

Rectangular Mapping-based Foveated Rendering

Giannan Ye*

DALAB, Shanghai Jiao Tong University

Yunchuan Li§

University of Maryland, College Park

Anqi Xie†

DALAB, Shanghai Jiao Tong University

Xubo Yang¶

DALAB, Shanghai Jiao Tong University

Susmija Jabbireddy‡

University of Maryland, College Park

Xiaoxu Meng||

Tencent Games Digital Content Technology Center



Figure 1: Left: Comparison of the proposed foveated rendering (upper-left) and the full-resolution rendering (lower-right). Right: As illustrated in the close-ups of the rendered images, compared with hyperbolic fall-off [9], linear fall-off [35], and log-polar fall-off with kernel functions [22], our method generally preserves sharper details on the peripheral regions with the same rendering budget.

ABSTRACT

With the speedy increase of display resolution and the demand for interactive frame rate, rendering acceleration is becoming more critical for a wide range of virtual reality applications. Foveated rendering addresses this challenge by rendering with a non-uniform resolution for the display. Motivated by the non-linear optical lens equation, we present rectangular mapping-based foveated rendering (RMFR), a simple yet effective implementation of foveated rendering framework. RMFR supports varying level of foveation according to the eccentricity and the scene complexity. Compared with traditional foveated rendering methods, rectangular mapping-based foveated rendering provides a superior level of perceived visual quality while consuming minimal rendering cost.

Index Terms: Computing methodologies—Computer graphics—Rendering—Visibility; Human-centered computing—Visualization—Visualization design and evaluation methods

* e-mail: wsyhdyjn@sjtu.edu.cn

† e-mail: samsara@sjtu.edu.cn

‡ e-mail: jsreddy@umd.edu

§ e-mail: yli93@terpmail.umd.edu

¶ e-mail: yangxubo@sjtu.edu.cn (corresponding author)

|| e-mail: xiaoxumeng@tencent.com (corresponding author)

1 INTRODUCTION

Virtual reality provides an immersive experience for users and has been applied to a variety of domains such as education, collaborative design, and entertainment. However, delivering a high-quality visual experience in real-time requires gigantic computational resources beyond the recent improvements in the hardware and software systems for rendering, which prevents VR from being accepted by the general public.

Foveated rendering manages a spatially varying pixel distribution for the display in virtual reality and accelerates the rendering process by leveraging the ocular limitations [37]. With the support of the eye-tracking system, a foveated renderer dynamically renders the foveal vision with higher quality and the peripheral vision with lower quality, which makes space for the improvement of rendering speed in 3D graphics.

The traditional image-mapping-based foveation system simplifies the implementation of the foveated rendering system with a two-pass pipeline, which naturally supports gradually-changing resolution from the foveal region to the peripheral region. However, traditional approaches such as the log-polar mapping method [22] and the visual-polar mapping method [13] generate curved coarse pixels in the peripheral region and obvious artifacts that are difficult to mitigate in post-processing. Moreover, traditional image-mapping-based foveated rendering methods cannot fully utilize the space in the frame buffer.

In this work, we propose the rectangular mapping-based foveated rendering (RMFR) based on the deferred shading pipeline [4]. First, we apply the rectangular mapping image transformation to the textures in the geometry buffer and save the transformed metadata into a transformed geometry buffer. Second, we calculate the lighting

with the information stored in the transformed geometry buffer and render the result to a reduced resolution buffer. Finally, we apply the inverse rectangular mapping to the reduced resolution buffer and render to the full-resolution screen.

Rectangular-based foveation system solves the problem of curved artifacts by keeping the pixels rectangular during mapping and processing the horizontal and vertical fields independently, which significantly improves the visual effect of the rectangular mapping-based foveation system. Moreover, the tolerances of content compression along the horizontal and vertical field-of-view (FoV) are different for most VR frames, decoupling the horizontal and vertical fields brings more flexibility to the fine-tuning of the shading rates separately.

To demonstrate the effectiveness of our proposed pipeline, we evaluate the image quality of RMFR against the state-of-the-art foveated rendering methods with objectively and subjectively. For objective evaluation, we apply multiple evaluation metrics, such as foveated peak signal-to-noise ratio (FPSNR), FovVideoVDP [19] and structural similarity (SSIM) to quantify the RMFR quality. For subjective evaluation, we conduct user studies to show the proposed method has the best average performance.

In summary, our contributions include

- designing the rectangular mapping-based foveated rendering to enable a parameterized trade-off between rendering quality and rendering speed for foveated rendering;
- providing maximized perceptual quality guarantee with adjustable decoupling foveation parameters;
- achieving minimal perceived loss of detail with 14.8% shading samples by mapping the rectangular-transformation onto the GPU.

We organize the rest of the paper as follows: Section 2 summarizes the development of foveated rendering. The proposed RMFR technique and the its implementations are presented in Section 3. Section 4 describes the user study. A comparison with state-of-the-art using multiple evaluation metrics is presented in Section 5. We discuss the advantages and limitations of the proposed framework in Section 6. Finally, we conclude in Section 7.

2 RELATED WORK

Foveated rendering systems vary resolution or sampling rate across the visual field based on visual factors such as visual acuity, luminance, and contrast. Various foveation systems have been used to approximate the human visual field. We discuss the similarities and differences among the different distributions.

Hyperbolic model. Guenter *et al.* [9] propose a visual acuity model which models the acuity as a function of the reciprocal of the eccentricity. This hyperbolic model serves as a good approximation for visual acuity at low eccentricities, after which the acuity drops more steeply. They render three image layers around the gaze point at progressively higher eccentricity with lower sampling rate. The three rendered layers with discrete sampling rates are combined with bilinear upsampling. However, this method achieves the foveation by repeatedly rendering the fovea region for 3 times, which lower the efficiency of the rendering pipeline. Stengel *et al.* [30] extend the hyperbolic model to incorporate the effect of smooth-pursuit eye movements by linearly extending the foveal region according to the gaze motion vector. To compensate for the missing pixels caused by sparsely distributed shading samples on the periphery, pull-push [8] interpolation is applied to create the full foveated image.

Inspired by the coarse pixel shading technique [33], Patney *et al.* [26] decouple shading and visibility by quantizing shading rates to a finite set of screen-aligned grids and implement foveated rendering by sampling coarse pixels in the peripheral vision. They

address temporal artifacts in foveated rendering by pre-filtering and temporal anti-aliasing. Since human eyes are sensitive to edges with high-contrast, contrast preservation is added to reduce the tunneling effect. Swafford *et al.* [31] employ foveally selective ray casting for 360° immersive virtual reality content, which renders with a variant of multi-layer relief mapping [27]. Deng *et al.* [5] incorporate the human psychophysics of visual- and stereo-acuity into an egocentric neural representation of 3D scenery. To achieve perceptually high-quality immersive interaction, they jointly optimize the latency/performance and visual quality, while mutually bridging human perception and neural scene synthesis. Polychronakis *et al.* [28] optimize the three-layer hyperbolic model [9] by devising a piecewise probability model and applying it to foveated path tracing.

The hyperbolic model has proven to be accurate for the fovea region with small eccentricity [9, 30]. However, the region beyond 30° eccentricity is rendered at a uniform, lower resolution, which limits speedup potential through foveated rendering.

