Journal on Interactive Systems, 2024, 15:1, doi: 10.5753/jis.2024.4352

Foveated Path Culling: A mixed path tracing and radiance field approach for optimizing rendering in XR Displays

Horácio Henriques [**Universidade Federal Fluminense** | ***horaciomacedo@id.uff.br***]
Alan de Oliveira [**Universidade Federal Fluminense** | ***alan.oliveira@ifro.edu.br***]
Eder Oliveira [**Universidade Federal Fluminense** | ***eder_oliveira@id.uff.br***]
Daniela Trevisan [**Universidade Federal Fluminense**| ***daniela@ic.uff.br***]
Esteban Clua [**Universidade Federal Fluminense** | ***esteban@ic.uff.br***]

Institute of Computing, Universidade Federal Fluminense, Av. Gal. Milton Tavares de Souza, s/n, São Domingos, Niterói, RJ, 24210-590, Brazil.

Received: 05 April 2024 • **Accepted:** 14 June 2024 • **Published:** 17 June 2024

Abstract: Real-time effects achieved by path tracing are essential for creating highly accurate illumination effects in interactive environments. However, due to its computational complexity, it is essential to explore optimization techniques like Foveated Rendering when considering Head Mounted Displays. In this paper we combine traditional Foveated Rendering approaches with recent advancements in the field of radiance fields, extending a previous work and including recent advancements based on Gaussian Splatting. The present paper proposes the usage of mixing real time path tracing at the fovea region of an HMD while replacing the images at the peripheral by pre-computed radiance fields, inferred by neural networks or rendered in real time due to Gaussian splats. We name our approach as Foveated Path Culling (FPC) due to the process of culling raycasts, diminishing the workload by replacing most of the screen raytracing tasks by a less costly approach. FPC allowed us for better frame rates when compared to purely path tracing while rendering scenes in real time, increasing the frame rate speedup proportionally to the display resolution. Our work contributes to the development of rendering techniques for XR experiences that demand low latency, high resolution and high visual quality through global illumination effects.

Keywords: Virtual Reality, Foveated Rendering, Radiance Fields, Visual Perception, 3D Gaussian Splatting

1 Introduction

New advancements in the field of computer graphics are enabling users with the power of real-time path tracing in order to achieve better visuals effects at stable frame rates in interactive environments due to hardware enhancements and novel AI based solutions that reduce latency while increasing image quality [Kilgariff *et al.*, 2018; Caulfield, 2022; Einhon and Mawdsley, 2023]. Meanwhile, new developments in the field of artificial intelligence enabled the spawn of a whole new field of neural rendering techniques: the Neural Radiance Fields (or NeRFs), in which multilayer *perceptrons* are fed with images and space coordinates in order to output new views of a scene [Mildenhall *et al.*, 2021]. The field of neural rendering for view synthesis is an exciting and important field of study that not only demands attention, but also inspires new and exciting works on visual perception and real-time rendering. Kerbl *et al.* [2023] is one of the first radiance field related solutions that achieves real-time display rates with high resolutions without trading off quality for speed nor recurring to raster render pipelines [Chen *et al.*, 2023].

At the same time, the popularization of extended reality (XR), specially Virtual Reality (VR), devices and applications [Alsop, 2024] demands better photorealism in graphics, which implies in including global illumination methods during render. However, this higher graphic fidelity comes with the downside of demanding more expensive hardware to ensure optimal user experience. This is due to the need for higher resolutions and lower latency than what is expected on standard displays to provide immersive experiences in XR [Swafford *et al.*, 2016; Albert *et al.*, 2017]. While not achieving high enough resolutions may cause user immersion to break, lower latencies may also cause harm to end users by the way of *cybersickness*[Porcino *et al.*, 2021]. This risk becomes even greater when we realize that VR is becoming more accessible to the public by means of education and work [Tanaka *et al.*, 2023; Souza *et al.*, 2023; Fernandes *et al.*, 2022]. Given this, it becomes imperative to search for strategies that optimize rendering [Swafford *et al.*, 2016; Weier *et al.*, 2016]. One of such strategies commonly used is the Foveated Rendering (FR), which relies on the fact that human sight is not equal throughout the entire eye, having lower acuity the further the captured image is from the center of the retina [Guenter *et al.*, 2012; Mohanto *et al.*, 2022].

In our previous work we proposed a novel foveated rendering approach that uses gaze-dependant properties of NeRF-like technologies in order to simulate path-traced effects in real time in the peripheral vision. We conducted tests with standby images in order to show seamless visual perceptions by combining both rendering techniques. In this extension we developed a Vulkan-based visualizer that outputs radiance fields and real-time ray traced scenes simultaneously. We also show and discuss performance improvements when using the hybrid rendering approach. We implemented 3DGS as the radiance field-based solution allowing better visual quality, better performance and faster training than the radiance field solution used in past works [Henriques *et al.*, 2023].

This is how we expand on the findings of the last work:

1. We give a new name for Foveated Rendering techniques that aim to replace path traced images for Radiance Fields due to their distance to the fovea: Foveated Path Culling;
2. We experimented our approach mixing the original 3D Gaussian Splatting implementation with a Vulkan path tracer, outputting graphics in real time by our technique;
3. We explored the performance gains and how they are affected by the foveal radius while using Gaussian Splatting at the periphery of the screen.

This work is organized as follows: the "Related Works" section presents in detail important subjects to the comprehension of this work; the "Foveated Path Culling" section details our architectural proposal and the proposed hybrid rendering approach. Following, the "Comparing Visual Quality" section we present a visual comparison of NeRF-based Foveated Path Culling against other foveated rendering techniques. At the "User Evaluation" section, we describe, analyze and discuss results of our user experiment; the "Performance Tests" section, on the other hand, has us discussing in more detail on how Foveated Path Culling with 3DGS enhanced rendering performance. Finally, the section "Future Works and Conclusion" presents our main conclusions and future works that may increase and outperform even better our actual findings.

2 Related Works

In this section we first briefly discuss about real time ray tracing elements, and what it implies in the field of VR. In the second subsection, we present different FR strategies. Lastly, we briefly discuss the field of radiance fields and how it can be integrated into FR implementations.

2.1 Path tracing

Path tracing is an approach to conventional ray tracing proposed by Kajiya [1986] in order to optimize the equation proposed in ray tracing's seminal work[Whitted, 1979]. The idea behind traditional ray tracing relies on launching a great number of rays in order to accurately simulate light transfer, specularity reflections and refraction. Path tracing, on the other hand, launches a single ray for each ray bounce on screen[Caulfield, 2022].

Combined with Monte Carlo methods, path tracing delivers the same properties given by traditional ray tracing in a much smaller time frame. Because path tracing works by tracing the rays back from the pixel to the light source in a limited number of bounces, it needs a certain number of samples per pixel in order to render an image while minimizing noise. Because of real-time constraints, using a high number of samples isn't always available, so there is a need for effective denoising solutions[Koskela *et al.*, 2019].

Nowadays, path tracing is how new GPUs with proper architectures are able to deliver real-time rendering with ray tracing[Kilgariff *et al.*, 2018]. However, that doesn't exclude rasterization from modern rendering pipelines; there are implementations of hybrid path tracing, which uses both path tracing and rasterization, in order to utilize strengths from both methods while delivering real-time rendered images[Barré-Brisebois *et al.*, 2019]. One example is the path traced effects implemented by the popular Unity Engine[Unity, 2024], which are implemented in the form of separate visual effects as volumes that alter the High Definition Rendering Pipeline[Unit, 2023].

On the matter of VR, the introduction of path tracing into the render pipeline represents a new challenge in order to maintain an appropriate performance under its own requirements. Ray tracing may also cause previous optimization efforts to become superfluous or insufficient to maintain adequate performance.

2.2 Foveated Rendering

Foveated Rendering encompasses various techniques aimed at exploiting the limitations of the human visual system to optimize computing resources by omitting rendering details that may not be perceived by the eye. This concept stems from the fact that visual acuity sharply declines as one moves away from the center of the retina, known as the fovea, due to an uneven distribution of photoreceptors. While the human visual system offers a wide monoscopic field of view of approximately 165 degrees per eye, the fovea covers only about 5 degrees around the optical axis. The area between the fovea and the periphery, known as the parafovea, can vary up to approximately 20 degrees [Guenter *et al.*, 2012; Kim, 2022; Mohanto *et al.*, 2022]. This area represents a small percentage of virtual reality headsets' screens, which tend to offer vertical FOVs that range from 80 up to 114 degrees [Infinite, 2024].

Although the concept of Foveated Rendering dates back to the nineties [Levoy and Whitaker, 1990], its relevance has surged with the recent proliferation of Virtual Reality (VR) and head-mounted displays (HMDs) [Mohanto *et al.*, 2022]. As VR gains broader appeal and users demand higher graphic fidelity, techniques that intelligently allocate computing power to conserve resources become increasingly vital. This is because lower fidelity graphics can compromise immersion, while higher latency may induce cybersickness [Albert *et al.*, 2017; Porcino *et al.*, 2021]. Moreover, the emergence of untethered HMDs reinforces the importance of optimizing energy consumption by devices, an area where Foveated Rendering may also provide assistance [Duinkharjav *et al.*, 2022].

Over the past decade, the field of Foveated Rendering has witnessed numerous approaches that handle peripheral imagery in diverse ways, incorporating gaze tracking or even eye tracking [Jabbireddy *et al.*, 2022; Mohanto *et al.*, 2022; Wang *et al.*, 2023]. Among the most popular techniques are adjustments to screen resolution at the periphery, gradually decreasing resolution further from the fovea. Additionally, other methods such as geometry simplification involve manipulating meshes to render geometry more efficiently at the periphery, while color simplification accounts for the reduced color sensitivity in peripheral vision [Mohanto *et al.*, 2022]. Depending on the render pipeline, path tracing techniques also introduce adaptations, such as reducing samples per pixel at the periphery of the screen [Koskela *et al.*, 2019].

One of the most notable techniques in the literature, to our knowledge, is Rectangular Mapping-based Foveated Rendering (RMFR), acclaimed for its superior visual quality indicators compared to more widely recognized methods, as stated in its paper. Despite its current implementation relying on raster-based pipelines, we regard it as the benchmark for visual quality when evaluating our approach, given its exceptional quality and accessibility on *GitHub* [Ye *et al.*, 2022; Ye, 2022].

