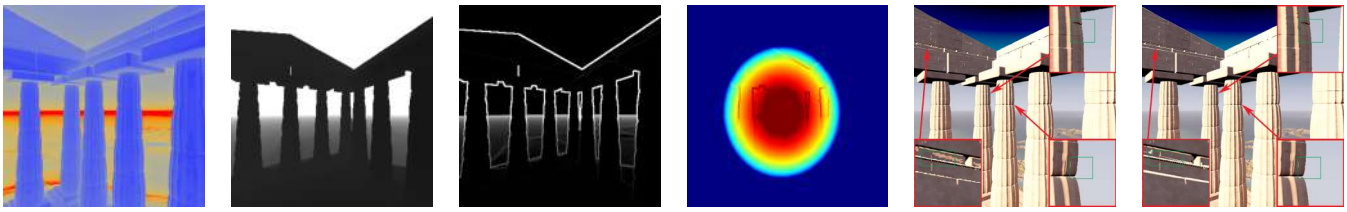
Eurographics Symposium on Rendering (2023)
T. Ritschel and A. Weidlich (Editors)

An Inverted Pyramid Acceleration Structure Guiding Foveated Sphere Tracing for Implicit Surfaces in VR

A. Polychronakis[1] & G. A. Koulieris [2] & K. Mania[1]

[1]TU of Crete, School of Electrical & Computer Engineering, Greece
[2] Durham University, Department of Computer Science, United Kingdom



(a) *Step count* **(b)** *Depth image* **(c)** *Edge map* **(d)** *Probability map* **(e)** *Ours* **(f)** *Ground truth*

Figure 1: *Our method enables foveated rendering for implicit surfaces described using signed distance functions (SDFs). (a) A computationally expensive scene described without any geometry, requires a varying number of steps per pixel to be traced (redder means more steps). The step count often increases near surface boundaries that correspond to edges in the rendered scene. Based on a low cost depth approximation of the scene (b), we detect edges (c), and then use this information to modulate a hyperbolic visual acuity model, guided by eye tracking, to obtain approximate pixel saliency (red is high saliency, eye fixation in the center) (d). Doing so, we establish that we accurately render only the (computationally expensive) edges that* are *detectable in the periphery. (e) We show that with our edge detection step, significant edges still appear even in the periphery in the final output (eye fixation in the center) whereas insignificant outer periphery edges are not rendered. (f) Ground truth image for comparison.*

Abstract

In this paper, we propose a novel rendering pipeline for sphere tracing signed distance functions (SDFs) that significantly improves sphere tracing performance. Previous methods simply focus on over-relaxing the step size by a fixed amount and thus reducing the total step count of the ray based on the error of the previous step at the full rendering resolution. Unlike those, our system reconstructs the final image in a multi-scale inverted pyramid fashion that provides progressively finer approximations of a surface's distance from the camera origin. We initiate sphere tracing at a very low resolution approximation of the scene which provides an initial estimate of the closest surface to a group of rays to be sphere traced. We shoot and trace those rays from that approximated distance instead of shooting them from the camera origin, providing a massive head-start for the rays to leap ahead in the 3D scene, successively generating the following level until the full resolution is reached. This significantly reduces the total step count. Moving up in the pyramid in higher and higher resolutions we repeat this process to further eliminate sphere tracing steps. The multiple resolution levels of the pyramid ascertain that we avoid jumps of the ray in the 3D scene that would potentially generate artefacts, especially around scene edges that might be missed when rendering at lower resolutions. This approach allows for a much more efficient use of computational resources and results in a significant boost in performance (more than 20x speed-up in some cases). Integrating a foveated rendering algorithm within the inverted pyramid pipeline further accelerates performance enabling 16x super-sample anti-aliasing of implicit surfaces in a VR headset. Our experiments demonstrate that our image manipulation remains imperceptible. Our benchmark evaluation indicated a significant boost in sphere tracing performance with or without foveated rendering applied. This enables efficiently rendering SDFs in VR headsets, often otherwise impossible due to limited performance.

1. Introduction

Sphere tracing signed distance fields (SDFs) is an elegant way to render complex surface geometry not described explicitly as a ge-

DOI: 10.2312/sr.20231128

ometric mesh, using instead mathematical equations or occupancy fields rather than polygons. Starting from the camera origin, rays traverse the scene incremented by a step determined by the distance of the current ray position to the nearest implicit surface. The process continues until a threshold is reached, at which point a color is assigned to the corresponding pixel. Sphere tracing SDFs has recently gained popularity due to its efficient memory utilization for rendering either manually or procedurally generated content [HSK89b]. Examples include procedural displacement mapping tools [Don05], artistic rendering [Ini22] or fully connected neural networks learning from multiple images [MST*20; OPG21; PCPM21] to estimate surfaces in those images.

Whereas rendering SDFs is extremely memory efficient as no geometry is being stored, its computational complexity is high when complex surfaces are being rendered. Performance significantly decreases when rays approach *"sharp"* scene element boundaries (edges), as the step size of the traced ray gets smaller and smaller until the nearest SDF is found. In such cases, additional steps are required in order to converge to the object surface (Fig. 2). Sphere tracing SDFs is not currently accelerated at the hardware level in GPUs in a similar way to how ray-tracing is, which involves specialised acceleration structures.

A straightforward way to significantly reduce the computational cost for sphere tracing is to decrease the number of steps required to traverse a scene, whilst maintaining perceived visual fidelity. An effective approach to this involves a fundamental characteristic of the human visual system (HVS), namely the drop of visual acuity as eccentricity (distance away from central gaze direction) increases. Concomitantly, a high level of perceptual sensitivity to edges in the periphery of vision remains [HW62]. A plethora of foveated rendering algorithms improve the performance of real-time rendering techniques by progressively reducing rendering quality in the periphery of vision based on eccentricity, while retaining high fidelity in the fovea, driven by real-time gaze tracking [MDT09; WRK*16; RWH*16]. Other foveated algorithms enhance their foveation models by investigating the sensitivity of the HVS to luminance, contrast, or noise [TAW*19; TTD22]. A transformation of coordinates from Cartesian to polar has also been exploited to maintain a more coherent sampling in the fovea region by reducing the resolution required for rendering [MDZV18; KLM*19].

Edges, sharp surface boundaries in the 3D world, are exceptionally costly for SDF rendering, but at the same time crucial for a high level of perceptual fidelity *even* in the periphery. This "tug-of-war" between reducing the number of steps required to render edges whilst not affecting their appearance significantly, inspired this work. We propose, for the first time, a foveated rendering method for sphere tracing SDFs that exploits this reduction in visual acuity in the periphery to reduce samples, while ensuring that salient edges even in the periphery, are rendered properly. We designed a sphere-tracing pipeline based on multiple rendering passes.

We propose an inverted pyramid rendering that selectively traces parts of the scene, in low quality, while others in high quality, depending on the expected perceptibility of artefacts. Within this architecture, we integrate a foveated rendering model, that, based on eye tracking, determines the pyramid level from which samples will be sourced, while preserving edge information in the 3D environment. We first detect scene edges based on depth information from the bottom level, low-resolution, swiftly rendered tip of the inverted pyramid which approximates the scene. We selectively mask the resolved edges in the depth map based on eye tracking data to obtain a foveation probability map, which is subsequently used to modulate the number of ray-marching steps as a ray approaches a surface. We incorporate variable refresh rate rendering to less perceptible scene areas.

Our perceptual study demonstrated that our rendering is indistinguishable from ground truth full-step sphere tracing, while measuring a remarkable speedup up to 20.4× in sphere tracing performance. Our specific contributions include:

- We develop an inverted pyramid rendering system (IPR) for sphere tracing SDFs. Our method accelerates rendering involving a multi-pass sphere tracing process based on multi-resolution rendering to eliminate sphere traversals that would go unnoticeable.
- We integrate a foveated rendering model to our system (FIPR). We expedite rendering in the visual field periphery based on a physiology-inspired probability model. Our algorithm strikes a balance between effectively rendering detectable edges in the periphery while optimising the associated high cost when sphere tracing edges.
- We perform a perceptual study for our foveated system, both to parametrise (foveal eccentricity angles & Gaussian blur) and evaluate our methodology, to establish that our manipulation is imperceptible.
- We perform a detailed performance assessment of our system, compared to the ground truth, measuring strong performance gains for a variety of scenes.