Polar-Space model. The excitation of the cortex can be approximated by a log-polar mapping of the eye’s retinal image [3]. Foveation effects could be achieved by mapping an image from the Cartesian space to the log-polar space and project the image back to the Cartesian space [2]. Meng *et al.* [22] present kernel foveated rendering (KFR), which parameterizes foveated rendering by embedding polynomial kernel functions in the classic log-polar mapping. Meng *et al.* implement a two-pass foveated rendering pipeline with deferred shading. In the first pass, they perform the kernel log-polar transformation for the G-buffer and render the result to a reduced resolution buffer. In the second pass, they conduct the inverse kernel log-polar transformation for the reduced resolution buffer and render to the full resolution screen. Koskela *et al.* [13] vary the resolution smoothly by mapping the visual field to the polar space and rendering uniformly in the polar space. Ray tracing methods naturally allow for smooth non-uniform sampling in the screen space. Liu *et al.* [18] propose a foveated rendering system that considers additional visual factors including geometry, combined depth of field (DOF)-retinal blur, and longitudinal chromatic aberration [10]. They convert the coordinates from screen space to polar space with a scaling strategy and apply a stochastic sampling based on the DOF.

In this work, we propose the rectangular mapping-based foveated rendering based on the deferred shading pipeline. By decoupling the horizontal and vertical fields, we bring more flexibility to the fine-tuning of shading rates separately and avoid the curved artifacts introduced in the logarithmic image transformation.

Linear model. Weier *et al.* [36] propose varying the acuity linearly with the increase of eccentricity. By modeling a linear fall-off foveation into the ray-tracing pipeline, the sampling probability reduces as the distance from the gaze point increases. Weier *et al.* [35] use depth-of-field information to design a post-process anti-aliasing technique. The depth-of-field effect that occurs when focusing on objects can be used as a low-pass filter to minimize the high-frequency artifacts in the peripheral region.

Other models. Friston *et al.* [7] formulate a general pixel or ray density function and propose a rasterization pipeline that transforms images with customized rolling functions. Our method falls in this line of research, but focuses on the decoupling of axes. Turson *et al.* [32] study the resolution requirements at different eccentricities as a function of luminance patterns and derive a low-cost predictor of the foveated rendering parameters. By analyzing the local luminance contrast of the image, the computational cost of foveated rendering improves significantly. Walton *et al.* [34] propose a real-time method to compute ventral metamers for the foveated rendering of near-eye displays. First, the system analyzes the input image to extract statistics. Then, it computes a feature pyramid comprising of steerable filter responses on multiple scales and synthesizes a new image by collapsing this pyramid.

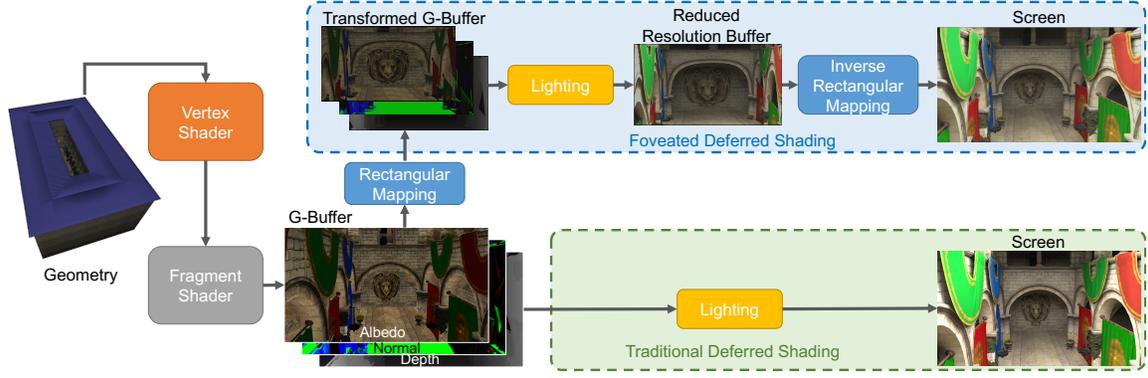


Figure 2: The pipeline of the rectangular mapping-based foveated rendering. We implement the foveated rendering with deferred shading. Instead of shading with the G-buffer in Cartesian coordinates, we apply the rectangular mapping image transformation for the G-buffer and save the transformed metadata into the transformed geometry buffer (TG-Buffer), we calculate the shading with the TG-Buffer and render to the reduced resolution buffer (RS-Buffer). Finally, we calculate the inverse rectangular mapping of the RS-Buffer and render to the full-resolution screen.

Multi-Res Shading [24] takes the advantage of the lens distortion property and splits the image into multiple viewports (e.g., 3×3 grid) with reduced sizes for the periphery to avoid over-shading the edges of the image. Compared with Multi-Res Shading, our method does not require customized hardware support and allows for continuous non-uniform compression.

Variable Rate Shading [25] increases the rendering performance and quality by varying the shading rate for different regions of the frame. A scene can be shaded with a mixture of rates, including non-square footprints for foveation. Our method avoids specifying the shading rate spatially and achieves a higher acceleration by shading with coarser pixels on the periphery.

3 OUR APPROACH

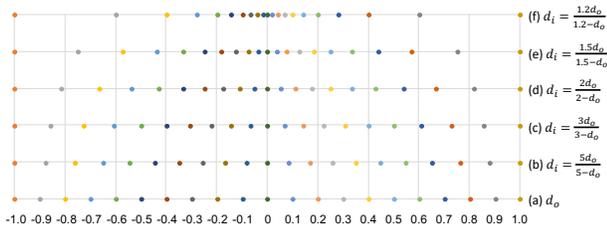


Figure 3: One-dimensional sampling map with different sampling equations. (a) d_o is the original uniform sampling in $[-1, 1]$; (b) - (f) $d_i = \frac{fd_o}{f - \text{abs}(d_o)}$ with $f = 5, f = 3, f = 2, f = 1.5$ and $f = 1.2$, respectively.

3.1 Re-distribution of Samples in a 1-D Space

Before introducing two-dimensional rectangular mapping-based image transformation, we start with the idea of shifting pixel distribution in one-dimensional space. Consider a spatial object point imaged by a simple converging lens of focal length f . The object distance and image distance from the center of the lens share a non-linear relationship, given by the original thin lens equation. Image distance d_i increases with object distance d_o ; however, the changing rate of d_i is a non-linear function of both f and d_o as shown in Equation (2).

Foveated rendering methods, in general, consider a non-uniform distribution of rendering samples, as opposed to a uniform distribution in traditional rendering methods. We propose to leverage the thin lens equation to obtain the non-uniform sample positions. We

treat the original uniform rendering samples as the objects located at d_o (shown in Figure 3 (a)). The new sample points, which are treated as the images at d_i , can be calculated through Equation (2) under different f (shown in Figure 3 (b)-(f))

$$\frac{1}{d_o} + \frac{1}{d_i} = \frac{1}{f}, \quad (1)$$

$$d_i = \frac{fd_o}{d_o - f}. \quad (2)$$

In this paper, we consider the case where $d_o \in (0, f)$ to ensure the continuity and monotonicity of the mapping function. For mathematical convenience, we rewrite Equation (2) to Equation (3), as it is more reasonable to map the negative samples to the negative samples in rendering

$$d_i = \frac{fd_o}{f - \text{abs}(d_o)}. \quad (3)$$

As demonstrated in Figure 3 (b)-(f), the distribution of d_i focuses more around the origin as f decreases. Suppose the fovea, left and right edges of the display respectively locates at the origin, $+1$ and -1 , foveated effect can be achieved by distributing more samples around the origin and fewer samples in the periphery, followed by the inverse function to map samples back to regular space.