2.3 Radiance Fields

Radiance Fields are a novel field in computer graphics, largely related to neural rendering due to its inception in the seminal work of Mildenhall *et al.* [2021] on NeRFs. They are described as functions that receive a 5D input (a 3D point in space and a 2D view direction vector) and output a color, typically denoted as an RGB value, along with a density parameter σ indicating the depth of the point relative to a specific viewpoint. The inclusion of density values aligns well with volume rendering techniques [Mildenhall *et al.*, 2021].

Even the seminal paper offers suggestions for optimizations aimed at accelerating training time and improving inference quality. These include leveraging higher dimensional positions to stabilize network learning and employing two networks simultaneously to better capture 3D scenes [Mildenhall *et al.*, 2021]. More recent works present within the field of Neural Radiance Fields propose many other network architectures [Barron *et al.*, 2022; Chen *et al.*, 2022; Lin *et al.*, 2024; Müller *et al.*, 2022; Müller *et al.*, 2021; Reiser *et al.*, 2021; Schwarz *et al.*, 2022; Sun *et al.*, 2022] and other rendering pipelines besides volume rendering [Chen *et al.*, 2023; Li *et al.*, 2023; Liu *et al.*, 2020; Schwarz *et al.*, 2022].

Since the inception of NeRFs, some works have tried to get rid of the neural networks in order to gain performance at converging results and rendering. Notable examples include Plenoxels [Fridovich-Keil *et al.*, 2022] and, more recently, 3D Gaussian Splatting [Kerbl *et al.*, 2023] (3DGS). Plenoxels replace multilayer perceptrons with spherical harmonics, while 3DGS involves optimizing three-dimensional Gaussian distributions. Particularly, 3DGS has garnered attention for its interactive frame rates and high-resolution outcomes, serving as inspiration for numerous subsequent works [Tang *et al.*, 2023; Wu *et al.*, 2023].

In our scenario, 3DGS holds particular significance as it enables rapid output of peripheral vision, thereby minimizing latency while rendering high-resolution views. While our prior work delved into NeRF-based outcomes, the extension of the work implements this new approach, capitalizing on its noted advantages.

3 Foveated Path Culling

The main contribution of our previous work primarily revolves around introducing a novel foveated rendering paradigm [Henriques *et al.*, 2023]. This paradigm employs a hybrid rendering approach by incorporating radiance fields to simulate peripheral vision in real-time rendered scenes, thereby enhancing the path tracing effects. The main motivation is that the gaze-dependant properties of radiance fields would provide a sufficiently adequate surrogate to gaze-dependant effects that path tracing commonly does. We justify the choice of NeRFs for the peripheral image due to their superior visual consistency compared to modifying the existing image in the standard rendering pipeline, despite a potential slight reduction in image quality. We also claim that NeRF is more convenient than raster-based approaches combined with path tracing, due to the fact that raster is unable to accurately simulate lighting effects, such as reflections, caustics, refraction or transparency, naturally achieved through path tracers[Whitted, 1980].

The hybrid system is based in a path tracer and a Gaussian Splatting rendering solution. The path tracer is developed in C++ and the Vulkan API, incorporating a multiple importance sampling (MIS) strategy. The path tracer follows the standard path tracing algorithm, which simulates the behavior of light in a scene by tracing rays from the camera into the scene and simulating their interactions with surfaces. The algorithm iteratively traces paths of light rays through the scene, accumulating radiance along each path until a termination condition is met.

To improve the efficiency and accuracy of the path tracer, we employ the MIS techniques, sampling from multiple probability distributions and combining the results to estimate the integral of interest. In our implementation, we apply MIS for both sampling the light source and sampling the Bidirectional Reflectance Distribution Function (BRDF).

The scene represented by the foveal region, a fully path traced scene rendered in real time, would also provide the training data over the chosen radiance field model. Ideally, this approach can be used regardless of the technique, with changes unique to each reconstruction technique used in the periphery. On our experiments, detailed in Sections 4 and 5, we used the scene in order to train a NeRF with multiresolution hash encoding. On the testing setup used to conduct the tests featured at Section 6, however, the path traced scene was used as data to train 3D gaussians.

Rendering the scene using Foveated Path Culling (FPC) consists of rendering both reconstructions on screen at the same time. That means having access to both representations of the same scene on memory: one mesh-based reconstruction for real-time path tracing, and access to the radiance field for whatever rendering technique is employed. In this work, this region uses raster strategies, since we are using 3DGS for peripheral vision Kerbl *et al.* [2023].

The rasterization step in 3DGS involves converting three-dimensional Gaussians into a two-dimensional image. This process entails projecting each splat onto the image plane based on its position and orientation relative to the camera. Nvidia CUDA provides a highly parallelizable and programmable framework that are able to create this particular gaussian rasterization tasks efficiently on GPU hardware[Laine and Karras, 2011]. Utilizing CUDA, the camera and projection matrices are applied to the splats' positions to transform them into screen-space coordinates. Following that, the contribution of each splat to the final image is computed by sampling its Gaussian distribution across neighboring pixels. This rasterization step ensures accurate projection

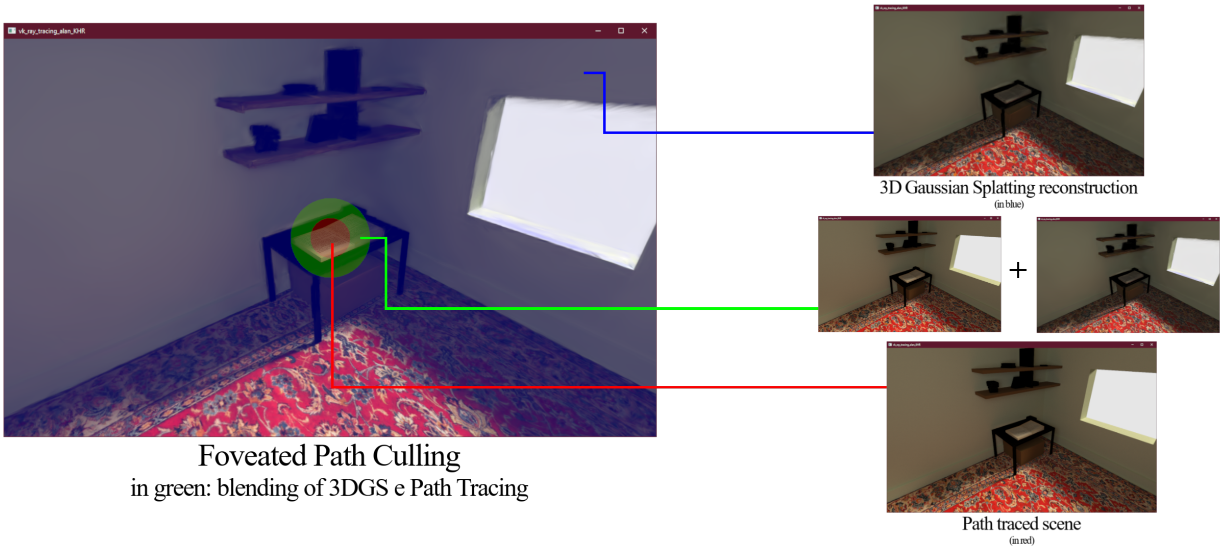

Foveated Path Culling
in green: blending of 3DGS e Path Tracing
3D Gaussian Splatting reconstruction
(in blue)
+
Path traced scene
(in red)

Figure 1. Demonstration on how does FPC composes the final image using 3DGS in the peripheral region. **In red:** the path-traced region, meant to be focused by the user fovea in a VR display; **in blue:** the 3DGS region, meant to cover the majority of the display and to be captured by the periphery of the eye; **in green:** the overlapping region between 3DGS and path tracing.

of the splats onto the image plane and appropriate blending of their Gaussian profiles, resulting in a visually coherent rendering of the volumetric data.

The final image is a composite of two rendering processes, controlled by two variables: an external region denoted as *fov-a* and an internal region denoted as *fov-b*, both measured in percentage of the screen. The peripheral image is rendered by the 3DGS rasterization process and covers the entire screen except for the central circular region, defined by *fov-b*, while the foveal image rendered by path tracer process covers a circular region defined by *fov-a*. As *fov-a* is always greater than *fov-b*, there is a portion of the screen where both renderings overlap. In this overlapping region we apply a linear interpolation, where the foveal image becomes less apparent the further we go from the foveal region, while the peripheral image becomes more visible. An illustration showing this process is presented in **Figure 1**.

4 Comparing Visual Quality

To quantify the differences between the original images produced by ray tracing, those created by another Foveated Rendering implementation, and our approach, we utilized the Structural Similarity Index Measure (SSIM)[Wang *et al.*, 2004]. We made this choice considering the alignment of SSIM with human visual perception, as SSIM effectively models the structural information, luminance, and contrast changes in the image that the human visual system perceives.

Our initial hypothesis is that our approach, while potentially producing slightly lower image quality, would demonstrate superior or comparable structural similarity values at the periphery of the images due to their consistent peripheral rendering. We validated this hypothesis by comparing the SSIM of our test cases with renders from the same scene while making use of Rectangular Mapped-based Foveated Rendering RMFR. We chose this FR method due to its greater SSIM values when comparing with other well-established FR implementations.

To generate the RMFR images, we set the compression parameter (σ) to 2.6 and both sampling distributions f_x and f_y to 0.1. We used these parameters since the compression parameter fit inside the threshold established by the original authors [Ye *et al.*, 2022] while the sampling distributions must be determined on a scene basis. This means that the SSIM values of RMFR presented in this comparison can be optimized.

The RMFR-generated images were compared to images created by our proposal. We composed 5 (five) frames of pre-rendered images of the same scene, dubbed “bedroom”, using our strategy. In our foveated image, the center of the image is generated at Unity Engine using the High Definition Render Pipeline and the ray-based implementations of effects such as Shadows, Screen Space Reflections, Ambient Occlusion, Contact Shadows and Global Illumination.

At the periphery of our foveated image, on the other hand, the frames were generated using Nvidia’s Instant Neural Graphics Primitives (Instant-NGP), trained with 148 images of the bedroom while using Logistic RGB activation, Exponential density activation, 0.95 density grid decay value, 0.0001 extrinsic and intrinsic L2 registrator and using the bedroom scene as training data during 30 (thirty) minutes. The loss function, calculated via Huber function, was at -7.289 at the end of training. The model was rendered to .png files at 403x422 for reasons we will disclose at section 5.2.