2. Related work

In this section, we provide an overview of sphere tracing SDFs & eye physiology, and analyze past research on foveated rendering and foveated video streaming.

2.1. Rendering Signed Distance Fields (SDFs)

Sphere tracing was originally proposed to render implicit surfaces [HSK89a] described by signed distance functions (SDFs). SDFs can outline 3D objects or procedural 3D worlds, fractals [HSK89b] as well as generate complex visual effects [ZRL*08]. SDFs, in traditional rasterization, improve displacement mapping [Don05] as well as rendering small-scale details with complex topology in a fragment shader with a few GPU instructions and little memory. Implicit neural representation uses learned SDFs or occupancy fields to represent scene geometry by training a neural network [MST*20; OPG21; PCPM21; RPLG21]. The networks learn from sparse images or point clouds to reconstruct objects, finding the SDF that matches the target object. Manually designing 3D objects and scenes is also possible, by combining simple SDFs for basic primitives (circles, boxes, cones, etc.) using union, difference & intersection operators to create intricate 3D scenes [Ini22]. Distortion or noise can be additionally applied as well as combining primitives. Although complex, dedicated tools can automatically do this [Van].

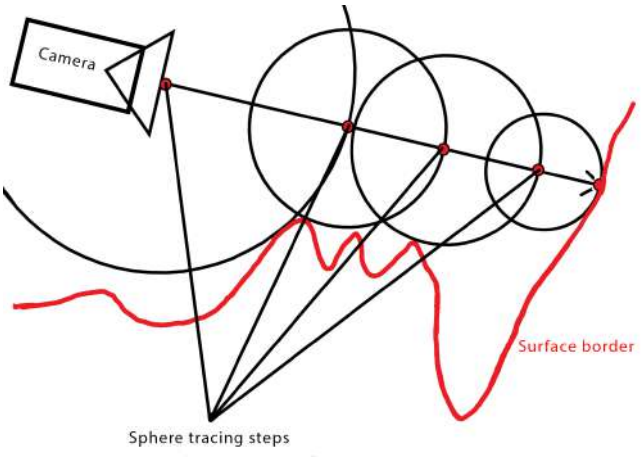


Figure 2: *The rays traverse the world by propagating based on their geometric distance from the closest implicit surface, as described by a signed distance function (SDF).*

Contrary to ray tracing that finds the closest triangle that intersects with a ray, in sphere tracing rays traverse the world propagated based on their geometric distance from the closest *implicit* surface, as described by a signed distance function (SDF): $R^3 \Rightarrow R$ representing the distance from the boundary of an implicit surface $f^-1(0)$. The SDF is a scalar field mapping each point in 3D space to a scalar value, representing the distance of that point to the implicit surface. The points *on* the implicit surface have zero distance, while the points *inside* or *outside* the surface have a negative or positive distance respectively:

$$SDF(x) = \begin{cases} -dist(x, f^-1(0)), \text{ray inside of } f^-1(0) \\ +dist(x, f^-1(0)), \text{ray outside of } f^-1(0) \end{cases}$$

The number of steps required for a ray to intersect with an implicit surface varies depending on scene complexity. The step size is adjusted based on the distance returned by the SDF.

Previous research attempted to reduce the step count [BV18; SN10; KSK*14] by over-relaxing the step size by a fixed amount and thus reducing the total step count of the ray based on the error at full resolution during the rendering pass, leading to increased performance. In this paper, our proposed approach utilises an approximated distance calculated from an inverted pyramid resolution multi-pass rendering approach increasing performance without introducing artefacts.

2.2. Physiology of the eye

The central zone in the human visual field of view (about 1.5–2°) is where foveal vision occurs, i.e., vision of the highest acuity [SRJ11]. Visual acuity corresponds to the ability to distinguish the details of objects and perceive a sharp image. Peripheral, or indirect, vision of increasingly lower acuity extends from the outer limits of foveal vision to the periphery [CSKH90]. This decline follows approximately a hyperbola [SRJ11]. The angular distance from the center of the visual field is known as eccentricity. At an eccentricity of 6°, visual acuity is reduced by 75%. Identifying edges is an essential ability of the HVS to estimate the position of object boundaries. Specific neurons sensitive to edges reside in the visual cortex [HW62]. Visual contrast sensitivity drives visual edge detection and provides insight into multi-channel spatial frequency selection of the visual system. We propose a foveated model that exploits the degraded visual acuity of the HVS in the periphery, while retaining the high-cost but perceptually significant edge information, when rendering SDFs.

2.3. Foveated rendering

Early foveated rendering achieved performance gains by dynamically manipulating geometric level-of-detail (LOD) based on eccentricity, often resulting in aliasing and high latency [OYT96; LHNW00]. Based on real-time gaze tracking, LOD has been reduced in unattended areas while the users performed simple tasks [LM00; CCW03], however, visual artifacts appeared due to unstable frame rates. Saliency models predict gaze position from low-resolution images [LDC06] by extracting features such as luminance and contrast, from task maps [LKC08; HLR*10], as well as from high-level context [KDCM14]. Low-level saliency models alone usually fail to predict gaze direction [SSWR08]. Early work on gaze-contingent displays [DÇ07; BDDG03; KAS*19] did not manipulate spatial resolution based on eccentricity.

Later work reduced the resolution, LOD and refresh rate in the periphery of human vision based on three eccentricity layers [GFD*12; SK97] but artifacts appeared in the periphery. To address this, a Gaussian blur post-processing step progressively increased filter-width based on eccentricity [PSK*16], inducing, though, a sense of tunnel vision that was dealt with by enhancing contrast in the periphery. Recently, a two-pass foveated rendering pipeline transformed Cartesian coordinates of the render frame to kernel log-polar coordinates leading to a lower resolution buffer [MDZV18]. Another foveated method renders with higher eccentricity for the dominant eye [MDV20].

Ray-tracing, unlike rasterization, allows for varied sampling of the output [MDT09; WRK*16; RWH*16] without rendering multiple passes to produce layers of lower quality or interpolating them [GFD*12]. Gaze-contingent ray tracing increases the samples per pixel near object silhouettes and high contrast areas [MDT09] but with visual artifacts in the periphery [FH14]. A foveated ray tracer that uses a sampling mask to generate a non-uniformly distributed set of pixels [SCMP19] avoids artifacts with temporal anti-aliasing. Cost reductions are achieved by generating rays based on a linear model not matching the pattern of the foveal receptor density [WRK*16; RWH*16]. Temporal artifacts appear due to reprojected frames and reconstruction based on a support image. [WRHS18] extends [WRK*16] by integrating a gaze-contingent depth-of-field (DoF) filter concealing artifacts.

A foveated method for path-tracing [KLM*19] generates rays [Kaj86] based on polar coordinates, similar to [MDZV18] and a probability model similar to the foveal receptor density used to reduce the generated rays as eccentricity increased. An emulated foveated path-tracer studied foveated path-tracing of more

than one sample per pixel and with a high number of bounces [PKM21], showing that performance gains can be above 2× - 3×. A luminance and contrast-aware foveated rendering technique [TAW*19] developed a predictor of foveated parameters using a low-resolution frame calculating luminance. Others use pixels from previous frames by spatiotemporally re-projecting them [FFM*21], evaluating artifacts and dis-occlusions based on a confidence value derived from image eccentricity, contrast and holesize. Low confidence areas are redrawn lowering primitive processing and shading cost. A foveated technique based on the eye's ability to detect high frequencies for certain eccentricities has been presented [TTD22]. A foveated volume rendering model adapts samples with Linde-Buzo-Gray sampling and natural neighbor interpolation [BSB*19]. Due to peripheral vision being sensitive to contrast changes and movement, a temporal filter attenuates undersampling artifacts, ignoring peripheral vision except for visual acuity.

Our work, despite using edge detection similarly to previous studies (e.g., [SGEM16; TAW*19]), has some crucial differences: (i) previous approaches focus either on rasterization or ray tracing meshes, whereas our methodology focuses on implicit surfaces described by SDFs; (ii) rasterization and ray tracing can benefit from specific hardware acceleration capabilities of GPUs whereas implicit rendering in its current form, cannot; (iii) we focus specifically on reducing the step count during sphere tracing by positioning the ray closer to the implicit surface thus reducing the number of steps required to accurately render the surface using an innovative inverted pyramid multi-resolution technique; (iv) we do not rely on the use of reprojected frames [WRK*16], which can introduce errors. We focus on accelerating SDF rendering using foveation, enabling their rendering in VR headsets, that is otherwise severely constrained due to performance limitations.