3.2 Rectangular Mapping Foveated Rendering Framework

Now we extend this idea to the two-dimensional Cartesian coordinates and introduce the complete rectangular mapping-based foveated rendering framework.

Overview As shown in Figure 2, our algorithm could be implemented with the deferred shading pipeline, which performs a lighting pass after generating the geometry buffers. The vertex shader step and the fragment shader step remain the same for the traditional deferred shading pipeline and the foveated deferred shading pipeline. In our design, we modify the lighting pass by introducing the rectangular mapping and inverse rectangular mapping steps.

Rectangular-mapping of the Geometry Buffer For the full-resolution G-buffer of resolution $W \times H$, we apply the rectangular mapping image transformation and save the transformed metadata into the transformed geometry buffer (TG-Buffer) of resolution $w \times h$. Given a foveal point (\hat{x}, \hat{y}) , each pixel in screen space with Cartesian coordinates (x, y) is mapped to (x', y') , where

$$x' = x - \hat{x}, \quad y' = y - \hat{y}.$$

For simplicity, we only list mapping equations for pixels with $x' > 0$ and $y' > 0$. Rectangular mapping transforms the coordinates (x', y') in the G-buffer to (u, v) in the TG-buffer space via Equation (4)

$$\begin{cases} u = N_X\left(\frac{f_x \cdot x'}{f_x + \text{abs}(x')}\right) \cdot w \cdot \left(1 - \frac{x}{W}\right) + \frac{x}{W} \cdot w, \\ v = N_Y\left(\frac{f_y \cdot y'}{f_y + \text{abs}(y')}\right) \cdot h \cdot \left(1 - \frac{y}{H}\right) + \frac{y}{H} \cdot h, \end{cases} \quad (4)$$

where f_x and f_y respectively control the pixel distributions in the horizontal and the vertical axes, and $N_X(\cdot)$ and $N_Y(\cdot)$ are normalization functions for the X- and the Y-axis defined in Equation (5)

$$N_X(x) = \frac{x}{\frac{f_x \cdot (W - \bar{x})}{f_x + (W - \bar{x})}}, \quad N_Y(y) = \frac{y}{\frac{f_y \cdot (H - \bar{y})}{f_y + (H - \bar{y})}}. \quad (5)$$

Equations for pixels with negative coordinates can be derived similarly.

Shading in the Reduced-resolution Buffer In the lighting pass, a pixel shader computes the direct and indirect lighting at each pixel using the information from TG-Buffer and renders the result to a reduced-resolution buffer (RS-Buffer). Denote the ratio between the width (height) of the screen and the width (height) of the RS-Buffer and TG-Buffer by $\sigma = \frac{W}{w} = \frac{H}{h}$. The shading cost is linear to the number of pixels to be shaded [13, 22]. The traditional deferred shading pipeline shades $W \times H$ pixels, while the RMFR pipeline shades $w \times h$ pixels. As a result, the theoretical speedup S of RMFR in shading step is

$$S = \frac{W \times H}{w \times h} = \sigma^2.$$

Rendering to the Screen In the final pass, the renderer maps the rendered scene in the RS-Buffer to the full-resolution screen, which is exactly the inverse of the aforementioned process. Specifically, a pixel with coordinates (u, v) in the RS-Buffer is transformed back to (x'', y'') in Cartesian coordinates by Equation (6)

$$x'' = \frac{f_x \cdot N_U(u)}{f_x - N_U(u)}, \quad y'' = \frac{f_y \cdot N_V(v)}{f_y - N_V(v)}, \quad (6)$$

where $N_U(\cdot)$ and $N_V(\cdot)$ are defined in Equation (7)

$$\begin{cases} N_U(u) = \frac{u - \frac{\bar{x}}{W} \cdot w}{w \cdot \left(1 - \frac{\bar{x}}{W}\right)} \cdot \frac{f_x \cdot (W - \bar{x})}{f_x + (W - \bar{x})}, \\ N_V(v) = \frac{v - \frac{\bar{y}}{H} \cdot h}{h \cdot \left(1 - \frac{\bar{y}}{H}\right)} \cdot \frac{f_y \cdot (H - \bar{y})}{f_y + (H - \bar{y})}. \end{cases} \quad (7)$$

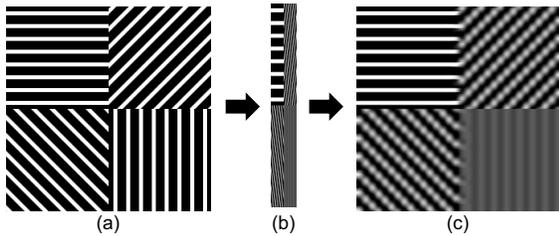


Figure 4: (a) A 256×256 image with horizontal pattern and vertical pattern; (b) the 256×32 compressed image of (a) with compression only in the X-axis; (c) the 256×256 image recovered from (b). The lower-right part with vertical pattern in the compression in the X-axis, while the upper-left part with horizontal pattern is not.

3.3 Advantages of RMFR

3.3.1 Improving Rendering Quality by Decoupling

Images with more vertical patterns are found to be sensitive to compression along the X-axis as shown in Figure 4, while those with a more horizontal pattern are sensitive to compression along the Y-axis. We can take the advantage of this fact by allowing for a greater X-axis foveation for the images with more horizontal patterns and a greater Y-axis foveation for images with more vertical patterns.

We consider a foveated rendering as a non-uniform compression of a full-resolution rendering. State-of-the-art polar-space foveated rendering methods [9, 22, 36] and warping methods [7, 34] process the pixels according to the distance to the gaze point; therefore these methods are unable to adjust the level of compression for each axis independently. In contrast, RMFR is able to process the horizontal and vertical FoV at different foveation levels. By decoupling the horizontal and vertical fields, RMFR allows more compression in a specific axis.

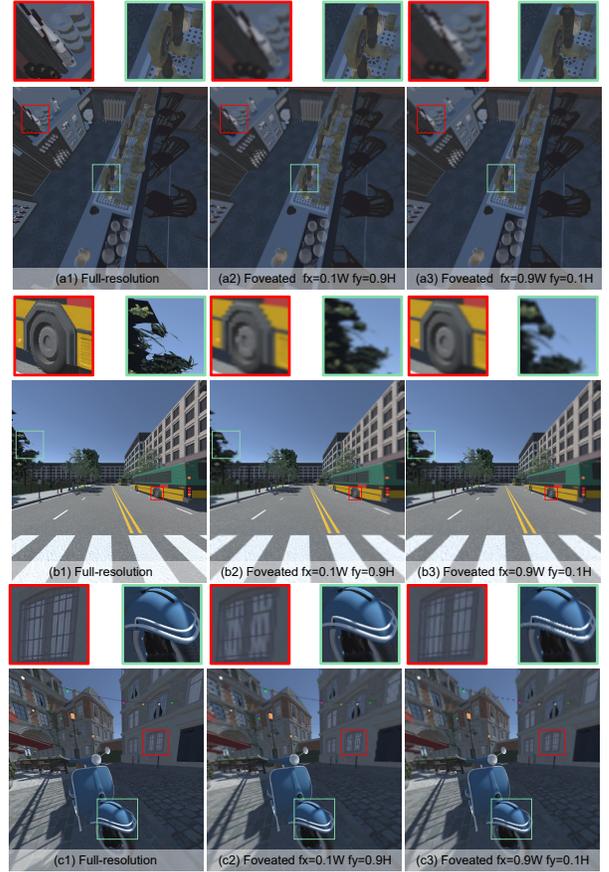


Figure 5: The comparison among different choices of f_x and f_y . The image in the first row achieves high quality with smaller f_x and larger f_y , indicating that the image allows more compression in the X-axis. In contrast, the image in the second row allows more compression in the Y-axis. For the image in the third row, the zoomed-in view with red outline achieves high quality with larger f_x and the zoomed-in view with green outline achieves high quality with larger f_y , indicating that the image has no axis preference.

