* Available: http://www.luxrender.net/wiki/Refine_Brush, Referenced: 26th of May 2017.

Andrew MacQuarrie and Anthony Steed. 2017. Cinematic Virtual Reality: Evaluating the Effect of Display Type on the Viewing Experience for Panoramic Video. In *Proceedings of the IEEE Virtual Reality.*

Hunter Murphy, Andrew Duchowski, and Richard Tyrrell. 2009. Hybrid Image/Model-Based Gaze-Contingent Rendering. *ACM Transactions on Applied Perception* 5, 4 (2009).

Matt Pharr and Greg Humphreys. 2010. *Physically Based Rendering: From Theory to Implementation* (2nd ed.). Morgan Kaufmann.

Pixar. 2017. *RenderMan 20 Documentation: Rendering Efficiently.* Available: https://renderman.pixar.com/resources/RenderMan_20/tutorialRenderingEfficiently.html, Referenced: 26th of May 2017.

Daniel Pohl, Xucong Zhang, and Andreas Bulling. 2016. Combining Eye Tracking with Optimizations for Lens Astigmatism in Modern Wide-Angle HMDs. In *Proceedings of the Virtual Reality.*

Martin Reddy. 2001. Perceptually Optimized 3D Graphics. *IEEE computer Graphics and Applications* 21, 5 (2001).

Thorsten Roth, Martin Weier, Jens Maiero, André Hinkenjann, and Yongmin Li. 2015. Guided High-Quality Rendering. In *Proceedings of the International Symposium on Visual Computing.*

Takashi Shibata. 2002. Head mounted display. *Displays* 23, 1–2 (2002).

Michael Stengel, Steve Grogorick, Martin Eisemann, and Marcus Magnor. 2016. Adaptive Image-Space Sampling for Gaze-Contingent Real-time Rendering. *Computer Graphics Forum* 35, 4 (2016).

Karthik Vaidyanathan, Marco Salvi, Robert Toth, Tim Foley, Tomas Akenine-Möller, Jim Nilsson, Jacob Munkberg, Jon Hasselgren, Masamichi Sugihara, Petrik Clarberg, et al. 2014. Coarse Pixel Shading. In *Proceedings of High Performance Graphics.*

Martin Weier, Thorsten Roth, Ernst Kruijff, André Hinkenjann, Arsène Pérard-Gayot, Philipp Slusallek, and Yongmin Li. 2016. Foveated Real-Time Ray Tracing for Head-Mounted Displays. 35, 7 (2016).

Martin Weier, Michael Stengel, Thorsten Roth, Piotr Didyk, Elmar Eisemann, Martin Eisemann, Steve Grogorick, André Hinkenjann, Ernst Kruijff, Marcus Magnor, Karol Myszkowski, and Philipp Slusallek. 2017. Perception-driven Accelerated Rendering. *Computer Graphics Forum* 36, 2 (2017).

Eric Weisstein. 2002. Lambert W-function. Online. (2002). Available: http://mathworld.wolfram.com/LambertW-Function.html, Referenced: 1st of June 2017.