2.4. Foveated streaming

Foveated video coding and compression streaming allocate bits based on the non-uniform resolution of the HVS by compressing more pixels in the periphery and less in the fovea, employing human contrast sensitivity as a function of spatial frequency and retinal eccentricity [GP98]. A foveal visual quality metric, i.e., the signal-to-noise ratio, achieved the best compression and rate control parameters for a target bit rate [LPB01; LPB02]. Foveation filtering (local bandwidth reduction) for foveated compression considered only static foveation mechanisms [LB03]. A wavelet-based distortion visibility measure was used to develop an attention-driven foveated video quality (AFViQ) metric [YEP14]. Pre-encoded videos [RYS*16] as well as real-time 360°video foveated stitching [LCC*17] has been presented.

A foveated video streaming system for cloud gaming requiring low latency, adapted video stream quality by adjusting encoding parameters on the fly to match gaze position [ISM17; IGS*20], based on learning from natural videos. Neural networks drove foveated video streaming (DeepFovea) that recreates a frame using a small fraction of pixels provided in every frame [KSL*19]. Foveated rendering using metamers was recently presented [WDF*21].

3. Implementation

3.1. Overview

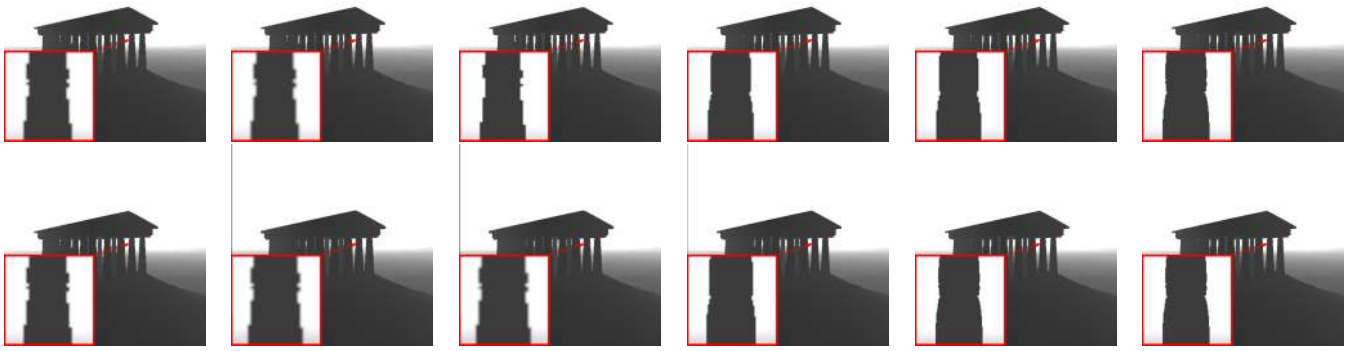
SDF rendering employs tracing rays that progressively approach the closest SDF from the ray origin until a threshold is met and then a color is assigned to the rendered pixel (Fig. 2). When ray marching SDFs, computation increases proportionally to SDF complexity leading to slow rendering times and, subsequently, to lower frame rates, which is particularly problematic for VR. Edges in the field-of-view are particularly salient and at the same time exceptionally costly, as a high number of steps is required to converge to an edge on screen. In the following implementation, we propose a novel approach to accelerate sphere tracing by approximating the distance that a ray will travel in a 3D scene without fully tracing at the full resolution of the frame buffer, while taking into account salient edges. We do this by first utilizing an inverted image pyramid (IPR), enabling reduced resolution passes to successively refine an approximate distance a ray has to travel and subsequently trace from that approximated position in the target resolution, reducing the total step count.

The highest resolution level of the pyramid is the target resolution frame buffer where all shading computation is performed. To find the final implicit surface that requires shading we progressively reconstruct that final image starting from the low resolution tip of the pyramid, which renders at high frame rates. At each level of the pyramid, each ray's distance that we calculate, corresponds to a group of 4 rays/pixels at the next level. We can thus advance 4 rays closer to a surface, rather than starting from the camera origin. This allows each ray to begin tracing (at the subsequent level) from the approximate distance rather than from the camera origin. We repeat this process for subsequent levels of the pyramid, until we reach the top level of the pyramid. As a result the total step count is significantly reduced compared to fully sphere tracing at the target resolution (Fig. 4).

However, since each lower resolution approximation does not accurately contain/represent what the higher resolution does, artifacts appear, because pixels may travel beyond the actual surface and might miss a surface that does not appear at a lower resolution altogether. This can result in low quality rendering, as depicted in Fig. 3. We eliminate those artefacts using a neighbor *min* filter when propagating ray distances within the pyramid (described in subsection 3.2.4). This filter ensures that rays close to surfaces begin tracing before passing the surface, resulting in correct shapes and a higher quality output, as shown in Fig. 3.

Initial testing demonstrated that three levels are sufficient to improve performance, i.e., the bottom level is 1/16th of the target resolution, the middle level is 1/4th of the target resolution. Increasing the number of levels to more than 3, did not increase performance due to an elevated number of rendering calls. IPR demonstrably improves computational efficiency (see Sec. 5.2) for sphere tracing without compromising the rendering quality which was evaluated in a small pilot study (see Sec. 4.3). In particular, IPR achieves 4.28× performance improvement.

To achieve the high frame rates required for VR applications, we subsequently integrate foveated rendering in IPR. Foveated rendering renders at the highest resolution where people fixate and



(a) *Level 3 output* **(b)** *Level 2 input* **(c)** *Level 2 output* **(d)** *Level 1 input* **(e)** *Level 1 output* **(f)** *Ground truth*

Figure 3: *The top row is the inputs and outputs for each level of our inverse pyramid rendering (IPR) without applying the minimum filter. The bottom row shows the inputs and outputs when we apply the minimum filter. The use of the min filter improves the quality of IPR.*

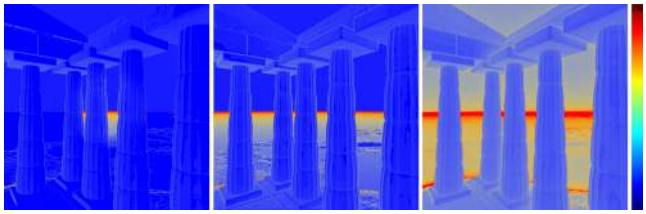


Figure 4: *Left to right, total step count FIPR vs IPR vs ground truth. For IPR & FIPR tracing steps are significantly reduced due to the inverted pyramid acceleration structure and foveated rendering respectively. Blue: 1 step, red: 256 steps.*

gradually reduces the resolution towards the periphery of the fixation. Previous foveated rendering approaches for ray/sphere tracing strive to reduce the number of generated rays with the aim of improving performance. IPR can accommodate foveated rendering depending on where a pixel is in the visual field as it can be directly sampled from a different level in the pyramid. We take this one step further: We want to ascertain that salient edges (which are also particularly costly) are rendered properly. Instead of simply *decreasing* the probability of a pixel to be fully traced as eccentricity increases, which is typical in most foveated methods, we *increase* the probability of pixels in the periphery in areas with prominent edges, ensuring that salient edges in the periphery are properly reconstructed. For such edges, we trace based on the higher resolution level of IPR in the foveal region, where the human eye is most sensitive to detail. We trace rays using the mid-level of IPR for the rest of the frame buffer and then interpolate to the full resolution at a lower refresh rate for those pixels. By doing so, we can maintain high image quality in the foveal region while significantly reducing the computational cost of sphere tracing in the periphery, ultimately allowing us to balance the trade-off between image quality and computational efficiency, leading to more efficient, higher fidelity rendering. We call our full model Foveated Inverted Pyramid Rendering (FIPR).