The results of our comparisons can be seen at Figure 2 and on Table 1. In the best cases, our approach achieved better SSIM scores, indicating a closer approximation to the original ray-traced images than the RMFR render. However, artifacts caused by NeRFs due to noisy textures on the training data or less than optimal feature matching can lead to inferior but comparable SSIM values.

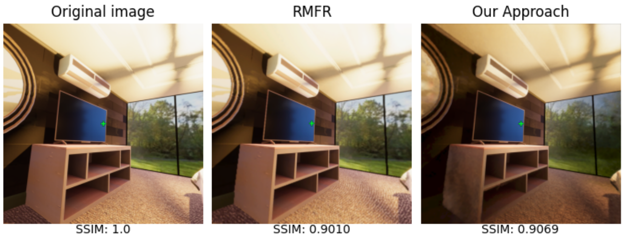


Figure 2. SSIM comparison of Image 04 using RMFR and our approach against a mixed rendered pipeline. Our approach achieved comparable SSIM scores, indicating, at times, a closer approximation to the original ray-traced images than the best Foveated Rendering solution we could find.

Table 1. Comparsion of SSIM values of test images

Image	RMFR SSIM	Our SSIM
Image 01	**0.8692**	0.8278
Image 02	0.8636	**0.8993**
Image 03	**0.8817**	0.8613
Image 04	0.9010	**0.9069**
Image 05	**0.9025**	0.8966

5 User Evaluation

Our main hypothesis in the present work claims that three-dimensional scenes inferred by NeRFs can be used to replace traditional rendering methods at the periphery of the eye without compromising the user's attention. The effect of having the central region of the screen generated by a mixed render pipeline and the peripheral region generated by a NeRF inference will be called "foveation" for simplicity since its intended effect is the same as any foveated scene by other techniques.

5.1 Apparatus

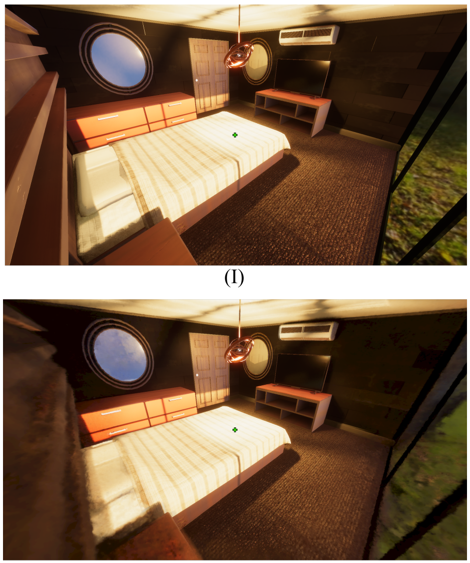
The images used on the comparison on the last section were used in the user experiment as well. Each image created with our approach became part of a pair of images, each pair representing one angle of the bedroom scene. The second image of each pair served as the reference, rendered using standard path-traced effects, similar to the fovea region of the foveated images.

We included a little cross at the middle in order to serve as a fixation point for every user in the middle of each one of the images used at the comparison. Figure 3 provides an example of an image pair with the cross included.

The reason for including the crosses is due to a limitation of the HMD used in the user tests, a Meta Quest 2 model, which does not have eye tracking capabilities. For devices with eye tracking, this would not be necessary, as the ray-traced region would be dynamically adjusted to meet the user gaze. However, our experiment demands a fixation point so the user would be less likely to divert their attention from the fovea region.

5.2 Methodology

The study implemented the Double-Stimulus Impairment Scale (DSIS) variant I of the Recommendation ITU-R BT.500-13 [Series, 2012], a method commonly used for subjective quality assessment of images. Participants were presented with a sequence of two stimuli: the reference and the degraded image. Participants were asked to evaluate the difference in perceptible quality between the two stimuli.



(I)

(II)

Figure 3. Pair of images used during our experiment of a single angle of the bedroom scene. The top image (I) was rendered in Unity Engine, using the aforementioned ray-based effects. The bottom image (II) was a composition of the Instant-NGP in peripheral area and path tracing in the center. In the middle of both images, we put an artificial fixation point for user experimentation.

5.3 Participants

The pool consisted of 11 participants, including 7 males and 4 females, with ages ranging from 21 to 51. Out of the 11 participants, 6 had prior experience with VR technologies. Additionally, 8 out of 11 participants reported some form of vision problem, all of whom had corrected vision. The visual acuity ranged from myopia to a combination of myopia, astigmatism, and hypermetropia. The participants' ages had an average of 30.9 years with a standard deviation of 11.2 years. The participants are detailed in Table 2.

Table 2. Participants' Information

Participant	Age	Gender	Experience	Vision Problem
P01	23	Male	Yes	No
P02	32	Male	Yes	Yes
P03	25	Male	No	Yes
P04	24	Male	Yes	Yes
P05	49	Female	Yes	Yes
P06	29	Male	Yes	Yes
P07	51	Female	Yes	Yes
P08	28	Male	No	Yes
P09	26	Male	No	No
P10	42	Female	No	Yes
P11	21	Female	No	Yes

5.4 Testing Procedure

The participants were instructed to remain stationary, directing their gaze towards the center of the screen. When required, they provided their opinion on the difference between two images. They were informed about the aim of the experiment and the possibility of experiencing symptoms of cybersickness, such as disorientation, headaches, nausea, and dizziness, although the chances were low due to the nature of the experiment.

When testing, the user is first put through a brief tutorial on how to behave during the experiment. Once we made sure their HMD was properly positioned, we presented a test image featuring the cross icon used as a fixation point, and provided instructions for the user to focus their gaze at the fixation pointed and rest its head, minimizing any unnecessary movement.

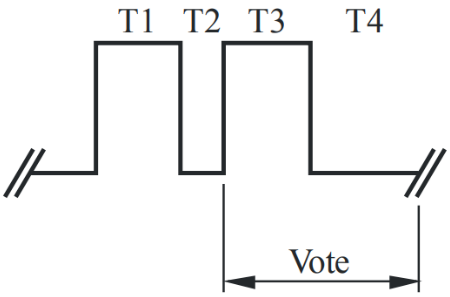


Figure 4. Phases of presentation [Series, 2012]: T1 = 2s - reference image, T2 = 1s - black screen, T3 = 2s - foveated image, T4 = 5s-11s - black screen.

Each user underwent a Degradation Category Rating (DCR) assessment, where they compared two images of each pair: the foveated image and the image of the same angle of the bedroom, generated in Unity using the aforementioned mixed render pipeline[Chellappa and Theodoridis, 2017]. To avoid potential bias, these images were randomly shown in pairs, as shown in Figure 4. Each image of the pair was displayed for two seconds, with a one-second break between them. First, the reference image was shown, followed by the foveated image. After each presentation, the user provided a subjective assessment on a scale from 1 to 5. A rating of 1 indicated that the difference between the images in a pair was most apparent and immersion-breaking, while a rating of 5 meant that the difference was imperceptible. The process was then repeated with another pair of images until every pair was tested.

In this study, we adapted the grading scale proposed by the ITU-R BT.500-13 recommendation to suit the nature of the images presented and the specific objectives of our research. Instead of using a five-point quality scale that focuses on image degradation, we chose a scale that assesses how perceptible differences between images in a pair are and how much they interfere with immersion. This approach allowed participants to focus more on the immersive experience provided by the images, which is crucial to the goals of this research.

For a single user, each pair was shown at least thrice, and we had five pairs of images. In total, each user represented a fifteen-round experiment. The flow of the experiment is presented at Figure 4.

5.5 Results Assessment

To ensure the most accurate evaluation possible, we employed the Double-Stimulus Impairment Scale (DSIS) method, also known as the EBU method[Series, 2012], widely used in quality perception studies. This method allowed for a systematic and comparable evaluation among the participants.

The study collected a total of 165 evaluations from the 11 participants. Each participant assessed fifteen pairs of images by evaluating each one of the five different pairs three times throughout the experiment.

5.5.1 Observer Screening

After data collection, we conducted an Observer Screening to ensure the reliability of our data. This screening process incorporates a detailed examination of participants' scores to identify potential outliers or significant deviations from the mean score.

The process involves the calculation of the ratio between two counters $(P_i$ and $Q_i)$ associated with each participant relative to the total number of scores from that participant for the entire session. Next, we calculate the absolute value of the difference between P_i and Q_i counters divided by the sum of P_i and Q_i. If the first ratio exceeds 5% and the second ratio is less than 30%, we must remove the observer associated with these counters from the analysis.

After this rigorous screening process, we determined that all participants in this study met the criteria, and we did not exclude any from our analysis.

5.5.2 Result Analysis

We calculated the average image quality scores to be 4.37 (to two decimal places of precision), which indicates a high overall perceived quality of the evaluated images. We also calculated the 95% confidence interval for the mean of scores, which ranges between 4.23 and 4.50. Figure 5 illustrates the average image quality scores by participants and the general public.

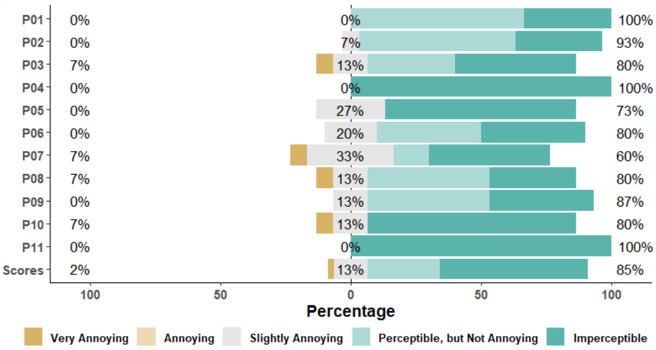


Figure 5. Average image quality scores by participants and the general public.

The results of the analysis are presented in Table 3. This table displays the mean scores, standard deviations, and 95% confidence intervals for each participant. Participants are identified by the codes P01 to P11. The table provides an overview of the responses on a five-grade impairment scale, where 5 indicates imperceptible and 1 indicates very annoying.

These data suggest that the perceived quality of the evaluated images was consistently high, with slight variation among participants and between different image sequences.