We measure Foveated Peak Signal-to-Noise Ratio (FPSNR) [14, 29], Peak Signal-to-Noise Ratio (PSNR) and Structural Similarity (SSIM) for the foveated rendering images, as shown in Figure 5. The images are rendered with compression parameter $\sigma = 2.2$ at a resolution of 1440×1600 . The results are shown in Figure 6.

For each scene, there exist parameters f_x^* and f_y^* with the highest FPSNR. When $f_x < f_x^*$, more pixels are distributed from the foveal region to the peripheral region as f_x increases. Therefore, peripheral quality in the X-axis improves and FPSNR increases with f_x , whereas when $f_x \geq f_x^*$, FPSNR decreases with f_x since the number of samples around the fovea cannot meet the requirement of high fovea quality. A similar pattern is also observed in f_y .

As shown in the first row in Figure 5 (a), we have $f_x^* = 0.2W$ and $f_y^* = 0.3H$, which allows more compression in the X-axis. For the image in the second row (b), we have $f_x^* = 0.3W$ and $f_y^* = 0.2H$, which allows more compression in the Y-axis. For the image in the third row (c), we have $f_x^* = 0.2W$ and $f_y^* = 0.2H$, indicating no preference for either axis. Images such as (a) and (b) are said to be “axis-preferred.”

For the rendered images, especially for the rendered images with “axis-preference,” we bring more flexibility to the fine-tuning of the shading rates separately and further improve the rendering quality by choosing proper f_x and f_y . The improvement in peripheral quality mitigates the effect of cybersickness [17].

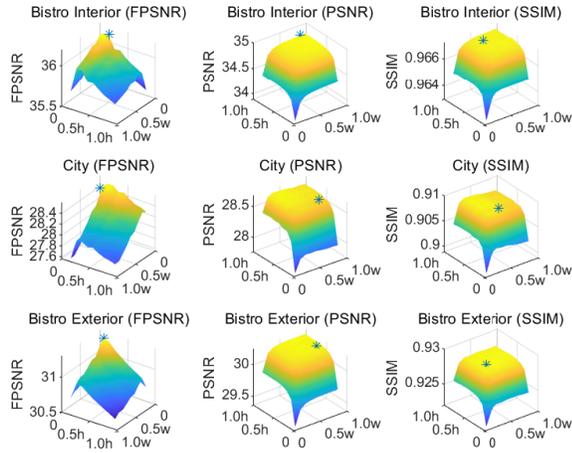


Figure 6: FPSNR, PSNR and SSIM plots of image (a), (b) and (c) shown in Figure 5 for rectangular mapping-based foveated rendering with $W = 1440$, $H = 1600$, $\sigma = 2.2$, $0 \leq f_x \leq W$ and $0 \leq f_y \leq H$.

3.3.2 Robustness in Anti-aliasing

RMFR is more friendly to traditional anti-aliasing methods. In the traditional log-polar mapping foveation system [22] and the visual-polar mapping foveation system [13], the U-value of the projected pixel position in the TG-Buffer is related to the distance between the pixel position in the G-buffer and the fovea position. As shown in Figure 7, log-polar foveated rendering (KFR) generates aliasing-free result only for concentric circles pattern centered at the fovea, which perfectly matches its mapping mechanism. For all the other patterns, the curved pixel mapping of KFR leads to curved artifacts, which causes obvious flashing effect that is difficult to mitigate. Traditional anti-aliasing methods that use neighborhood information could not handle the curved artifacts well with limited kernel size. Instead, rectangular mapping based foveation system keeps the pixels rectangular during mapping and processes the X-axis and the Y-axis in the G-Buffer independently and makes it possible to process the artifacts with the state-of-the-art anti-aliasing approaches.

3.3.3 Utilizing the RS-Buffer Completely

As shown in Figure 7 (b), traditional polar-mapping foveation systems do not completely utilize the space in the RS-Buffer. By

adopting rectangular mapping with decoupled X-axis and Y-axis, RMFR could use all the pixels in the RS-Buffer.

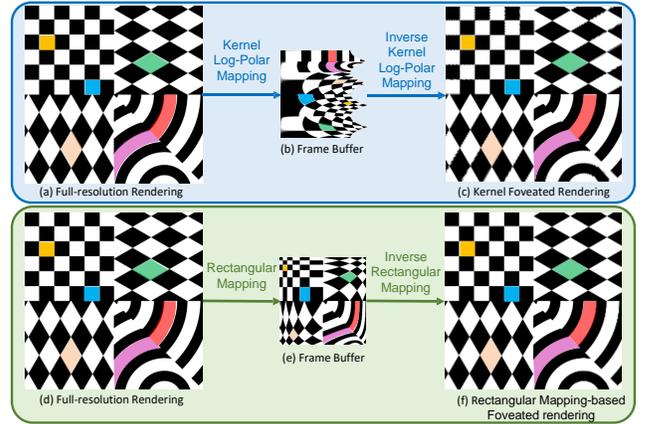


Figure 7: Comparison of the artifacts between log-polar foveated rendering (KFR) and rectangular mapping-based foveated rendering (RMFR). Log-polar foveated rendering (upper) generates curved artifacts in (c) that are difficult to mitigate. By decoupling the X-axis and Y-axis, RMFR (lower) shows a smoother foveation effect in (f). Examples of the correspondence regions are marked in different colors. The frame buffers in the middle are of the same size. For KFR, the parameters are set to $\sigma = 2.6$, $\alpha = 4$. For RMFR, we set $\sigma = 2.6$ and $f_x = 0.38W$, $f_y = 0.38H$.

In the next section, we describe user studies to find the optimized parameters and validate that the images rendered with the proposed algorithm are perceptually identical to the full-resolution rendering.

4 USER STUDY

4.1 Apparatus

Our user study apparatus consists of a personal computer with an Nvidia GTX 980 graphics card, an Intel Core i7-4790K CPU and an HTC Vive Pro Eye head-mounted display. The Vive headset is integrated with a 120 Hz infrared eye-tracking system and a 2880×1600 resolution screen (1440×1600 per eye). We use an XBOX controller for the interaction between the participant and the user study system. The user study took place in a quiet room.

As shown in Figure 8, the computer-generated environments used for the user study consist of 3 scenes (Test Scenes (a), (b) and (c)) from the *Amazon Lumberyard Bistro* [1] and *Emerald City Square* [23] and 3 intentionally selected and modified corners (Test Scenes (d), (e) and (f)) from above scenes to highlight certain “axis-preference.” These scenes are rendered with the Unity game engine.

To ensure that the participants are familiar with the user study system, we requested the participants to complete all the tasks as a trial run and familiarize themselves fully with the interaction before the formal tests.

4.2 Participants

12 (9 males and 3 females) participants were recruited for this study. The participants aged from 21 to 32 ($M=23.9$, $STD=2.9$) with normal or corrected-to-normal vision. The participants had a variety of education and professional background. None of the participants was involved in this project prior to the user study.