3.2. Foveated Inverted Pyramid Rendering (FIPR)

FIPR is a five pass pipeline (Fig. 5). Our method was implemented in the Unity 3D Engine using compute shaders and applied as a full-screen quad post-processing effect to the camera. Each pass corresponds to one kernel, implemented using compute shaders. Specifically, the low, mid, and high levels of our inverted pyramid structure, the edge filter, the foveated mask, and the post-processing Gaussian blur effects are all implemented as separate compute shader kernels. The minimum filter is applied to each level (mid / high) of the inverse structure before sphere tracing, thus it is not a separate kernel. We now describe each pass:

3.2.1. Pass 1: Low-res Depth Image

The first sphere tracing pass starts at the base of the inverted pyramid ($1/16^{th}$ of the screen resolution), generating a low resolution depth map (Fig. 6a) storing the marched distance from the camera origin to the closest implicit surface. The distances to surfaces are only accurate for this low resolution, and will act as a rough approximation for subsequent tracing in higher levels. This low resolution approximation becomes the input to a probability model which decides which rays will be spawned (depth image forwarded to pass 2) taking into account the scene's rough edge information, but also guides the reconstruction and used for sampling at the target resolution (forwarded to pass 4).

3.2.2. Pass 2: Salient Edge Detection

In the second pass, we apply Sobel edge detection on the low resolution depth map to gather scene edge information, which is computationally cheap as it is applied at $1/16^{th}$ of the target resolution (Fig. 6b). Texture pixels can take one of two values, 1 (edge) or 0 (not an edge). We then upscale that edge texture to the target rendering resolution and forward it to pass 3.

3.2.3. Pass 3: The Foveation Mask

In the third pass, we create the foveation mask. Eye tracking data from the VR headset provide the fixation point. We generate a hyperbolic eccentricity-based probability map (Eq. 1) corresponding

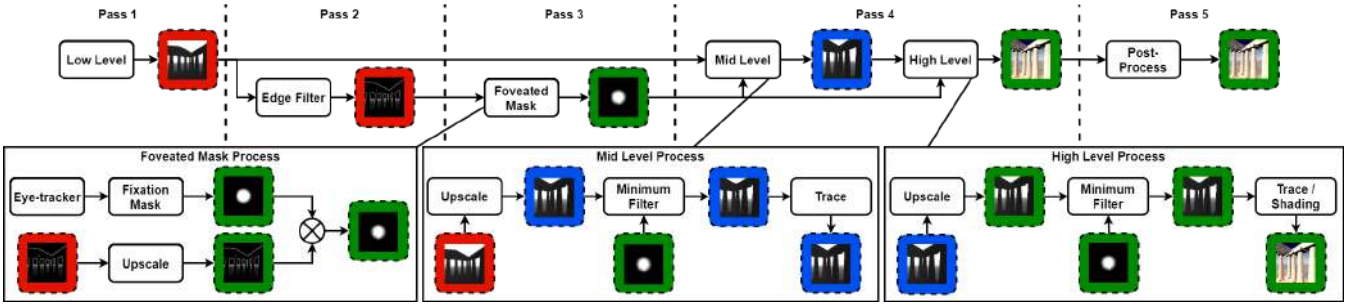
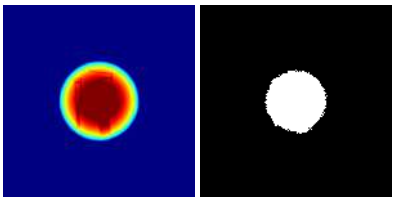


Figure 5: *We accelerate rendering of SDFs by accounting for the foveation angle and edges in the periphery in a five-pass pipeline. The red frame indicates buffers at 1/16th of the screen resolution, blue 1/4th of the screen resolution, green denotes full (output) resolution.*

(a) *Depth map* **(b)** *Edge map* **(c)** *Visual acuity*



(d) *Probability* **(e)** *Foveated mask*

Figure 6: *We generate a binary foveated mask used to decide which pixels are re-marched in the final high quality rendering, as a product of a hyperbolic acuity model and an edge detection map over a computationally cheap depth map.*

to the fall-off of visual acuity from the fixation point to the periphery; measured visual acuity for various human eye eccentricities from [MS47] as fitted by [PKM21]. This hyperbolic fall-off probability map has three distinct zones: fovea, periphery, and outer periphery as in [WRK*16]. In the fovea, probability is set to 1; in the periphery there is a hyperbolic fall-off; in the outer periphery, pixel probability is set to 0.

We subsequently multiply this map with the calculated buffer containing edges from the second pass (Fig. 6e). Multiplying the probability map with the edge detected depth map ensures that even in the periphery, salient edges will be traced at full resolution, consistent with the human visual system's high sensitivity to peripheral edges. The resulting texture indicates the final probability assigned for each pixel to be sphere traced in the high level of our IPR, as seen in Fig. 6d.

We then construct a binary foveated mask indicating pixels of high importance to be queued for sphere tracing at the highest level only if $\xi_q < p(x,y)$ with $\xi_q \in [0,1]$ being a uniformly distributed random number. This converts the probability into a Boolean True

$$p(\omega) = 0.90964e^{\frac{\omega}{2.9661}} + 0.00792 \quad (1)$$

1: *$p(\omega)$ is the probability of the pixel, and ω is the eccentricity angle of the pixel.*

or False value indicating whether a pixel needs to be traced at full resolution, as seen in Fig. 6e. Non-foveal regions and peripheral regions without edges are rendered at a lower resolution, significantly accelerating throughput. The foveated mask is forwarded to both passes 4 and 5.

3.2.4. Pass 4: Rendering & Reconstruction

In the fourth pass, we assemble the complete frame using the high fidelity (foveal/edge regions) pixels and the (lower refresh rate) sampled peripheral pixels. The depth texture from pass 1 and the binary foveated mask from pass 3 are inputs for pass 4. We first sphere trace in the mid-level of our acceleration structure where the resolution is now equal to 1/4th of the target frame buffer. We first check whether a pixel is of high importance in the binary mask; if it is, we check its neighbours and apply a minimum filter to the depth buffer - this is the distance from which the ray will start tracing. The minimum filter simply replaces the pixel's depth value with the minimum value (i.e., distance/depth) of that pixel and its neighbors (3x3 filter, see Sec. 4).

Rationale: The minimum filter alleviates a possible artefact of IPR. When rendering at lower resolutions, some surfaces might be completely missed, while they can be visible at the higher target resolution. The minimum filter pushes a ray's tracing origin a bit further back towards the camera origin, ensuring that when the ray is marched it will not be placed behind a surface and thus miss it altogether. Using a minimum filter, pixels that had traversed almost the same distance as their neighboring pixels in the 3D environment will remain close to the surface based on the approximate distance rather than resetting further back to the camera origin, regardless of the scene configuration. This eliminates artefacts and improves performance due to each pixel repositioning itself based on information gathered from neighboring pixels, compared to using a fixed offset as in [BV18; SN10; KSK*14]. When the minimum filter is not applied, this leads to faster convergence to the surface but a surface could be missed. As the size of the minimum filter increases,

quality is further improved, but performance drops. We discuss a possible limitation of this approach in Sec. 5.3.

Sphere tracing starts from the generated approximate distance recorded in the depth texture. This reduces the steps that a ray will have to march as compared to a ray being spawned from the camera origin. At the mid-level of the acceleration structure, a new depth texture is produced based on the new approximated distance which is then forwarded to the highest level of the acceleration structure. At the highest level, rendered at the target resolution, after having calculated the final distance to the closest surface for each pixel, we apply shading calculations to high importance pixels. For low importance pixels, we reduce the sampling refresh rate based on which approximate distance is calculated, to increase overall rendering performance, since those pixels mostly go unnoticed [GFD*12; MIGS22]. The final texture is a composite of both high-fidelity and low-fidelity pixels, as guided by the foveation mask. This texture is forwarded to pass 5 for post-processing.

3.2.5. Pass 5: Post-Processing

Most foveated rendering pipelines use post-processing effects to hide, reduce, or eliminate visual artefacts in the periphery due to the lower fidelity rendering; usually a Gaussian blur filter [PSK*16; PKM21; SCMP19]. In the fifth (and last) pass, we apply a Gaussian blur guided by the foveated mask. This process smooths any visible transitions between the high-fidelity and low-fidelity areas. The foveated mask from pass 3 and the texture inclusive of shading from pass 4 are inputs to this pass. Based on the foveated mask, we apply a Gaussian blur to the transition zones between low and high fidelity pixels. The final image is a composite of both high-fidelity and low-fidelity pixels, as indicated by the foveation mask. In our study (see Sec. 4) we investigated if a Gaussian blur filter is helpful when we apply foveated rendering to hide peripheral artefacts.