Table 3. Results by Participants

Participant	Mean Score	SD	Confidence Interval
P01	4.33	0.49	(4.23, 4.43)
P02	4.27	0.59	(4.17, 4.37)
P03	4.13	1.13	(4.03, 4.23)
P04	5.00	0.00	(4.90, 5.10)
P05	4.47	0.92	(4.37, 4.57)
P06	4.20	0.77	(4.10, 4.30)
P07	3.93	1.22	(3.83, 4.03)
P08	4.00	1.07	(3.90, 4.10)
P09	4.27	0.70	(4.17, 4.37)
P10	4.47	1.19	(4.37, 4.57)
P11	5.00	0.00	(4.90, 5.10)

5.5.3 Discussion

Our statistical analyses yield a 95% confidence interval for the mean score, ranging between 4.23 and 4.50. This boundary suggests that repeated studies would consistently achieve a mean score exceeding 4.23. This score positions the evaluated images within the “perceptible but not annoying” (4) and “imperceptible” (5) brackets of the five-grade degradation scale.

6 Performance Tests

In this section, we will delve deeper into how Foveated Path Culling enables us to achieve higher frame rates than solely path tracing a 3D scene. In the original paper, we conducted tests in a basic testing scenario with two separate applications rendering peripheral and foveal images, and utilized profilers to measure performance improvements. Our aim is to validate and build upon the results outlined by Henriques *et al.* [2023], gaining a more comprehensive understanding of the speedup provided by FPC.

The primary motivation behind changing from a neural radiance field method to 3DGS for capturing peripheral vision stems from the potential for improved latency with our technique. By avoiding the use of neural networks, we can achieve faster rendering times while maintaining state-of-the-art visual fidelity in the reconstruction [Kerbl *et al.*, 2023]. Whereas the original approach involved rendering the peripheral image at a lower resolution to maintain a stable frame rate higher than fully path tracing a scene, 3DGS enables us to achieve true real-time frame rates at higher resolutions.

Theoretically, measuring any performance gains from any Foveated Path Culling implementation will revolve mostly around understanding the computational cost for rendering the peripheral vision and comparing it against the cost with path tracing the same region. In the case presented on this work, we are comparing the capacity of rendering 3DGS at high resolutions against rendering path traced scenes. Any FPC rendered scene will then be as costly as rendering the 3DGS reconstruction of said scene in a given resolution plus the cost of rendering the path traced foveal radius on the screen. The cost for rendering such scene is somewhere within the cost of fully path tracing a scene and fully reconstructing a scene through a radiance field, and we will be successful in our hypothesis if the cost for using FPC is lower than entirely path tracing said scene.

This section is organized as follows: a subsection describing the testing environment under which the results in this paper were measured; a subsection exploring the performance gain within different resolutions, further establishing this approach’s applicability to high-resolution cases; a subsection describing the role played by the foveal radius in the speedup given by the approach, with a subsubsection exploring how Foveated Path Culling holds up rendering various views of the same scene, with different speedups according to the region displayed on screen, and; a section exploring how the sample-per-pixel number on the foveal size impacts the frame rate.

6.1 Testing scenario

The performance tests were carried out in a custom-built 3D scene specifically designed by the researchers for conducting 3D Gaussian Splatting (3DGS) experiments. The visualizer’s frame rate was measured during the rendering of these scenes. Our objective was to gain a deeper understanding of the conditions under which we achieve speedup using our approach and to ascertain if the previous findings could be replicated. This exploration led to valuable insights into the nature of our approach that were not evident in the previous experiment.

The following results were captured on a computer with the following specifications: AMD Ryzen 7 5700X CPU; NVIDIA GeForce RTX 3060 12GB GPU and 16GB DDR4 RAM. All tests in this section were conducted using another 3D scene, constructed by the authors using 3D assets found on the Internet, which we named Room. The materials used in said Room are lambertian diffuse models, mirrors and glossy surfaces. The entire Room scene is composed by 21K triangles and 2 light sources: a sun light, and an interior point light.

Due to technical limitations, we were restricted to rendering frames with a maximum height of 1061 pixels, with no observed limitations on horizontal resolution during our tests. While this prevented us from matching the exact resolution of the Meta Quest 2’s display screen, our visualizer enabled us to render the scene with an approximate pixels count equivalent to the Meta Quest 2’s theoretical pixel count per eye.

6.2 Speedup with Multiple Resolutions

In this test, we looked at how Foveated Path Culling using 3DGS for the peripheral image behaved at different resolutions as we measured how many frames per second (FPS) we were able to render. As we aimed to look for different resolutions, this test kept the foveal region as a circle centered on the screen, with its radius fixed at 20% of the screen’s height. This approach is appropriate since that the Meta Quest 2’s eye has a vertical FOV of approximately 100 degrees, which renders 20 degrees of the screen around the center (i.e. 20% of the screen’s height) as the foveal-plus-parafoveal regions. As we experiment with resolutions that matches the Meta Quest 2’s eye aspect ratio, we can better approximate how such foveal region can have an impact over the render process.

The FPS was captured throughout ten seconds after the startup of the application. Between each capture, the program was recompiled in order to prevent any bias from previous tests to alter the performance. Every single capture had the visualizer's camera fixed at the same spot in the scene with the same intrinsic parameters, since as future sections will show, different regions of the same scene may be rendered at a different rate.

The resolutions featured at **Table 4** were handpicked due to some reasons: 800x600 is a small enough resolution that does not warrant FPC usage and does not benefit from the technique. 1012x1061 is a resolution with the same proportions as the single Meta Quest 2 eye screen, with positive results. 1954x1061 has a pixel count close to the popular 1920x1080 resolution. Last but not least, 3316x1061 has a pixel count close to the single Meta Quest 2 eye screen, even if it does not match the aspect ratio. The path tracing region was rendered at 1 SPP in this test.

All of these results were captured at the origin point of the scene, featured on **Figure 6**.

Table 4. Comparison of Foveated Path Culling's FPS when compared to path tracing the same scene view at multiple resolutions. The second line, with a 1012x1061 resolution, shares the Meta Quests 2's eye aspect ratio, despite its smaller pixel count.

Width	Height	Avg. FPS	Avg. RT FPS	Speedup
800	600	265.4	330.2	-19.62%
1012	1061	195.2	152.0	28.28%
1954	1061	107.0	77.7	37.71%
3316	1061	66.5	46.4	43.31%

As we analyze the numbers on **Table 4**, it is worth emphasizing that, while FPC excels in rendering high-resolution scenes, its performance dips as the resolution decreases. However, this drawback is not a worry for us, as our application is meant to enhance rendering in high-resolution environments. As we will elaborate in the following section, FPC's performance improves proportionally with the resolution.

In our optimal test scenario, we achieved a significant 43.31% speedup, elevating performance from a median value of 46 FPS to 66 FPS. This outcome validates the original work's assumption that a more sophisticated testing setup would yield greater performance. FPC represents a much more relevant gain with bigger resolutions, due to the fact that the expense of path tracing a region escalates more rapidly compared to the cost of rendering a 3DGS reconstruction.

On other note, using different aspect ratios may be convenient, as a vertical FOV that covers 20% of the display encompasses a smaller region of the screen on a wider display compared to a taller display. This results in a smaller number of path traced frames being replaced by 3DGS. Recognizing this, we ran another round of the same test, fixing the aspect ratio and varying the resolution. The results, shown at **Table 5**, confirms our first result that FPC does allow for greater frame rates at greater resolutions, as speedup increases with the resolution used.

Table 5. Comparison of Foveated Path Culling's FPS when compared to path tracing the same scene view at multiple resolutions, following the Meta Quest 2's eye aspect ratio.

Width	Height	Avg. FPS	Avg. RT FPS	Speedup
782	820	248.4	253.1	-1.85%
859	900	224.0	210.0	6.67%
930	975	211.2	180.5	17.00%
1012	1061	195.2	152.0	28.42%

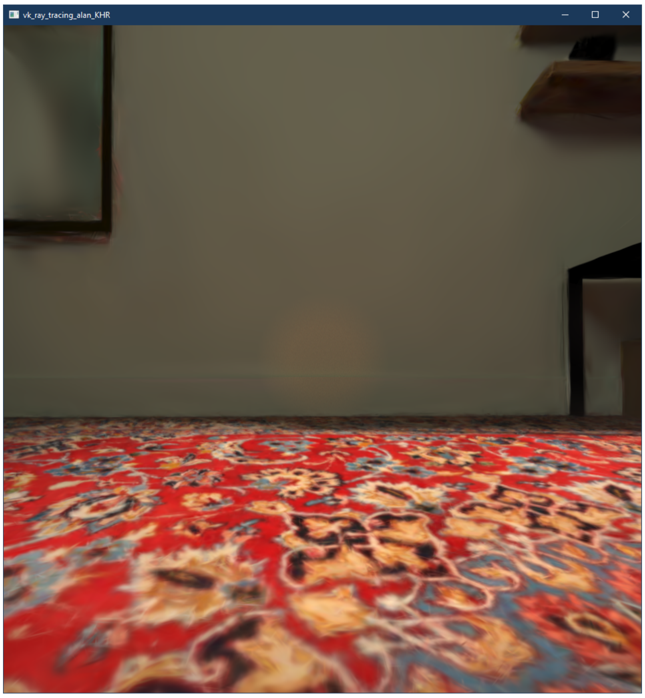


Figure 6. View with FPC used at the experiment described at Section 6.2.

6.3 Foveal Radius' Impact on Speedup

In this test, we conducted a ten-second measurement of the visualizer's frame rate while progressively increasing the foveal radius. We then compared this performance to fully path tracing the same scene view for an additional ten seconds. The foveal radius was varied within a range of 10% of the screen's height up to 90% over the ten-second duration. The human fovea typically spans 5 degrees surrounding the center of the retina, while the parafovea region may extend up to 20 degrees [Mohanto *et al.*, 2022]. As previously stated, in a standard Meta Quest 2, 20 degrees of FOV is roughly equivalent of 20% of the screen's height. The path tracing region was rendered at 1 sample per pixel (SPP) in this test.