4.3 Experimental Design and Task

4.3.1 Pilot Study: Estimation of the Sampling Distribution Parameters

We conduct a pilot study to obtain a coarse estimate of the optimal sampling distribution parameters f_x^* and f_y^* under human visual per-

ception, in which participants score the quality of foveated rendering with different parameters f_x and f_y in a random sequence. In this experiment, the compression parameter σ ranges from 2.2 to 2.8 with a step size of 0.3; and the sampling distribution parameters f_x and f_y range from $0.05W(H)$ to $0.8W(H)$ with a step size of 0.25. We conduct the test on 6 scenes with 48 trials each, 288 tests in total.

For scene m with σ_i , we present the participant two frames: (1) the full-resolution rendering; and (2) the foveated rendering with (σ_i, f_x, f_y) , where f_x, f_y are selected from the shuffled parameter array. The two frames are presented in a random order for 2 seconds each and are separated by a black-screen interval of 0.75 second. Then, we ask the participant to score the difference between the two frames they observed. The score $S(m, \sigma_i, f_x, f_y)$ contains three confidence levels: 3 represents *perceptually identical*; 2 represents *minimal perceptual difference*; and 1 represents *significant perceptual difference*. Average score is computed for all the foveation parameter σ for scene m .

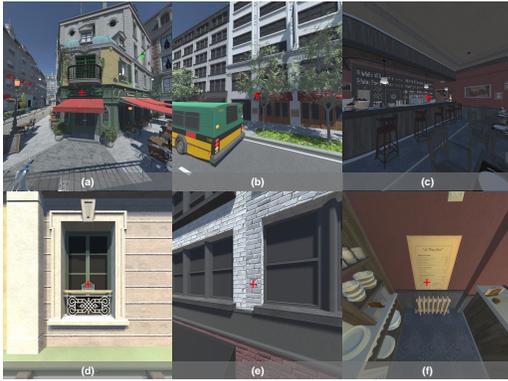


Figure 8: The computer-generated environments used for user study. The red crosses show the gaze points for each scene.

4.3.2 Main Study: Estimation of Compression Parameter

In the main experiment, participants score the quality of foveated rendering with different parameters σ in a random sequence. For scene m , we present the participant with two frames: (1) the full-resolution rendering; and (2) the foveated rendering with (σ, f_x^*, f_y^*) , where σ is selected from the shuffled parameter ranging from 1.4 to 2.8 with a step size of 0.2, and (f_x^*, f_y^*) is the best parameter combination inferred from the result of Section 4.3.1.

The two frames are presented in a random order and participants are asked to score the difference analogously to that in Section 4.3.1. When the process is finished, we calculate the average score of all the trials for scene m with foveation parameter σ , and compute the average score for each single σ . The test is conducted on 6 scenes with 8 trials each, 48 tests in total.

4.3.3 Comparison Study: Comparison between the Proposed Method and Other Methods

To demonstrate the strength of the proposed method, we conduct a *two-alternative-force-choice* (2AFC) experiment. In each trial, we present the participants with a random pair of frames, one of which is generated by the proposed method and the other is generated by a different foveated rendering method or full resolution ground truth. The two frames are presented in a random order in a similar way to that in Section 4.3.1. The participants are asked to vote for the image with better visual quality in each round. At the end of the experiment, we analyze the choices of all the trials to assess the preference of the proposed method.

In our experiment, three other methods and the full-resolution ground truth are compared with the proposed method. We conduct

the test on 6 scenes with 4 trials each (compare the proposed method with three other methods and the full-resolution ground truth), 24 tests in total.

4.3.4 Controlling for Lack of Attention and Exhaustion

In our main study, we adapt a similar checking of attention and exhaustion as that in Meng *et al.* [21]. 30% of the trials are selected as validation trials in the main study to ensure the validity of the data. In the comparison between the tested images and the reference images, we presented the participants with identical full-resolution images as a validation. If the participant declared these validation trials as not identical ($score \neq 3$), we would ask the participant to pause and take a break for at least 30 seconds, and then continue the user study. Meanwhile, we would record this choice as an *error*. If $error \geq 3$ in the random tests, we would terminate the user study and discard the data of this participant.

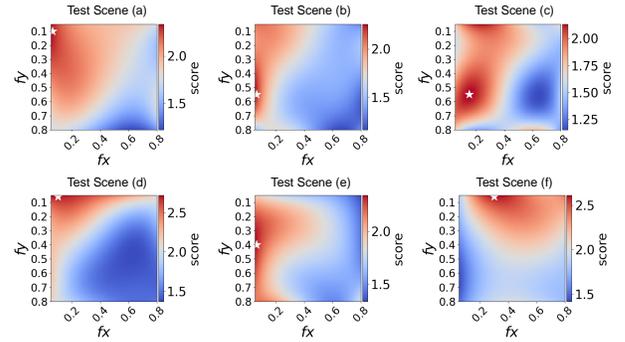


Figure 9: Averaged perceptual score as a function of f_x and f_y for each test scene in Figure 8, where white stars indicate f_x^* and f_y^* . f_x and f_y range from $0.0H$ (W) to $0.8H$ (W). Red indicates high perceptual score while blue indicates low. We find consensus “axis-preference” among most participants.

4.4 Results

4.4.1 Estimation of the Sampling Distribution Parameter

After the pilot study, for every participant k , we average the perceptual scores $S_k(m, \sigma_i, f_x, f_y)$ across σ_i and denote the results by $\bar{S}_k(m, f_x, f_y)$. We find similarity and consensus preferred “hotspots” among most participants, so we further average $\bar{S}_k(m, f_x, f_y)$ across participants (denoted by $\bar{S}(m, f_x, f_y)$). The smoothed $\bar{S}(m, f_x, f_y)$ of each scene across the f_x and f_y grid is shown in Figure 9, which suggests that different scenes have different hotspots for f_x and f_y .

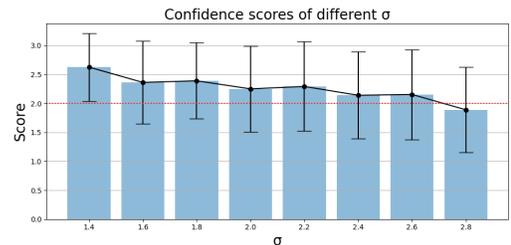


Figure 10: The averaged score for each σ across all participants and scenes, where the red dotted line indicates the score threshold of 2.0. The averaged score decreases with σ as expected.

We provide a short remark on the determination of f_x^* and f_y^* for each test scene. For test scenes (b), (c) and (e) as shown in Figure 9,

the optimal f_x^* is small and f_y^* is relatively large. The result matches our experimental results in Section 3.3 that the images with more flat tilted lines tolerate horizontal compression and allows for more compression in the periphery along the X-axis with small f_x^* . On the Y-axis, f_y^* is relatively large as small f_y may cause obvious artifacts. For test scene (f), the tilted lines are steep, which implies a smaller f_y^* as opposed to the previous situation. For test scenes (a) and (d), the high-frequency details are primarily distributed in the foveal region, and the scene complexity of the periphery region is low, which leads to small f_x^* and f_y^* in Figure 9 and more valid shading effort in the central region.

4.4.2 Estimation of the Compression Parameter σ

We calculate the average score for each σ across all participants and scenes to estimate the overall optimal σ that is applicable to more general cases, which is shown in in Figure 10. Since the sizes of the TG-Buffer and the RS-Buffer becomes small with a larger σ , the number of pixels to be shaded also decreases. As a result, the visual quality of the final rendering results degrades as σ increases. To achieve the maximum acceleration rate with acceptable visual quality, we set a score threshold of 2, at which the users consider the quality level of the rendering results marginally different from the ground truth. We find that $\sigma = 2.6$ is the largest σ that keeps the score above the threshold.