(a) *Greek Temple* **(b)** *Column & Lights* **(c)** *Primitives*

Figure 7: *Scenes used in the evaluation, by Inigo Quilez [Ini22], with permission.*

4. Evaluation

4.1. Overview

We conducted two two-alternative forced choice (2AFC) studies. The first (pilot) study aims to investigate whether the perceived quality of our non-foveated acceleration structure (IPR) is similar to ground truth, i.e., it does not introduce any artefacts. The second study aims to parametrize our foveated method by establishing parameters for which the drop in quality in the periphery of vision remains unnoticeable.

We used simplified versions of high-quality scenes by Inigo Quilez (with permission [Ini22], Fig. 7), to maintain acceptable frame rates, i.e., by removing soft shadows and ambient occlusion. The first scene named "Primitives", represents an open environment that renders primitive shapes on a plane with a checkerboard pattern for the ground. The second scene named "Column & Lights", features a more complex environment in a closed setting. This scene includes a repeating pattern of columns and five point-lights that move around the columns. Finally, the third scene named "A Greek Temple", represents an open cultural heritage environment including a procedurally generated cliff terrain and a temple located on a cliff. These scenes depict complex indoor/outdoor environments without explicit geometry and are described using SDFs.

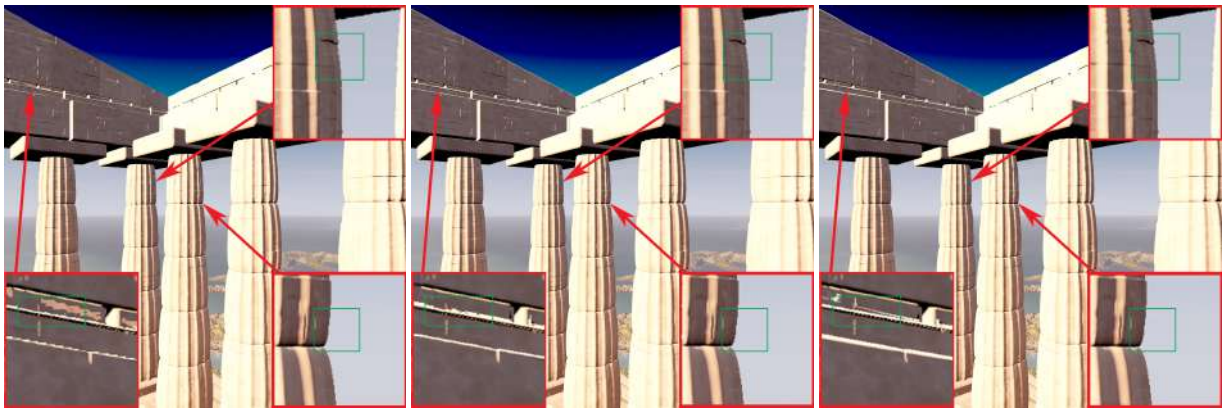
4.2. Hardware Setup

We used an HTC Vive Pro Eye Head-mounted Display (HMD), with a resolution of 1440 $\times$ 1600 per eye, i.e., combined resolution 2880 $\times$ 1600 and a refresh rate of 90Hz. The embedded eye tracker on our HMD has a sampling rate of 120Hz and an accuracy of 0.5°0.5 - 1.1°in the central field of view. The computer used had an Intel i9-12900F CPU, 32GB RAM, and a single Nvidia GPU RTX-3070ti with 8GB RAM. Eye tracking is not integrated into most desktop setups as, recently, in VR headsets. Therefore, foveated rendering in VR is meaningful also due to the much higher rendering cost in HMDs compared to desktop displays and the readily available eye-tracking. Desktop setups can of course employ our method using an external desktop eye tracker.

4.3. First (Pilot) Study: Inverse Pyramid Rendering (IPR)

The first pilot study compared the quality of the Inverse Pyramid Rendering with ground truth rendering, where all pixels are traced from the camera origin in a single pass. In this study, we wanted to establish that the acceleration structure does not introduce any visible artefacts. At the same time, we benchmarked our structure to demonstrate that it can perform faster than standard sphere tracing. Pilot testing indicated that the optimal value for the filter size is 3x3. If we increase the filter size, the performance drops without any noticeable quality improvement. If the filter size is small the quality is worse than the ground truth as surfaces might be missed.

We conducted a 2AFC pair experiment with participants wearing the HTC Vive Pro Eye HMD and using an Xbox controller to compare the quality for a series of trials. Each sequence pair consisted of the ground truth rendering and our non-foveated IPR, in random order. Participants were asked "Was the quality the same in both sequences?". For this pilot study we recruited 5 participants from our campus, with an average age of 28.5y (SD 1.5y) and a normal or corrected-to-normal vision. For each scene, there were 6 trials. It took approximately 8 minutes for each participant to complete the experiment. Users uniformly found the quality produced by our non-foveated proposed IPR to be the identical to the ground truth. This established that the IPR structure *does not* introduce artefacts and we could proceed with introducing and parametrising foveated rendering in the main study.



(a) *Foveated IPR (FIPR)* **(b)** *Non-foveated IPR* **(c)** *Ground truth*

Figure 8: *Quality comparison between foveated inverse pyramid rendering (FIPR), inverse pyramid (IPR), and ground truth rendering. Notice how IPR and ground truth treat edges identically. FIPR distorts insignificant edges in the periphery (eye fixation at the center of the image).*

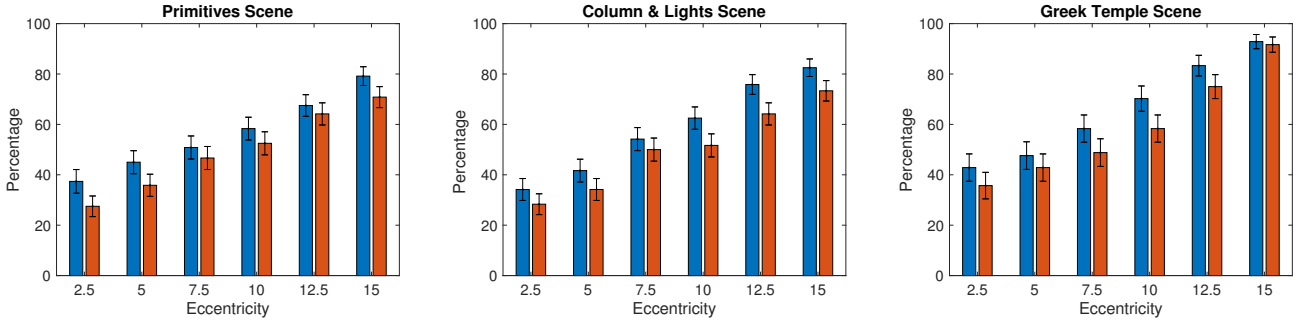


Figure 9: *User study results. The blue bars correspond to the percentage of trials indicating that foveated inverse pyramid rendering (FIPR) without Gaussian blur is identical to ground truth, and the orange bar corresponds to the percentage of trials indicating that foveated inverse pyramid rendering (FIPR) with Gaussian blur is identical to ground truth.*

4.4. Second (Main) Study: Foveated Inverse Pyramid Rendering (FIPR)

The second study aimed to evaluate whether the drop of quality in the periphery of vision is noticeable, and determine the optimal parametrisation values (dependent variables), i.e., the eccentricity angle of the fovea and the presence or not of Gaussian blur, for our foveated inverse pyramid rendering (FIPR), so that the quality reduction remains imperceptible. Please see the limitations sections as to why additional variables (refresh rate, super-sampling) were not included in the study (see Sec. 5.3). We employed the same two-alternative forced choice (2AFC) experimental methodology as in the pilot study. Fovea eccentricity varied from 2.5 to 15 degrees in steps of 2.5. Each pair to be compared consisted of the ground truth rendering and the FIPR method.