Foveated Path Culling consistently demonstrated better frame rates at higher resolutions. Our experiment tested four resolutions: 3316x1061, 1954x1061, 1012x1061 and 1280x780. The 3316x1061 resolution has almost the same pixel count of the single Meta Quest 2's eye, the 1954x1061 has almost the same number of pixels than the commonly used 1920x1080 resolution, 1012x1061 is the largest resolution we could attain within our visualizer, and 1280x780 is another popular resolution that would allow us to observe how foveal radius impacts performance at lower resolutions. Each resolution underwent three test rounds, from which the mean frame rate was calculated for each radius. The principal analysis focused on the highest resolution, presented

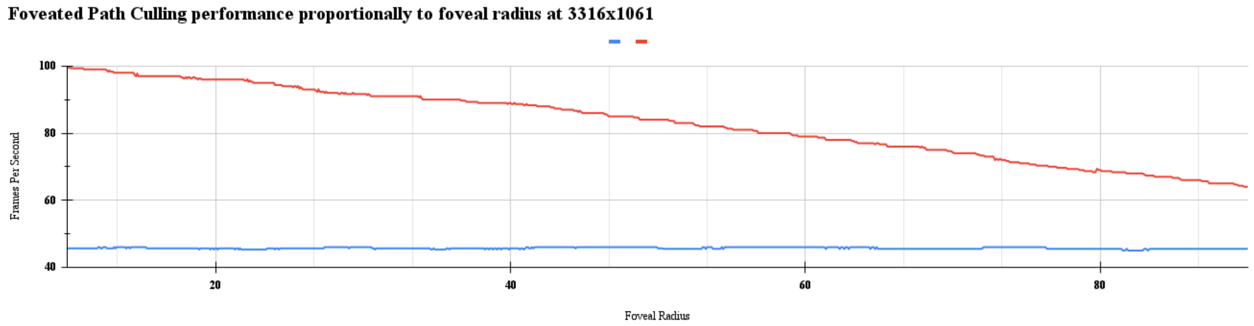


Figure 7. Exploration on how the radius of the foveal region impacts the system's performance, correlating the radius of the fovea with the frame rate of the visualizer at any given moment at the same time we compare them to the visualizer's performance when path tracing the exact same scene at the same resolution of 3316x1061. The Foveal Radius axis is the percentage of the vertical resolution being covered by the foveal region. **In red**: the performance of Foveated Path Culling while varying the radius. **In blue**: the performance of fully path tracing the scene for ten seconds after compiling the application.

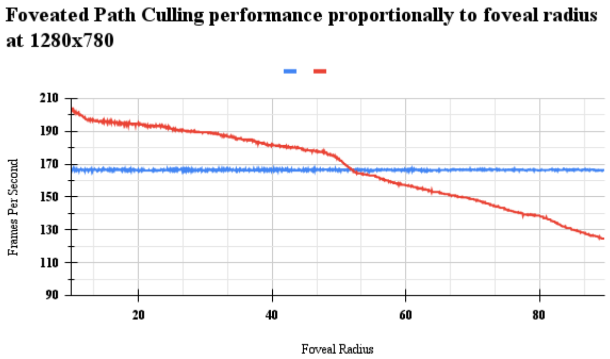


Figure 8. Performance decrease of Foveated Path Culling with the widening of the foveal radius at 1280x780 resolution. FPC becomes as costly as path tracing when the foveal radius achieves 51% of the screen's height, and becomes more costly at 52%. The Foveal Radius axis is the percentage of the vertical resolution being covered by the foveal region. **In red**: the performance of Foveated Path Culling while varying the radius. **In blue**: the performance of fully path tracing the scene for ten seconds after compiling the application.

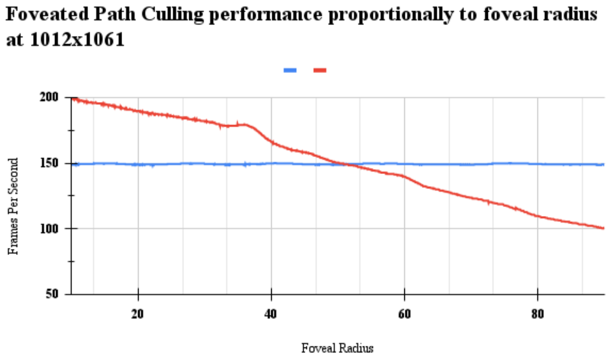


Figure 9. Performance decrease of Foveated Path Culling with the widening of the foveal radius at 1012x1061 resolution. FPC becomes as costly as path tracing when the foveal radius achieves 50% of the screen's height, and becomes more costly at 52%. The Foveal Radius axis is the percentage of the vertical resolution being covered by the foveal region. **In red**: the performance of Foveated Path Culling while varying the radius. **In blue**: the performance of fully path tracing the scene for ten seconds after compiling the application.

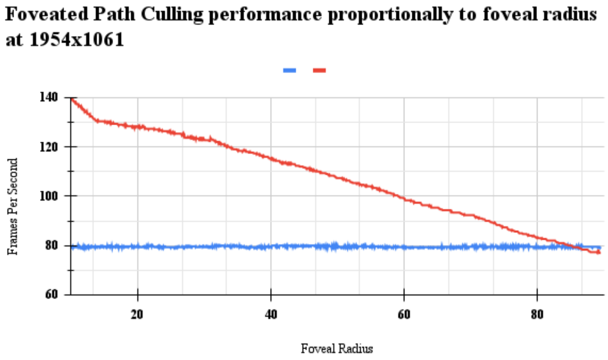


Figure 10. Performance decrease of Foveated Path Culling with the widening of the foveal radius at 1954x1061 resolution. FPC becomes as costly as path tracing when the foveal radius achieves 85% of the screen's height, and becomes more costly at 86%. This result points to a tendency of FPC's performance enhancement to become more relevant usage to increase the more we increase resolutions on screen. The Foveal Radius axis is the percentage of the vertical resolution being covered by the foveal region. **In red**: the performance of Foveated Path Culling while varying the radius. **In blue**: the performance of fully path tracing the scene for ten seconds after compiling the application.

in **Figure 7**, while corresponding figures for smaller resolutions are detailed in **Figures 9**, **8**, and **10**. We conducted each round rendering the same spot of the aforementioned bedroom, shown at **Figure 11**.

One general tendency revealed by this experiment is that the resolution increases proportionally with the threshold in which the foveal radius becomes as efficient as standard Path Tracing. The sole scenario wherein the frame rate in FPC dipped below the 72 FPS was in **Figure 7** at the point that the foveal radius was beyond what is expected for any real foveated rendering application, while still maintaining an above 60 FPS performance. It is noteworthy that even in the extreme scenario of covering 90% of the height of the screen, we still observed superior performance compared to fully path tracing the scene. On other testing scenarios, however, both path tracing and FPC consistently maintained frame rates exceeding 72 FPS, which aligns with the contemporary industry standard for VR experiences [Warburton *et al.*, 2023].

It is important to highlight the differences between the performance metrics featured on **Table 4** and **Figure 9**. In

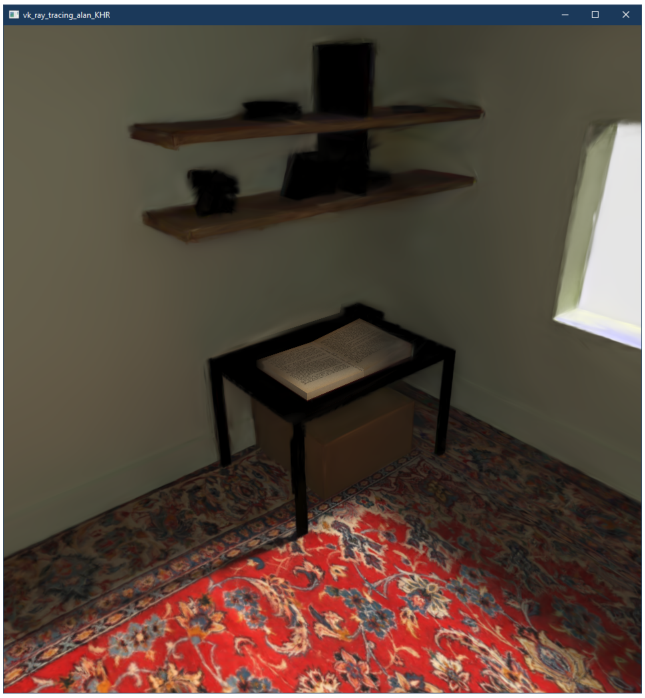


Figure 11. View with FPC used at the experiments described at Sections 6.3 and 6.4.

the **Figure 9**, when the foveal radius is around 20% of the screen's height, FPC achieves a 14% speedup, while at the table FPC enables a 28% speedup. This discrepancy, albeit manifesting predominantly in smaller regions, points to an inconsistency on how does FPC may improve frame rates during rendering.

6.3.1 Performance Exploration of Different Regions

Due to the nature of real-time rendering, not all regions of a scene are rendered equally. Factors such as multiple light sources, multiple different materials, polygonal density on screen, optimizations techniques such as occlusion culling and foveated rendering all play a role on how fast a path tracer can display new frames on screen. At the same time, rendering 3DGS depends on the number of different gaussians on screen at each given time, and the amount of space they take on the display. With this knowledge, it is important to determine if Foveated Path Culling can still consistently improve frame rates whenever rendering scenes in real time.

Because of that, we tested Foveated Path Culling's performance on various spots of the same scene, and compared against the frame rate of fully path tracing the same region. Every single image present in the **Figure 12** and its respective frame rates were rendered in 1280x780, since it showed the least amount of speedup while exploring the rendering of FPC. Every single image present in the **Figure 13** and its respective frame rates were rendered in 3316x1061. It is reasonable to assume that if we observed a speedup in our worst-case scenario, as evidenced by the results from the last two sessions, we would naturally anticipate an improvement in performance in our best-case scenario. However, it was imperative for us to confirm this hypothesis.

Analyzing the results from the examples rendered in 1280x780, it is clear that Foveated Path Culling's speedup is really variable according to the region. On one handpicked example, FPC is outperformed by path tracing, but even at its worst it never performs worse than the VR industry standard of 72 FPS. Interestingly, we observe that the frame rate varies greatly within FPC while keeping mostly stable on path tracing at 1 SPP, showing that most of its frame rate variation comes from rastering 3DGS. However, even this suboptimal example follows the trend of improving performance while diminishing the foveal radius.

On the other hand, we had to check out for the 3316x1061 example, since they render the same number of pixels than a single Meta Quest 2 eye. Even with a fairly stable frame rate when path tracing the scene, Foveated Path Culling had frame rates ranging from 60 FPS up to 115 FPS when rendering with foveal radius of 50% of the vertical screen, or from 66 FPS to 141 FPS when rendering with foveal radius of 20% of the vertical screen. Even at the worst example, looking through the angle at the same position that caused FPC to be outperformed on **Figure 12**, we still achieved a noteworthy speedup of 20% on a wider foveal radius.