5 COMPARISON

5.1 Evaluation Methods and Error Metrics

Since our approach aims at real-time foveated rendering, we focus on comparing the proposed method against existing real-time foveated rendering methods. Specifically, we report the results from the hyperbolic fall-off foveation system [9], linear fall-off foveation system [36], log polar-mapping foveation system with polynomial kernel functions [22], and the proposed rectangular mapping-based foveation system. To conduct an apple-to-apple comparison, we tuned the parameters of the state-of-the-art methods to fix the theoretical speedup across all methods in the following experiments. Then we compare the visual quality of different foveation models.

For the linear fall-off foveation system [36], we utilize the linear resolution fall-off equation with parameters proposed by the authors, i.e., set the eccentricity threshold for the fovea region $r_1 = 10^\circ$, the eccentricity threshold for the middle region $r_2 = 20^\circ$, and the minimum sampling rate $p_{min} = 0.047$.

The parameters for the original hyperbolic-off foveation system [9] are determined by a user study with desktop setups. Since we care more about the VR setups, we adopt the same eccentricity threshold as that in the linear fall-off foveation system [36], and apply the resolution parameters proposed by the authors, i.e., set sampling factor (pixel size) for the inner layer $s_1 = 1$, sampling factor (pixel size) for the middle layer $s_2 = 2$, and sampling factor (pixel size) for the outer layer $s_3 = 12.35$.

For the log polar-mapping foveation system [22], we apply the parameters proposed by the authors, i.e., set screen-framebuffer-ratio parameter $\sigma = 2.6$ and kernel function parameter $\alpha = 4.0$.

The approximated analytical pixel density is shown in Figure 11, where we plot the resolution factor as a function of distance from the center of the visual field. The mean pixel density is fixed for all methods so the theoretical speedup are the same. For the hyperbolic fall-off model, the inner-, middle-, and outer-layer use three different resolutions and the three layers are blended as the final result. For the linear fall-off model, the inner- and outer-layer use fixed resolutions. The resolution of the middle-layer decreases with the eccentricity. For log-polar fall-off model with polynomial kernel functions, the shading pixels concentrate at the fovea and the pixel density decreases steeply with the increase of eccentricity. For RMFR, we improve the overall visual effect by applying a higher and smoothly-changing pixel density for the peripheral region.

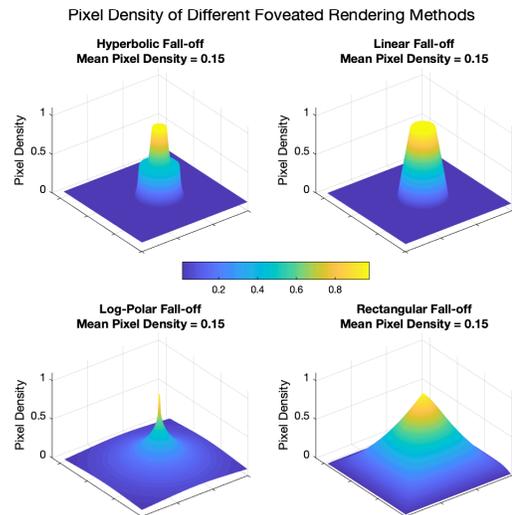


Figure 11: Analytical model of pixel density as a function of distance to the visual center of (a) hyperbolic fall-off; (b) linear fall-off; (c) log-polar fall-off; and (d) rectangular mapping-based foveated rendering (proposed). The X- and Y-axis are the width and height of the image with *aspect ratio* = 1440 : 1600. The Z-axis denotes the pixel density. For RMFR, we keep sufficient samples for the fovea and improve the overall visual effect by applying a higher and smoothly-changing pixel density for the peripheral region.

5.2 Subjective Evaluation

As described in Section 4.3.3, we conduct a 2AFC experiment to compare the proposed RMFR method and other methods. The result is shown in Table 2.

First, we observed close-to-random-guess among trials that compare full resolution ground truth and the proposed RMFR method (48.6% voted for RMFR, $p > 0.05$ in binomial test shows no significant difference). Meanwhile, a significantly higher ratio of voting for the proposed method over other three method was observed. As for comparison with Hyperbolic Fall-off, the vote ratio for RMFR is 68.1% with $p < 0.01(**)$ in binomial test. The preference toward RMFR against Linear Fall-off method is 65.3% with $p < 0.01(**)$ in binomial test. In 69.4% cases, users voted for RMFR against Log-Polar Fall-off with significant difference $p < 0.001(***)$.

5.3 Objective Evaluation

5.3.1 Qualitative Evaluation

We show the visual effect of different foveation methods at 1440×1600 resolution in Figure 1 (for scene *Amazon Lumberyard Bistro* [1]). The images on the upper-left show our visual effect with fovea at the center of the screen, and the images on the lower-right show the full-resolution rendering. Our method generally performs better at preserving the peripheral regions with details, as illustrated in the close-ups on the right of the rendered images. The first zoomed-in views (red) shows the image quality for the fovea regions. All foveated rendering methods render the fovea region with high visual quality. The second and the third zoomed-in views (yellow and blue, respectively) show the image quality of the peripheral regions. The rectangular-based foveated rendering renders the fovea regions with good visual quality and preserves better detail for the peripheral regions. Detailed comparisons of the proposed method with the state-of-the-art are available in the supplementary materials.

Table 1: A comparison of FPSNR (col 2 - col 5), PSNR (col 6 - col 9), SSIM (col 10 - col 13), and FovVideoVDP [20] (col 14 - col 17) to evaluate the hyperbolic fall-off foveation system [9], linear fall-off foveation system [36], log polar-mapping foveation system with polynomial kernel functions [22], and the rectangular mapping-based foveation system (RMFR) implementations on Open Research Content Archive scenes [1, 6, 23]. The averaged value is measured over the 6 scenes. Rectangular mapping-based foveated rendering (proposed) generally provides the best rendering quality.

Metric	FPSNR↑				PSNR↑				SSIM↑				FVVDV↑			
Method	Hyp	Lin	KFR	RMFR	Hyp	Lin	KFR	RMFR	Hyp	Lin	KFR	RMFR	Hyp	Lin	KFR	RMFR
Average	27.36	25.73	25.82	28.89	26.96	25.28	25.99	27.95	0.87	0.86	0.87	0.89	7.37	6.38	7.46	8.12
Bistro Exterior 1	23.69	21.63	22.39	24.99	23.88	21.99	23.29	23.76	0.83	0.82	0.83	0.85	7.64	6.65	7.79	7.76
Bistro Exterior 2	24.43	22.94	23.45	25.17	23.72	22.16	22.66	23.74	0.80	0.78	0.79	0.82	6.65	5.39	6.74	7.25
Bistro Interior 1	33.05	30.53	31.36	35.42	32.90	30.61	31.52	34.21	0.95	0.95	0.95	0.97	8.17	7.35	8.05	8.84
Bistro Interior 2	32.40	30.79	30.30	34.47	32.07	30.34	30.78	34.52	0.95	0.95	0.95	0.97	7.71	6.82	7.80	8.64
City 1	27.35	26.13	25.64	29.33	26.19	25.05	25.60	27.78	0.91	0.90	0.90	0.93	7.43	6.53	7.62	8.29
City 2	23.23	22.36	21.81	23.99	22.98	21.53	22.08	23.68	0.79	0.78	0.78	0.82	6.61	5.57	6.75	7.94

Method	Vote Ratio
RMFR v. Hyperbolic	68.1%
RMFR v. Linear	65.3%
RMFR v. Log-polar	69.4%
RMFR v. Full-Res	48.6%

Table 2: Comparison of the RMFR method and other methods with a 2AFC experiment. Compared with hyperbolic fall-off, linear fall-off, and log-polar fall-off with polynomial kernel functions, participants show preference to our method. We also observed close-to-random-guess among trials that compare full-res rendering and RMFR.