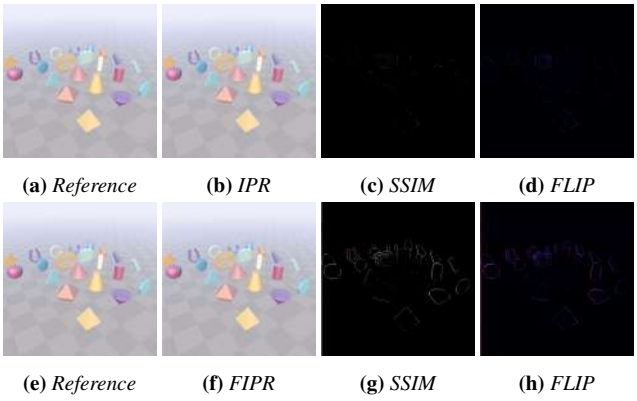
We evaluated the performance gains when rendering in VR for the three scenes. We calculated the time to render a frame with our method. We compared it with the time it takes to render the ground truth (full sphere tracing). The target resolution expected by the HTC Vive Pro Eye due to pinchusion pre-distortion correction, even though the specifications report a resolution of 1440×1600 (a full resolution of 2880×1600) is equal to 2468×2740 per eye (full resolution of 4936×2740). We could not perform any direct performance comparison with previous foveated rendering techniques as they did not involve sphere tracing of SDFs sequences, making the comparison invalid, since we are not using geometry as in [SGEM16; TAW*19; KVJT16; WRK*16]. Previous work also took advantage of hardware acceleration for mesh rendering (e.g., geometry shaders and ray tracing acceleration structures) to increase performance. Readily available hardware acceleration structures are not available for sphere tracing. Implementing an apples-to-apples comparison was therefore impossible. Instead, we thoroughly gauged the performance of our technique both in terms of visual fidelity and computational performance against ground truth rendering, measuring substantial performance gains.

Participants A total of 20 participants (6 female) with normal or corrected-to-normal vision took part in the study, with an average age of 28.5y (SD 2.5y). 72 trials were presented for each scene (totaling 216 trials) and the average time it took for each participant to complete each scene session was approximately 15 minutes (45

minutes in total). A 10-minute resting session between sessions was allocated to reduce fatigue. Before the experiment started, each user performed an eye tracker calibration.

4.5. Objective SSIM & FLIP Metrics

We utilized the SSIM [WBSS04] and FLIP [ANA*20] metrics to assess the image quality of our IPR and FIPR methods compared to the ground truth obtained through full sphere tracing. Fig. 10 presents a visual representation of the local SSIM values and an image with the differences produced by FLIP and table 1 shows the global SSIM value and the mean error of the FLIP for each scene.



(a) *Reference* **(b)** *IPR* **(c)** *SSIM* **(d)** *FLIP*

(e) *Reference* **(f)** *FIPR* **(g)** *SSIM* **(h)** *FLIP*

Figure 10: *Visualization of the local SSIM and FLIP difference/error map for the primitives scene where the top row shows the results for ground truth vs IPR and the bottom row for ground truth vs FIPR. Practically no difference for ground truth vs IPR. Zooming-in can reveal details for ground truth vs FIPR; the maps were not processed/optimised otherwise to not introduce artificial* error *pixels.*

	G. Temple	Col. & Lig.	Primit.
GT vs IPR SSIM	0.99579	0.99192	0.99889
GT vs IPR FLIP	0.011108	0.046423	0.050560
GT vs FIPR SSIM	0.9777	0.94939	0.9902
GT vs FIPR FLIP	0.030102	0.077929	0.109450

Table 1: *Evaluating the visual fidelity of IPR and FIPR with respect to ground truth. The global SSIM value indicates high structural similarity and the mean error measured by the FLIP metric approaches zero for all scenes in IPR, indicating a very high similarity to ground truth. In FIPR, global SSIM decreases and FLIP mean error increases for all scenes, as expected, due to the foveated rendering's quality drop in the periphery. These results were obtained with an eccentricity foveal setting of 7.5°.*

5. Results & Discussion

5.1. Visual fidelity

In Fig. 8, we present a visual comparison of our inverse pyramid rendering (IPR) with and without foveated rendering as compared to the ground truth. Further examples exist in the supplemental video.

We present the results of the user study in Fig. 9. For eccentricity values over 7.5° users could not identify any difference in quality between our FIPR method and the ground truth. Users tended to prefer the ground truth rendering over our method when the eccentricity was lower, and that preference towards the ground truth surprisingly increased when blur was applied in the periphery, i.e., blur made the manipulation more apparent.

There were slight variations in the results across the three scenes (Fig. 7). Users commented that in the "Primitives" scenes which featured primitive shapes described by SDFs, their attention was primarily focused on the sharp shape edges. Consequently, they were able to identify minor artefacts when low eccentricities were used and concluded that the drop in quality was significant compared to the ground truth.

Similarly, in the "Column & Lights" scene, users reported that when their gaze was focused on a column, they were able to detect a decrease in quality in the surrounding columns for low eccentricities. This can be attributed to the fact that both scenes have geometries with closely positioned edges.

In contrast, in the "Greek temple" scene almost all users found the quality to be the same as the ground truth for higher eccentricity values to a percentage above 80% compared to the other scenes. This was possibly due to the scene featuring fewer, larger shapes producing less noticeable quality transitions. It is evident from these results that the users found the quality to be the same when the eccentricity is equal or higher than 7.5°and that blur was not required, as blur potentially signaled that quality had dropped.

We hypothesize that this might occur due to the increased complexity of the "Greek Temple" compared to simpler scenes such as "Primitives" which drove users to fixate on a single point of the scene. Users were more sensitive to details such as the primitive's shape and observed drops in quality that may have occurred by focusing on the primitives' shapes. We hypothesize that with faster rendering and a more responsive eye tracker in a VR headset (see Sec. 5.3), the drop in quality will not be noticeable due to the faster update of gaze fixation on the screen. Our findings underscore the need for further research to explore these variations and determine the optimal approach to use for various scenes.

5.2. Performance

In Fig. 11, we present the measured performance improvements for IPR without foveated rendering. The speed-up varies from 1.22× to 4.28× as the samples per pixel increase to enable super-sample anti-aliasing (SSAA). In Fig. 12, we present the render time for each scene for various eccentricity levels, samples per pixel, and refresh rates for the low importance pixels for IPR when foveated rendering is applied.

Our results showcase a significant improvement in rendering performance with a speed-up that varies from 1.22× to 20.04× as the eccentricity level drops from 15°to 2.5°, the samples per pixel increase from 1 to 16 and the refresh rate of the low significance

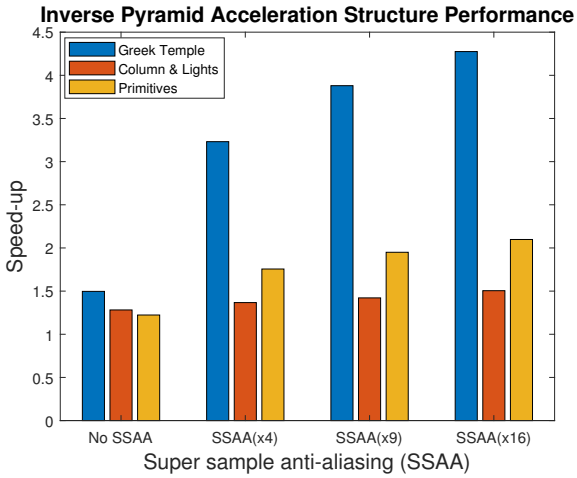


Figure 11: *Speed-up of inverse pyramid rendering (IPR) without foveation, with super-sampling varying from 1 to 16.*

pixels is reduced to 1/2 or 1/4 of the refresh rate of the pixels in the fovea when using IPR.

FIPR showed significant improvements in performance for the highly complex 'Greek Temple' scene, even when the samples per pixel were set to their lowest value, and the refresh rate for the periphery was lower. This was because of the higher complexity of the scene compared to the other two scenes, and our foveated inverse pyramid rendering achieved a speedup ranging from 1.23× to 12.65×. In the 'Column & Lights' scene, because of the repetitive nature of the geometry and high saliency of edges, our method achieved a speedup ranging from 1.08× to 10.01× when the samples per pixel were more than 1 or the refresh rate in the periphery was different from the fovea. Similarly, in the 'Primitives' scene, our method achieved a speedup in performance that ranged from 1.37× to 20.04×, except when no super-sample anti-aliasing was applied, and the refresh rate at the periphery was the same as the fovea (Fig. 11).