The variation of performance in the same scene using Foveated Path Culling comes from rastering different regions of the same scene. Rendering the lightened region of the floor on the scene causes 3DGS to lose performance due to its number of points, which increases the rendering time for each frame. This is bound to happen to any 3DGS reconstruction with many varying colors. 3DGS does not, however, lose performance whenever rendering scene objects from closer, differently from path tracing.

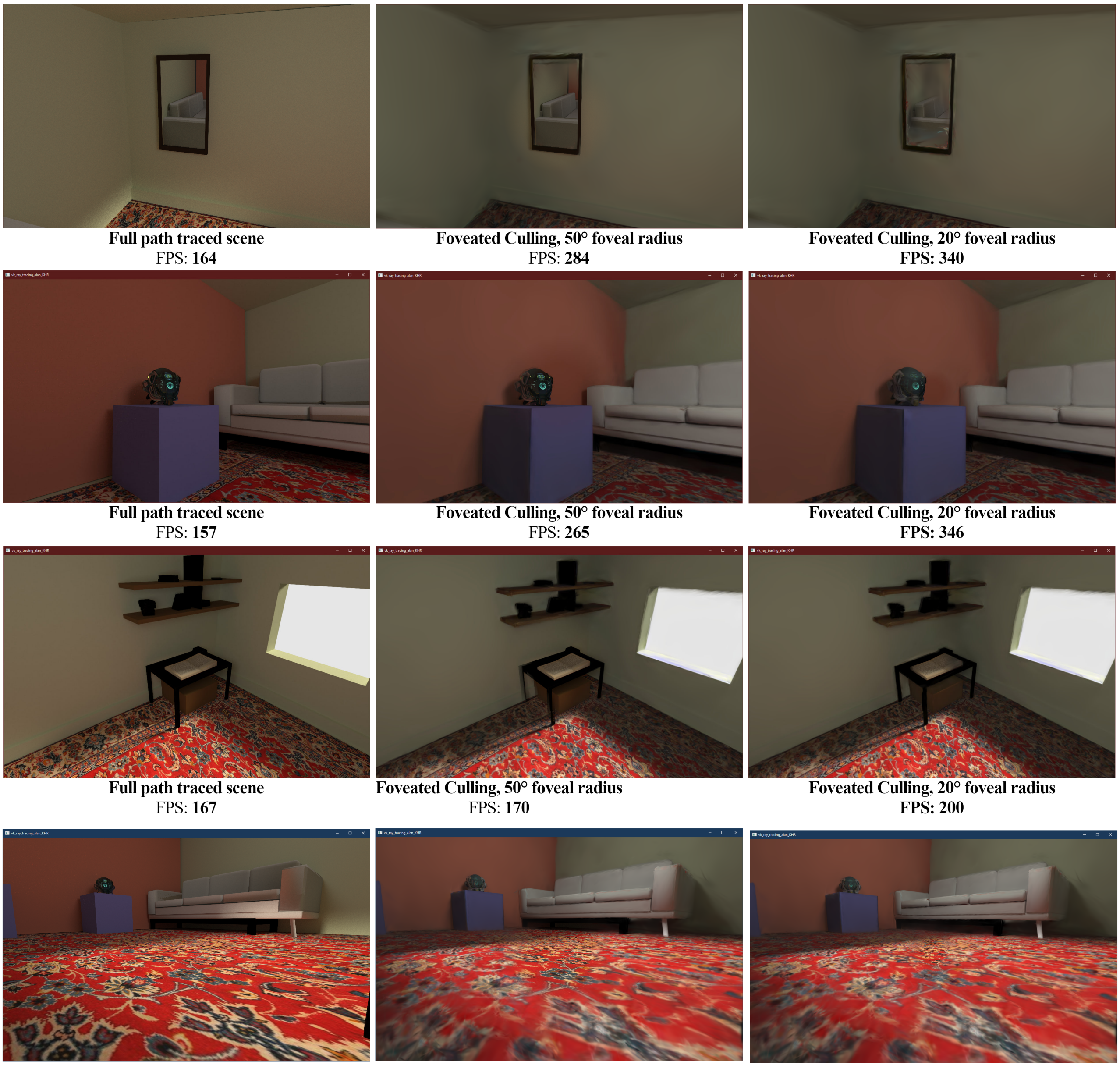
6.4 Samples-Per-Pixels' Impact on Speedup

The number of samples per pixel used in the foveal region significantly influences the frame rate of the screen, considering the real-time path tracing aspect of the foveal region. It's intuitive to expect that increasing the samples per pixel in the foveal region would require more computational effort to generate new frames, similar to the impact seen when expanding the foveal radius. However, the extent to which performance deteriorates with higher SPP values is uncertain to us at the moment.

To gain a deeper insight into this impact, we conducted performance tests akin to those detailed in subsection 6.3. We maintained a static camera position focused on the location depicted in **Figure 11** while collecting frame rate data for each frame rendered. Across three iterations, we then varied the foveal radius from 10% of the screen's height up to 90% over a 10-second interval. We performed this procedure for varying SPP measurements: initially at 1SPP, mirroring the conditions outlined in subsection 6.3, followed by repetitions at 2SPP, 4SPP, and 6SPP. Lastly, we conducted the testing regimen with full path tracing of the image, forgoing the use of FPC. All tests were conducted rendering the images at 1012x1061 due to the aspect ratio and technical limitations previously stated.

The collected results can be seen at **Figure 14**.

As we can see, increasing the number of SPP does decrease the performance of the scene. When measuring FPS at the 10% radius, as we got 200 FPS with only 1 SPP, which becomes 192 FPS with 2 SPP (a 4% performance decrease),



Full path traced scene
FPS: **164**

Foveated Culling, 50° foveal radius
FPS: **284**

Foveated Culling, 20° foveal radius
FPS: 340

Full path traced scene
FPS: **157**

Foveated Culling, 50° foveal radius
FPS: **265**

Foveated Culling, 20° foveal radius
FPS: 346

Full path traced scene
FPS: **167**

Foveated Culling, 50° foveal radius
FPS: **170**

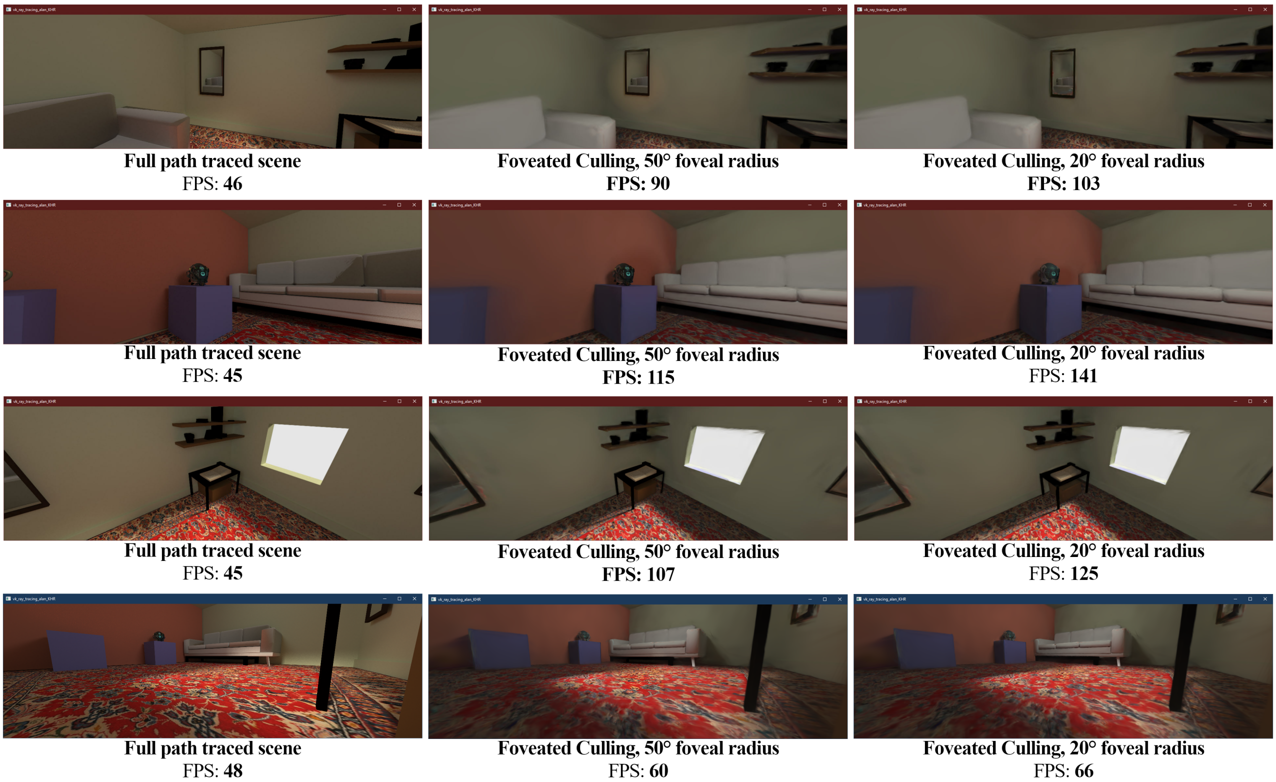
Foveated Culling, 20° foveal radius
FPS: 200

Full path traced scene
FPS: **159**

Foveated Culling, 50° foveal radius
FPS: **129**

Foveated Culling, 20° foveal radius
FPS: **139**

Figure 12. Examples of different angles on the same scene and their respective frame rates during real time rendering in 1280x780. They are organized in three columns: the leftmost images are path traced only; the center images show an application of Foveated Path Culling with the foveal radius of 50% of the screen's height; the rightmost images shows FPC with the foveal radius of 20% of the screen's height. The lower row has an example on how FPC can be outperformed by path tracing when rendering frames in lower resolutions.