5.3.2 Quantitative Evaluation

We evaluate the average image quality of the whole frame with Peak Signal-to-Noise Ratio (PSNR) and Structural Similarity (SSIM). The foveation quality is assessed by Foveated Peak Signal-to-Noise Ratio (FPSNR) [14, 29] and FovVideoVDP [20].

The FPSNR, PSNR, SSIM and FovVideoVDP values of different foveated rendering systems are shown in Table 1. Compared to the hyperbolic fall off foveation system [9], linear fall-off foveation system [36], and log polar-mapping foveation system [22], the proposed rectangular mapping-based foveated rendering (RMFR) generally provides the best rendering quality.

6 DISCUSSION

6.1 Quality and Speedup

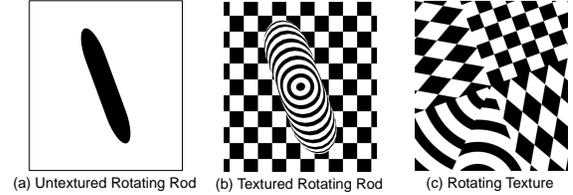
To show that the RMFR is applicable to diversified VR environments with different image patterns, we rotation textured patterns in Figure 12 for 180° with a step size of 10°. We calculate FPSNR and FovVideoVDP scores at each step using the four foveation methods with the same compression scale. RMFR generates the highest score with the optimal parameters at each rotation step. The averaged metrics for each method are shown in Figure 12.

For image-mapping-based foveated rendering framework, the theoretical speedup of lighting S is linear in the squared ratio between the size of the screen and the size of the interior buffer ($S \propto \sigma^2$). We implement the RMFR on an Nvidia GTX980 graphics card with $\sigma = 2.6$ using the deferred shading pipeline. We report the rendering time comparison between the full-resolution rendering and RMFR in Table 3. RMFR reduces the frame time by 45.46%, and achieves a speedup of 1.83× with a 2880 × 1600 resolution.

6.2 Limitations and Future Work

We notice a gap between the f_x^* , f_y^* estimated by the user study (subjective) and those by FPSNR (objective). Since RMFR generally provides better subjective and objective performance as illustrated in Section 5, we envision that RMFR provides better visual experience for the VR users.

We leave the efficient automatic estimation of f_x^* and f_y^* as future work. Further research may investigate integrating the subjective visual acuity with contrast, saliency [11, 15], and LCA [10, 18] to



Test	Metric	RMFR	Log-polar	Hyperbolic	Linear
a	FPSNR	35.90	34.49	30.65	27.99
	FVVDV	9.56	9.49	8.91	8.32
b	FPSNR	23.98	19.33	19.11	15.85
	FVVDV	8.42	7.29	6.88	5.40
c	FPSNR	29.87	26.94	26.26	22.89
	FVVDV	8.90	8.35	8.31	7.42

Figure 12: Top: the snapshots of rotating patterns (a), (b), and (c) used to measure foveation artifacts. Bottom: the average scores of FPSNR and FovVideoVDP for RMFR and other methods with $\sigma = 2.6$ for (a), (b), and (c).

Procedure	Timing (ms)	
	Full-resolution	RMFR(Proposed)
Depth Pass	1.22	1.34
Shadow Pass	5.27	4.98
Defer Pass	4.13	4.85
Skybox	0.08	0.03
Shading/Pass1	14.63	2.04
Pass2	/	0.20
TAA	0.87	0.85
Total GPU Time	26.2	14.29

Table 3: Rendering time comparison between the full-resolution rendering and RMFR at 2880 × 1600 with $\sigma = 2.6$, broken down by steps.

estimate the sampling distribution parameters, and mitigate the gap between the subjective evaluation and the objective evaluation.

7 CONCLUSION

In this paper, we present rectangular mapping-based foveated rendering, which achieves a significant speed-up by rendering the scene in the foveated frame buffer. RMFR outperforms state-of-the-art real-time foveated rendering approaches in visual quality while maintaining minimal rendering cost. Our approach could be easily integrated into the current rasterization rendering pipeline. We also envision that RMFR would be beneficial for streaming networked VR applications [12, 16].