Based on the lowest acceptable eccentricity value (Eccentricity = 7.5°) where quality is perceived to be identical to the ground truth, we achieve a speedup ranging from 1.1× to 10.29× for the "Column & Lights" scene, 1.1× to 16.03× for the "Primitives" scene and 1.58× x to 8.83× for the "Greek Temple" scene as the samples per pixel increased from 1 to 16 and the refresh rate for periphery was reduced from 1 to 1/4 of the refresh rate of the fovea area.

Our inverse pyramid rendering is demonstrably highly effective in rendering complex scenes that render at very low frame rates or may not even run at all in VR. Our proposed inverse pyramid rendering demonstrated better performance when the 3D environments comprised of open areas such as mountains or terrains compared to enclosed areas such as caves or dungeons. This is probably due to the lower frequency content in open spaces containing less sharp edges in the open environments. Said another way, in open environments, more rays are not passing close to surfaces compared to enclosed spaces which results in fewer steps and a faster approximation of surface distance for most rays.

Our inverse pyramid rendering even without foveated rendering can significantly improve performance both for one sample-per-pixel or multiple samples-per-pixel. In both cases significantly fewer rays are spawned and marched due to the pyramid and many rays are terminated much faster due to sampling from different levels of the pyramid.

Overall, our inverse pyramid rendering with or without foveation has demonstrated significant improvements in rendering complex VR scenes, achieving faster performance and more accurate approximations of surface distance, even in complex environments.

5.3. Limitations

Our proposed depth filtering method does not guarantee preservation of all fine details at a coarse level. Our user study comparing IPR to ground truth reassured us that any missing details from high-frequency content were indeed imperceptible. The SSIM and FLIP metrics ran corroborate our findings that there exist only inconsiderable differences between the images. In addition, our methods focused on ray marching geometric edges that have a severe adverse effect on rendering performance. But the textural or shading edges can also be jarring to the user; our method was not designed to accelerate those types of edges since they do not affect performance. But, our method readily supports accounting for them in the edge map by using a low-res albedo map or low-res fully shaded map (in addition to the depth map) during the edge processing step. However, this has zero positive effect on performance and the generation of those maps introduces additional rendering costs.

While our proposed method achieves much higher performance (as discussed in sec. 5.2), the HTC Vive Pro Eye has a limitation of locking the frame rate to 45 frames per second (fps) if the frame rate drops below 60 FPS. This limitation may prevent our method from reaching its full potential for VR applications, being unable to update foveated rendering at the measured eye fixation rate, i.e., at a lower refresh rate.

In the implementation section (3), we mention that the refresh rate of the approximate distances for the non-foveal/non-edge pixels can be reduced so that the overall performance of sphere tracing is improved. We excluded the refresh rate variable from the experiment due to the -sometimes- low fps in VR (the headset locking fps down to 45, mentioned above) creating noticeable latency and, thus, visible artifacts especially when sudden movements occur. Furthermore, the samples-per-pixel increase was not included due to the ground truth's extreme drop in performance, which would produce motion sickness.

On a desktop setup, pre-testing using a random fixation pattern indicated that the refresh rate could be lowered down to 1/4 of the foveal refresh rate without introducing perceptible artefacts. Further evaluation is required with a desktop eye tracker to determine whether the drop in refresh rate in the periphery is noticeable.

We detected an average latency of 22.25ms which seems to have an effect on our results due to the slow detection of the gaze position in the headset. Even though the eye tracker refresh rate of the HTC Vive Pro Eye is set to 120Hz, it is insufficient for tracking fast saccades [PKM21]. Even when using the higher eccentricity

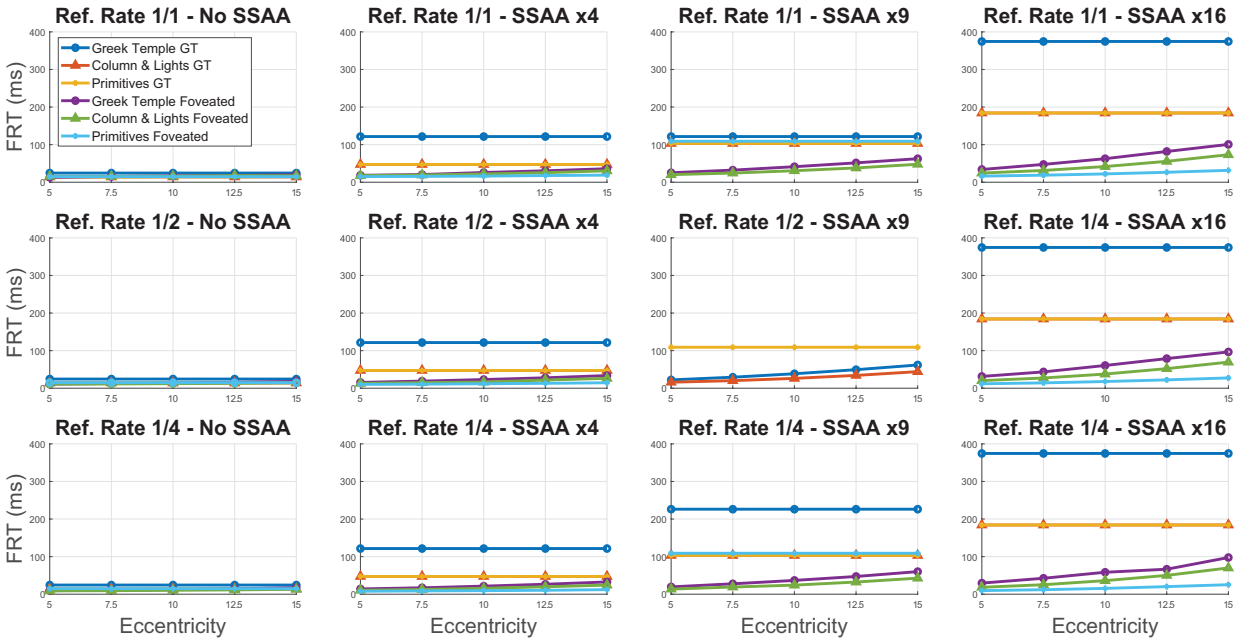


Figure 12: *Rendering time for stereo rendering (two viewports) for each scene, foveated and non-foveated (ground truth) in milliseconds. From left to right, super-sampling varies from 1 to 16. From top to bottom, the refresh rate outside the fovea varies from 1/1 to 1/4 the rate of the fovea. Our method enables interactive rendering for scenes that would otherwise be impossible to run in VR.*

angle (15°), if our subjects performed relatively rapid saccades, the alterations due to foveated rendering were sometimes visible. We hypothesize that subjects selected a much higher eccentricity angle threshold than they would have otherwise due to latency limitations of the headset. We reckon that this will become a non-issue for faster computers and more modern eye tracked headsets in the future.

6. Conclusion

We introduced IPR for rendering implicit surfaces using sphere tracing. We explored the introduction of foveated rendering to IPR, creating FIPR, considering the necessity for rendering high fidelity edges in the periphery. Through a perceptual evaluation, we estimated parameters for our method to ensure that our manipulation remains imperceptible. We also benchmarked the performance of our IPR/FIPR methods compared to ground truth. Our analysis indicated a significant boost in sphere tracing performance using both our methods (i.e., with or without foveation), enabling rendering of complex SDF-based scenes in VR that would otherwise be impossible.

Future work could include the use of neural networks to predict the distance and direction of a ray based on low resolution depth/distance maps, as well as extending our current inverted pyramid acceleration structure to handle shading effects such as sphere-tracing soft-shadows, ambient occlusion, sub-scattering surfaces, and semi-transparent materials.

Acknowledgement

This research forms part of the project 3D4DEPLHI, co-financed by the European Union and Greek funds through the Operational Program Competitiveness, Entrepreneurship and Innovation, under the call 'Specific Actions, Open Innovation for Culture' (project code: T6YBΠ-00190)

References

[ANA*20] ANDERSSON, PONTUS, NILSSON, JIM, AKENINE-MÖLLER, TOMAS, et al. "FLIP: A Difference Evaluator for Alternating Images". *Proceedings of the ACM on Computer Graphics and Interactive Techniques* 3.2 (2020), 15:1–15:23. DOI: 10.1145/3406183 9.