Full path traced scene
FPS: **46**

Foveated Culling, 50° foveal radius
FPS: 90

Foveated Culling, 20° foveal radius
FPS: 103

Full path traced scene
FPS: 45

Foveated Culling, 50° foveal radius
FPS: 115

Foveated Culling, 20° foveal radius
FPS: **141**

Full path traced scene
FPS: 45

Foveated Culling, 50° foveal radius
FPS: 107

Foveated Culling, 20° foveal radius
FPS: **125**

Full path traced scene
FPS: **48**

Foveated Culling, 50° foveal radius
FPS: **60**

Foveated Culling, 20° foveal radius
FPS: **66**

Figure 13. Examples of different angles on the same scene and their respective frame rates during real time rendering in 3316x1061. They are organized in three columns: the leftmost images are path traced only; the center images show an application of Foveated Path Culling with the foveal radius of 50% of the screen's height; the rightmost images shows FPC with the foveal radius of 20% of the screen's height. The lower row, with frames equivalent to those that were outperformed by path tracing when rendered in lower resolutions, is an example of FPC's tendency of performing better in higher resolutions.

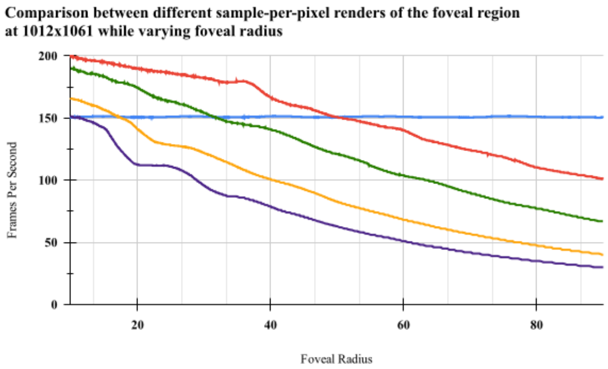


Figure 14. Performance comparison between different sample-per-pixels at the foveal region while varying the foveal radius, rendered at 1012x1061. **In blue:** frame rate of the fully path traced view, static at 150 FPS; **in red:** frame rate of FPC at 1SPP on the foveal region, varying from 200 FPS down to 100 FPS, just as seen on **Figure 9**; **in green:** frame rate of FPC at 2SPP on the foveal radius, starting at 192 FPS and decreasing to 67 FPS; **in orange:** frame rate of FPC at 4SPP on the foveal radius, starting at 166 FPS and decreasing to 40 FPS; **in purple:** frame rate of FPC at 6SPP on the foveal radius, starting at 151 FPS and decreasing to 30 FPS.

166 FPS with 4 SPP (a 17% performance decrease) and 151 FPS with 6 SPP (a 24.5% performance decrease). At the largest foveal radius of 90% of the vertical resolution, these gaps widen: we get a 33% performance decrease with 2 SPP at 67 FPS, a 60% performance decrease with 4 SPP at 40 FPS, and lastly a 70% performance decrease with 6 SPP at 30 FPS.

7 Future Work

Even with this work's advancements, we still have to explore a feasible solution over dynamic scenes, since every experiment seen in both works considered only static scenes. We believe this work leaves the door open for interesting works on mixing dynamic elements into this mixed setup of the same scene rendered through different methods. We believe investigating works such as HexPlanes in Cao and Johnson [2023] and 4D Gaussian Splatting (4DGS) described in Wu *et al*. [2023] are interesting for dynamic scenes, and deserve attention in future endeavors. On the other hand, works such as "ReLight My NeRF"[Toschi *et al*., 2023] and ReNeRF[Xu *et al*., 2023], which present solutions for changing lights with Radiance Fields, must be closely observed, since they offer an interesting solution for approximating real-time lighting for peripheral vision.

It is interesting to explore the impact of using compressed 3DGS to better understand the impact it could have over the overall speedup and user perception. It looks like a natural evolution of this work to use these compressed gaussian splats due to its minimal loss of visual quality and reduced size when compared to the standard 3DGS[Niedermayr *et al.*, 2023], or a solution such as RadSplat, given its outstanding frame rates and visual quality[Niemeyer *et al.*, 2024]. It is our belief that such works may allow less powerful systems to render interactive scenes in real-time.

Even though we are able to properly move the camera in a scene while using Foveated Path Culling at real-time frame rates, we found that the 3DGS model does not fit perfectly with the standard model used on path tracing. For some reason, we had to manually correct the overlapping models during testing in order to produce images, warranting small adjustments whenever we moved the camera. We believe that the reconstructed 3DGS model suffers from distortion due to the SFM points used for optimizing the 3D Gaussians, generating slightly different models from their reference.

Last but not least, we have yet to test this technique when passed down to a real virtual reality headset, and then properly test user perception of Foveated Path Culling powered by 3DGS. User testing may be appropriate once we got a hold of visual quality while using a compressed 3DGS model since the usage of a lighter model for peripheral images is bound to grant us better performance.

8 Conclusion

This work expanded upon the findings of the work “A mixed path tracing and NeRF approach for optimizing rendering in XR Displays” by further exploring the performance enhancements provided by that approach and changing the Radiance Field-based solution employed by it. We also gave the strategy of mixing Radiance Fields and traditional path tracing in order to achieve Foveated Rendering a name: Foveated Path Culling, due to the culling of samples in the periphery of the eye.

In this work, we changed the Neural Radiance Field used in Henriques *et al.* [2023] from [Müller *et al.*, 2022] to 3D Gaussian Splatting due to its visual quality and capacity of outputting frames in real time consistently in high resolutions[Kerbl *et al.*, 2023]. We also present a tool in order to proper render frames using Foveated Path Culling with 3DGS and capture performance information, so we can deepen our understanding on how FPC contributes to the development of rendering techniques for interactive XR experiences.

Within our findings, it has become clear that Foveated Path Culling works as long as one is using a high resolution, so that the path tracing for lighting effects become more expensive than accessing the Gaussian Splatting. We also tested the role of the foveal radius on the performance gain provided by this Foveated Rendering technique, and concluded that it represents a speedup whenever it is as big as theoretically necessary according to the disposition of photorreceptors in the human eye. Last, but not least, we experimented with different number of samples-per-pixels in the foveal region.

We believe that this work is a suitable expansion on the original work, by further proving that Foveated Path Culling is a promising new direction for studies on Foveated Rendering and the integration of Radiance Fields on render pipelines of interactive experiences in Virtual Reality.

Declarations

Acknowledgements

The authors would like to thank Medialab UFF and the students that frequent the laboratory for the continuous help with the development of this study, be them direct or indirectly.

The first author acknowledge the usage of Chat-GPT 3.5 as a tool for reviewing the usage of the English language and suggest in order to deliver a smoother reading experience for the readers, since English is not the authors’ first language. No AI tools were used in any other capability during the production of this text.

Funding

This work was partially funded by Coordenação de Aperfeiçoamento de Pessoal de Nível Superior – Brasil (CAPES) and by FAPERJ - Fundação Carlos Chagas Filho de Amparo à Pesquisa do Estado do Rio de Janeiro.

Authors’ Contributions

HM, EC and DT contributed to the conception of this study. AO is the main developer of the application used for the new experiments, with HM adapting the application for capturing data. EO and HM analyzed the results, with EO orienting on the most suitable statistical methods to employ. HM is the main contributor and writer of this manuscript, as AO contributed to Section 3, and EO contributed to Section 5. HM conducted the experiments. All authors read, reviewed and approved the final manuscript.

Competing interests

The authors declare that they have the following competing interests.

Availability of data and materials

The main application and models used in the experiments are available under request.

References

Albert, R., Patney, A., Luebke, D., and Kim, J. (2017). Latency requirements for foveated rendering in virtual reality. *ACM Transactions on Applied Perception (TAP)*, 14(4):1–13. DOI: https://doi.org/10.1145/3127589.

Alsop, T. (2024). Virtual reality (VR) - statistics & facts. https://www.statista.com/topics/2532/virtual-reality-vr/ (accessed: 17 June 2024).

Barré-Brisebois, C., Halén, H., Wihlidal, G., Lauritzen, A., Bekkers, J., Stachowiak, T., and Andersson, J. (2019). Hybrid rendering for real-time ray tracing. In *Ray Tracing Gems*, pages 437–473. Springer. DOI: https://doi.org/10.1007/978-1-4842-4427-2.

Barron, J. T., Mildenhall, B., Verbin, D., Srinivasan, P. P., and Hedman, P. (2022). Mip-nerf 360: Unbounded anti-aliased neural radiance fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5470–5479. DOI: https://doi.org/10.1109/CVPR52688.2022.00539.

Cao, A. and Johnson, J. (2023). Hexplane: A fast representation for dynamic scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 130–141.

Caulfield, B. (2022). What is path tracing? NVIDIA Blog. https://blogs.nvidia.com/blog/2022/03/23/what-is-path-tracing/ (accessed: 17 June 2024).

Chellappa, R. and Theodoridis, S. (2017). *Academic Press Library in Signal Processing, Volume 6: Image and Video Processing and Analysis and Computer Vision*. Elsevier Science & Technology, Saint Louis. DOI: https://doi.org/10.1016/C2016-0-00726-X.

Chen, A., Xu, Z., Geiger, A., Yu, J., and Su, H. (2022). Tensorf: Tensorial radiance fields.

Chen, Z., Funkhouser, T., Hedman, P., and Tagliasacchi, A. (2023). MobileNeRF: Exploiting the Polygon Rasterization Pipeline for Efficient Neural Field Rendering on Mobile Architectures. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 16569–16578, Vancouver, BC, Canada. IEEE. DOI: https://doi.org/10.1109/CVPR52729.2023.01590.

Duinkharjav, B., Chen, K., Tyagi, A., He, J., Zhu, Y., and Sun, Q. (2022). Color-perception-guided display power reduction for virtual reality. *ACM Trans. Graph. (Proc. SIGGRAPH Asia)*, 41(6):144:1–144:16.

Einhon, E. and Mawdsley, J. (2023). Generate groundbreaking ray-traced images with next-generation nvidia dlss. https://developer.nvidia.com/blog/generate-groundbreaking-ray-traced-images-with-next-generation-nvidia-dlss/ (accessed: 17 June 2024).

Fernandes, F., Castro, D., and Werner, C. (2022). Immersive learning research from svr publications: A re-conduction of the systematic mapping study. *Journal on Interactive Systems*, 13(1):205–220. DOI: https://doi.org/10.5753/jis.2022.2472.

Fridovich-Keil, S., Yu, A., Tancik, M., Chen, Q., Recht, B., and Kanazawa, A. (2022). Plenoxels: Radiance fields without neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5501–5510. DOI: https://doi.org/10.1109/CVPR52688.2022.00542.