REFERENCES

- [1] Amazon Lumberyard. Amazon Lumberyard Bistro, Open Research Content Archive (ORCA). <http://developer.nvidia.com/orca/amazon-lumberyard-bistro>, July 2017. Accessed: 2021-03-28.
- [2] M. Antonelli, F. D. Igual, F. Ramos, and V. J. Traver. Speeding up the log-polar transform with inexpensive parallel hardware: graphics units and multi-core architectures. *Journal of Real-Time Image Processing*, 10(3):533–550, Sep 2015. doi: 10.1007/s11554-012-0281-6
- [3] H. Araujo and J. M. Dias. An introduction to the log-polar mapping. In *Proceedings II Workshop on Cybernetic Vision*, pp. 139–144, 1996.
- [4] M. Deering, S. Winner, B. Schediwy, C. Duffy, and N. Hunt. The triangle processor and normal vector shader: A vlsi system for high performance graphics. In *Proceedings of the 15th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '88, p. 21–30. Association for Computing Machinery, New York, NY, USA, 1988. doi: 10.1145/54852.378468
- [5] N. Deng, Z. He, J. Ye, P. Chakravarthula, X. Yang, and Q. Sun. Foveated neural radiance fields for real-time and egocentric virtual reality. *arXiv preprint arXiv:2103.16365*, 2021.
- [6] Epic Games. Unreal Engine Sun Temple, Open Research Content Archive (ORCA). <https://developer.nvidia.com/ue4-sun-temple>, October 2017. Accessed: 2021-03-28.
- [7] S. Friston, T. Ritschel, and A. Steed. Perceptual rasterization for head-mounted display image synthesis. *ACM Trans. Graph.*, 38(4), July 2019. doi: 10.1145/3306346.3323033
- [8] S. J. Gortler, R. Grzeszczuk, R. Szeliski, and M. F. Cohen. The lumigraph. In *Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '96, p. 43–54. Association for Computing Machinery, New York, NY, USA, 1996. doi: 10.1145/237170.237200
- [9] B. Guenter, M. Finch, S. Drucker, D. Tan, and J. Snyder. Foveated 3D graphics. *ACM Trans. Graph.*, 31(6):164:1–164:10, Nov. 2012. doi: 10.1145/2366145.2366183
- [10] P. A. Howarth and A. Bradley. The longitudinal chromatic aberration of the human eye, and its correction. *Vision Research*, 26(2):361–366, 1986. doi: 10.1016/0042-6989(86)90034-9
- [11] L. Itti. Automatic foveation for video compression using a neurobiological model of visual attention. *IEEE Transactions on Image Processing*, 13(10):1304–1318, 2004. doi: 10.1109/TIP.2004.834657
- [12] Y. Jin, M. Chen, T. Goodall, A. Patney, and A. C. Bovik. Subjective and objective quality assessment of 2d and 3d foveated video compression in virtual reality. *IEEE Transactions on Image Processing*, 30:5905–5919, 2021. doi: 10.1109/TIP.2021.3087322
- [13] M. Koskela, A. Lotvonen, M. Mäkitalo, P. Kivi, T. Viitanen, and P. Jääskeläinen. Foveated Real-Time Path Tracing in Visual-Polar Space. In T. Boubekeur and P. Sen, eds., *Eurographics Symposium on Rendering - DL-only and Industry Track*. The Eurographics Association, 2019. doi: 10.2312/sr.20191219
- [14] S. Lee, M. S. Pattichis, and A. C. Bovik. Foveated video quality assessment. *IEEE Transactions on Multimedia*, 4(1):129–132, 2002. doi: 10.1109/6046.985561
- [15] W.-T. Lee, H.-I. Chen, M.-S. Chen, I.-C. Shen, and B.-Y. Chen. High-resolution 360 video foveated stitching for real-time vr. In *Computer Graphics Forum*, pp. 115–123. Wiley Online Library, 2017.
- [16] D. Li, R. Du, A. Babu, C. Brumar, and A. Varshney. A Log-Rectilinear Transformation for Foveated 360-Degree Video Streaming. In *IEEE Transactions on Visualization and Computer Graphics*, vol. 27 of *TVCG*, pp. 2638–2647. IEEE, Mar. 2021. doi: 10.1109/TVCG.2021.3067762
- [17] Y.-X. Lin, R. Venkatakrishnan, R. Venkatakrishnan, E. Ebrahimi, W.-C. Lin, and S. V. Babu. How the presence and size of static peripheral blur affects cybersickness in virtual reality. *ACM Trans. Appl. Percept.*, 17(4), Nov. 2020. doi: 10.1145/3419984
- [18] J. Liu, C. Mantel, and S. Forchhammer. Perception-driven hybrid foveated depth of field rendering for head-mounted displays. In *Proceedings of 2021 IEEE International Symposium on Mixed and Augmented Reality*. IEEE, United States, 2021. 2021 IEEE International Symposium on Mixed and Augmented Reality, ISMAR ; Conference date: 04-10-2021 Through 08-10-2021.
- [19] R. K. Mantiuk, G. Denes, A. Chapiro, A. Kaplanyan, G. Rufo, R. Bachy, T. Lian, and A. Patney. Fovvideovdp: A visible difference predictor for wide field-of-view video. *ACM Transactions on Graphics (TOG)*, 40(4):1–19, 2021.
- [20] R. K. Mantiuk, G. Denes, A. Chapiro, A. Kaplanyan, G. Rufo, R. Bachy, T. Lian, and A. Patney. Fovvideovdp: A visible difference predictor for wide field-of-view video. *ACM Trans. Graph.*, 40(4), jul 2021. doi: 10.1145/3450626.3459831
- [21] X. Meng, R. Du, and A. Varshney. Eye-dominance-guided foveated rendering. *IEEE Transactions on Visualization and Computer Graphics*, pp. 1–1, 2020. doi: 10.1109/TVCG.2020.2973442
- [22] X. Meng, R. Du, M. Zwicker, and A. Varshney. Kernel foveated rendering. *Proc. ACM Comput. Graph. Interact. Tech.*, 1(1):5:1–5:20, July 2018. doi: 10.1145/3203199
- [23] K. A. Nicholas Hull and N. Benty. NVIDIA Emerald Square, Open Research Content Archive (ORCA). <http://developer.nvidia.com/orca/nvidia-emerald-square>, July 2017. Accessed: 2021-03-28.
- [24] NVIDIA Corporation. VRWorks - Multi-Res Shading. <https://developer.nvidia.com/vrworks/graphics/multireshading>.
- [25] NVIDIA Corporation. VRWorks - Variable Rate Shading (VRS). <https://developer.nvidia.com/vrworks/graphics/variablerateshading>.
- [26] A. Patney, M. Salvi, J. Kim, A. Kaplanyan, C. Wyman, N. Benty, D. Luebke, and A. Lefohn. Towards foveated rendering for gaze-tracked virtual reality. *ACM Trans. Graph.*, 35(6):179:1–179:12, Nov. 2016. doi: 10.1145/2980179.2980246
- [27] F. Policarpo and M. M. Oliveira. Relief mapping of non-height-field surface details. In *Proceedings of the 2006 symposium on Interactive 3D graphics and games*, pp. 55–62, 2006.
- [28] A. Polychronakis, G. A. Koulteris, and K. Mania. Emulating foveated path tracing. In *Motion, Interaction and Games*, pp. 1–9. 2021.
- [29] S. Rimac-Drlje, M. Vranješ, and D. Žagar. Foveated mean squared error—a novel video quality metric. *Multimedia tools and applications*, 49(3):425–445, 2010.
- [30] M. Stengel, S. Grogorick, M. Eisemann, and M. Magnor. Adaptive image-space sampling for gaze-contingent real-time rendering. *Comput. Graph. Forum*, 35(4):129–139, July 2016. doi: 10.1111/cgf.12956
- [31] N. T. Swafford, J. A. Iglesias-Guitian, C. Koniaris, B. Moon, D. Cosker, and K. Mitchell. User, metric, and computational evaluation of foveated rendering methods. In *Proceedings of the ACM Symposium on Applied Perception*, SAP '16, pp. 7–14. ACM, New York, NY, USA, 2016. doi: 10.1145/2931002.2931011
- [32] O. T. Tursun, E. Arabadzhiyska-Koleva, M. Wernikowski, R. Mantiuk, H.-P. Seidel, K. Myszkowski, and P. Didyk. Luminance-contrast-aware Foveated Rendering. *ACM Trans. Graph.*, 38(4):98:1–98:14, July 2019. doi: 10.1145/3306346.3322985
- [33] K. Vaidyanathan, M. Salvi, R. Toth, T. Foley, T. Akenine-Möller, J. Nilsson, J. Munkberg, J. Hasselgren, M. Sugihara, P. Clarberg, T. Janczak, and A. Lefohn. Coarse pixel shading. In *Proceedings of High Performance Graphics*, HPG '14, pp. 9–18. Eurographics Association, Goslar Germany, Germany, 2014.
- [34] D. R. Walton, R. K. D. Anjos, S. Friston, D. Swapp, K. Akşit, A. Steed, and T. Ritschel. Beyond blur: Real-time ventral metamers for foveated rendering. *ACM Trans. Graph.*, 40(4), July 2021. doi: 10.1145/3450626.3459943
- [35] M. Weier, T. Roth, A. Hinkenjann, and P. Slusallek. Foveated Depth-of-Field Filtering in Head-Mounted Displays. *ACM Trans. Appl. Percept.*, 15(4):26:1–26:14, Sept. 2018. doi: 10.1145/3238301
- [36] M. Weier, T. Roth, E. Kruijff, A. Hinkenjann, A. Pérard-Gayot, P. Slusallek, and Y. Li. Foveated real-time ray tracing for head-mounted displays. *Comput. Graph. Forum*, 35(7):289–298, Oct. 2016. doi: 10.1111/cgf.13026
- [37] M. Weier, M. Stengel, T. Roth, P. Didyk, E. Eisemann, M. Eisemann, S. Grogorick, A. Hinkenjann, E. Kruijff, M. Magnor, K. Myszkowski, and P. Slusallek. Perception-driven accelerated rendering. *Comput. Graph. Forum*, 36(2):611–643, May 2017. doi: 10.1111/cgf.13150