Guenter, B., Finch, M., Drucker, S., Tan, D., and Snyder, J. (2012). Foveated 3d graphics. *ACM Transactions on Graphics (TOG)*, 31(6):1–10. DOI: https://doi.org/10.1145/2366145.2366183.

Henriques, H., Oliveira, E., Clua, E., and Trevisan, D. (2023). A mixed path tracing and nerf approach for optimizing rendering in xr displays. In *Proceedings of the 25th Symposium on Virtual and Augmented Reality*, pages 123–130.

Infinite (2024). iNFINITE|VR Headset database and utility. https://www.infinite.cz/projects/HMD-tester-virtual-reality-headset-database-utility (accessed: 17 June 2024).

Jabbireddy, S., Sun, X., Meng, X., and Varshney, A. (2022). Foveated rendering: Motivation, taxonomy, and research directions. *arXiv e-prints*, pages 1–16. DOI: https://doi.org/10.48550/arXiv.2205.04529.

Kajiya, J. T. (1986). The rendering equation. In *Proceedings of the 13th annual conference on Computer graphics and interactive techniques*, pages 143–150. DOI: https://doi.org/10.1145/15922.15902.

Kerbl, B., Kopanas, G., Leimkühler, T., and Drettakis, G. (2023). 3d gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics*, 42(4).

Kilgariff, E., Moreton, H., Stam, N., and Bell, B. (2018). NVIDIA Turing Architecture In-Depth | NVIDIA Technical Blog. https://developer.nvidia.com/blog/nvidia-turing-architecture-in-depth/ (accessed: 17 June 2024).

Kim, J. (2022). Individualized Foveated Rendering. page 60.

Koskela, M., Lotvonen, A., Mäkitalo, M., Kivi, P., Viitanen, T., and Jääskeläinen, P. (2019). Foveated Real-Time Path Tracing in Visual-Polar Space. In Boubekeur, T. and Sen, P., editors, *Eurographics Symposium on Rendering - DL-only and Industry Track*, pages 1–12. The Eurographics Association. DOI: https://doi.org/10.2312/sr.20191219.

Laine, S. and Karras, T. (2011). High-performance software rasterization on gpus. In *Proceedings of the ACM SIGGRAPH Symposium on High Performance Graphics*, pages 79–88.

Levoy, M. and Whitaker, R. (1990). Gaze-directed volume rendering. In *Proceedings of the 1990 symposium on interactive 3d graphics*, pages 217–223. DOI: https://doi.org/10.1145/91385.91449.

Li, Y., Yu, Z., Choy, C., Xiao, C., Alvarez, J. M., Fidler, S., Feng, C., and Anandkumar, A. (2023). Voxformer: Sparse voxel transformer for camera-based 3d semantic scene completion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9087–9098.

Lin, C.-Y., Fu, Q., Merth, T., Yang, K., and Ranjan, A. (2024). Fastsr-nerf: Improving nerf efficiency on consumer devices with a simple super-resolution pipeline. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 6036–6045.

Liu, L., Gu, J., Zaw Lin, K., Chua, T.-S., and Theobalt, C. (2020). Neural Sparse Voxel Fields. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M. F., and Lin, H., editors, *Advances in Neural Information Processing Systems*, volume 33, pages 15651–15663. Curran Associates, Inc.

Mildenhall, B., Srinivasan, P. P., Tancik, M., Barron, J. T., Ramamoorthi, R., and Ng, R. (2021). Nerf: Representing scenes as neural radiance fields for view synthesis. *Communications of the ACM*, 65(1):99–106. DOI: https://doi.org/10.1145/3503250.

Mohanto, B., Islam, A. T., Gobbetti, E., and Staadt, O. (2022). An integrative view of foveated rendering. *Computers & Graphics*, 102:474–501. DOI: https://doi.org/10.1016/j.cag.2021.10.010.

Müller, T., Evans, A., Schied, C., and Keller, A. (2022). Instant neural graphics primitives with a multiresolution hash encoding. *ACM Trans. Graph.*, 41(4):1–15. DOI: https://doi.org/10.1145/3528223.3530127.

Müller, T., Rousselle, F., Novák, J., and Keller, A.

(2021). Real-time neural radiance caching for path tracing. *ACM Transactions on Graphics*, 40(4):1–16. DOI: https://doi.org/10.1145/3450626.3459812.

Niedermayr, S., Stumpfegger, J., and Westermann, R. (2023). Compressed 3d gaussian splatting for accelerated novel view synthesis. *arXiv preprint arXiv:2401.02436*.

Niemeyer, M., Manhardt, F., Rakotosaona, M.-J., Oechsle, M., Duckworth, D., Gosula, R., Tateno, K., Bates, J., Kaeser, D., and Tombari, F. (2024). Radsplat: Radiance field-informed gaussian splatting for robust real-time rendering with 900+ fps. *arXiv.org*.

Porcino, T., Trevisan, D., and Clua, E. (2021). A cybersickness review: causes, strategies, and classification methods. *Journal on Interactive Systems*, 12(1):269–282. DOI: https://doi.org/10.5753/jis.2021.2058.

Reiser, C., Peng, S., Liao, Y., and Geiger, A. (2021). Kilonerf: Speeding up neural radiance fields with thousands of tiny mlps. *arXiv e-prints*, pages 1–11. DOI: https://doi.org/10.48550/arXiv.2103.13744.

Schwarz, K., Sauer, A., Niemeyer, M., Liao, Y., and Geiger, A. (2022). Voxgraf: Fast 3d-aware image synthesis with sparse voxel grids. *arXiv preprint arXiv:2206.07695*, pages 1–22. DOI: https://doi.org/10.48550/arXiv.2206.07695.

Series, B. (2012). Methodology for the subjective assessment of the quality of television pictures. *Recommendation ITU-R BT*, 500:1–46.

Souza, A. M. d. C., Aureliano, T., Ghilardi, A. M., Ramos, E. A., Bessa, O. F. M., and Rennó-Costa, C. (2023). Dinosaurvr: Using virtual reality to enhance a museum exhibition. *Journal on Interactive Systems*, 14(1):363–370. DOI: https://doi.org/10.5753/jis.2023.3464.

Sun, C., Sun, M., and Chen, H.-T. (2022). Direct voxel grid optimization: Super-fast convergence for radiance fields reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5459–5469. DOI: https://doi.org/10.1109/CVPR52688.2022.00538.

Swafford, N. T., Iglesias-Guitian, J. A., Koniaris, C., Moon, B., Cosker, D., and Mitchell, K. (2016). User, metric, and computational evaluation of foveated rendering methods. In *Proceedings of the ACM Symposium on Applied Perception*, pages 7–14, Anaheim California. ACM. DOI: https://doi.org/10.1145/2931002.2931011.

Tanaka, E. H., Almeida, L. d., Gouveia, G. S. d. F., Clerici, R. P. S., Alves, A. H. F., and Oliveira, R. R. d. (2023). A collaborative, immersive, virtual reality environment for training electricians. *Journal on Interactive Systems*, 14(1):59–71. DOI: https://doi.org/10.5753/jis.2023.2685.

Tang, J., Ren, J., Zhou, H., Liu, Z., and Zeng, G. (2023). Dreamgaussian: Generative gaussian splatting for efficient 3d content creation. *arXiv preprint arXiv:2309.16653*.

Toschi, M., De Matteo, R., Spezialetti, R., De Gregorio, D., Di Stefano, L., and Salti, S. (2023). Relight my nerf: A dataset for novel view synthesis and relighting of real world objects. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20762–20772.

Unit (2023). Getting started with ray tracing. Unity Technologies. https://docs.unity3d.com/Packages/com.unity.render-pipelines.high-definition@14.0/manual/Ray-Tracing-Getting-Started.html (accessed: 17 June 2024).

Unity (2024). Unity real-time development platform | 3d, 2d, vr & ar engine. Unity Technologies. https://unity.com/ (accessed: 17 June 2024).

Wang, L., Shi, X., and Liu, Y. (2023). Foveated rendering: A state-of-the-art survey. *Computational Visual Media*, 9(2):195–228. DOI: https://doi.org/10.1007/s41095-022-0306-4.

Wang, Z., Bovik, A. C., Sheikh, H. R., and Simoncelli, E. P. (2004). Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4):600–612. DOI: https://doi.org/10.1109/TIP.2003.819861.

Warburton, M., Mon-Williams, M., Mushtaq, F., and Morehead, J. R. (2023). Measuring motion-to-photon latency for sensorimotor experiments with virtual reality systems. *Behavior Research Methods*, 55(7):3658–3678.

Weier, M., Roth, T., Kruijff, E., Hinkenjann, A., Pérard-Gayot, A., Slusallek, P., and Li, Y. (2016). Foveated Real-Time Ray Tracing for Head-Mounted Displays. pages 289–298. The Eurographics Association and John Wiley & Sons Ltd.. DOI: https://doi.org/10.1111/cgf.13026.

Whitted, T. (1979). An improved illumination model for shaded display. In *Proceedings of the 6th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '79, page 14, New York, NY, USA. Association for Computing Machinery. DOI: https://doi.org/10.1145/800249.807419.

Whitted, T. (1980). An improved illumination model for shaded display. *Commun. ACM*, 23(6):343–349. DOI: https://doi.org/10.1145/358876.358882.

Wu, G., Yi, T., Fang, J., Xie, L., Zhang, X., Wei, W., Liu, W., Tian, Q., and Wang, X. (2023). 4d gaussian splatting for real-time dynamic scene rendering. *arXiv preprint arXiv:2310.08528*.

Xu, Y., Zoss, G., Chandran, P., Gross, M., Bradley, D., and Gotardo, P. (2023). Renerf: Relightable neural radiance fields with nearfield lighting. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 22581–22591.

Ye, J. (2022). Rectangular-Mapping-based-Foveated-Rendering. https://github.com/Bob-Yeah/Rectangular-Mapping-based-Foveated-Rendering (accessed: 17 June 2024).

Ye, J., Xie, A., Jabbireddy, S., Li, Y., Yang, X., and Meng, X. (2022). Rectangular mapping-based foveated rendering. In *2022 IEEE Conference on Virtual Reality and 3D User Interfaces (VR)*, pages 756–764. IEEE. DOI: https://doi.org/10.1109/VR51125.2022.00097.