[BDDG03] BAUDISCH, PATRICK, DECARLO, DOUG, DUCHOWSKI, ANDREW T, and GEISLER, WILSON S. "Focusing on the essential: considering attention in display design". *Communications of the ACM* 46.3 (2003), 60–66 3.

[BSB*19] BRUDER, VALENTIN, SCHULZ, CHRISTOPH, BAUER, RUBEN, et al. "Voronoi-Based Foveated Volume Rendering". *EuroVis 2019 - Short Papers*. Ed. by JOHANSSON, JIMMY, SADLO, FILIP, and MARAI, G. ELISABETA. The Eurographics Association, 2019. ISBN: 978-3-03868-090-1. DOI: 10.2312/evs.20191172 4.

[BV18] BÁLINT, CSABA and VALASEK, GÁBOR. "Accelerating Sphere Tracing". *EG 2018 - Short Papers*. Ed. by DIAMANTI, OLGA and VAXMAN, AMIR. The Eurographics Association, 2018. DOI: 10.2312/egs.20181037 3, 6.

[CCW03] CATER, KIRSTEN, CHALMERS, ALAN, and WARD, GREG. "Detail to attention: exploiting visual tasks for selective rendering". *ACM International Conference Proceeding Series*. Vol. 44. 2003, 270–280 3.

[CSKH90] CURCIO, CHRISTINE A, SLOAN, KENNETH R, KALINA, ROBERT E, and HENDRICKSON, ANITA E. "Human photoreceptor topography". *Journal of comparative neurology* 292.4 (1990), 497–523 3.

[DÇ07] DUCHOWSKI, ANDREW T and ÇÖLTEKIN, ARZU. "Foveated gaze-contingent displays for peripheral LOD management, 3D visualization, and stereo imaging". *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)* 3.4 (2007), 1–18 3.

[Don05] DONNELLY, WILLIAM. "Per-pixel displacement mapping with distance functions". *GPU gems* 2.22 (2005), 3 2.

[FFM*21] FRANKE, LINUS, FINK, LAURA, MARTSCHINKE, JANA, et al. "Time-Warped Foveated Rendering for Virtual Reality Headsets". *Computer Graphics Forum* 40.1 (2021), 110–123. DOI: https://doi.org/10.1111/cgf.14176 4.

[FH14] FUJITA, MASAHIRO and HARADA, TAKAHIRO. "Foveated real-time ray tracing for virtual reality headset". *Light Transport Entertainment Research* (2014) 3.

[GFD*12] GUENTER, BRIAN, FINCH, MARK, DRUCKER, STEVEN, et al. "Foveated 3D Graphics". *ACM Trans. Graph.* 31.6 (Nov. 2012). ISSN: 0730-0301. DOI: 10.1145/2366145.2366183 3, 7.

[GP98] GEISLER, WILSON S. and PERRY, JEFFREY S. "Real-time foveated multiresolution system for low-bandwidth video communication". *Human Vision and Electronic Imaging III*. Ed. by ROGOWITZ, BERNICE E. and PAPPAS, THRASYVOULOS N. Vol. 3299. International Society for Optics and Photonics. SPIE, 1998, 294–305. DOI: 10.1117/12.320120 4.

[HLR*10] HILLAIRE, SEBASTIEN, LECUYER, ANATOLE, REGIA-CORTE, TONY, et al. "A real-time visual attention model for predicting gaze point during first-person exploration of virtual environments". *Proceedings of the 17th acm symposium on virtual reality software and technology*. 2010, 191–198 3.

[HSK89a] HART, J. C., SANDIN, D. J., and KAUFFMAN, L. H. "Ray Tracing Deterministic 3-D Fractals". *Proceedings of the 16th Annual Conference on Computer Graphics and Interactive Techniques*. SIGGRAPH '89. New York, NY, USA: Association for Computing Machinery, 1989, 289–296. ISBN: 0897913124. DOI: 10.1145/74333.74363 2.

[HSK89b] HART, J. C., SANDIN, D. J., and KAUFFMAN, L. H. "Ray Tracing Deterministic 3-D Fractals". *SIGGRAPH Comput. Graph.* 23.3 (July 1989), 289–296. ISSN: 0097-8930. DOI: 10.1145/74334.74363 2.

[HW62] HUBEL, DAVID H and WIESEL, TORSTEN N. "Receptive fields, binocular interaction and functional architecture in the cat's visual cortex". *The Journal of physiology* 160.1 (1962), 106 2, 3.

[IGS*20] ILLAHI, GAZI KARAM, GEMERT, THOMAS VAN, SIEKKINEN, MATTI, et al. "Cloud Gaming with Foveated Video Encoding". *ACM Trans. Multimedia Comput. Commun. Appl.* 16.1 (Feb. 2020). ISSN: 1551-6857. DOI: 10.1145/3369110 4.

[Ini22] INIGO QUILEZ. https://iquilezles.org/demoscene/. Accessed: 2022-09-30. 2022 2, 7.

[ISM17] ILLAHI, GAZI, SIEKKINEN, MATTI, and MASALA, ENRICO. "Foveated video streaming for cloud gaming". *2017 IEEE 19th International Workshop on Multimedia Signal Processing (MMSP)*. 2017, 1–6. DOI: 10.1109/MMSP.2017.8122235 4.

[Kaj86] KAJIYA, JAMES T. "The Rendering Equation". *SIGGRAPH Comput. Graph.* 20.4 (Aug. 1986), 143–150. ISSN: 0097-8930. DOI: 10.1145/15886.15902 3.

[KAS*19] KOULIERIS, GEORGE ALEX, AKŞIT, KAAN, STENGEL, MICHAEL, et al. "Near-eye display and tracking technologies for virtual and augmented reality". *Computer Graphics Forum*. Vol. 38. 2. Wiley Online Library. 2019, 493–519 3.

[KDCM14] KOULIERIS, GEORGE ALEX, DRETTAKIS, GEORGE, CUNNINGHAM, DOUGLAS, and MANIA, KATERINA. "C-LOD: Context-aware material level-of-detail applied to mobile graphics". *Computer Graphics Forum*. Vol. 33. 4. Wiley Online Library. 2014, 41–49 3.

[KLM*19] KOSKELA, MATIAS, LOTVONEN, ATRO, MÄKITALO, MARKKU, et al. "Foveated Real-Time Path Tracing in Visual-Polar Space". *Eurographics Symposium on Rendering - DL-only and Industry Track*. Ed. by BOUBEKEUR, TAMY and SEN, PRADEEP. The Eurographics Association, 2019. ISBN: 978-3-03868-095-6. DOI: 10.2312/sr.20191219 2, 3.

[KSK*14] KEINERT, BENJAMIN, SCHÄFER, HENRY, KORNDÖRFER, JOHANN, et al. "Enhanced Sphere Tracing". *Smart Tools and Apps for Graphics - Eurographics Italian Chapter Conference*. Ed. by GIACHETTI, ANDREA. The Eurographics Association, 2014. ISBN: 978-3-905674-72-9. DOI: 10.2312/stag.20141233 3, 6.

[KSL*19] KAPLANYAN, ANTON S., SOCHENOV, ANTON, LEIMKÜHLER, THOMAS, et al. "DeepFovea: Neural Reconstruction for Foveated Rendering and Video Compression Using Learned Statistics of Natural Videos". *ACM Trans. Graph.* 38.6 (Nov. 2019). ISSN: 0730-0301. DOI: 10.1145/3355089.3356557 4.

[KVJT16] KOSKELA, MATIAS, VIITANEN, TIMO, JÄÄSKELÄINEN, PEKKA, and TAKALA, JARMO. "Foveated path tracing". *International Symposium on Visual Computing*. Springer. 2016, 723–732 8.

[LB03] LEE, SANGHOON and BOVIK, A.C. "Fast algorithms for foveated video processing". *IEEE Transactions on Circuits and Systems for Video Technology* 13.2 (2003), 149–162. DOI: 10.1109/TCSVT.2002.808441 4.

[LCC*17] LEE, WEI-TSE, CHEN, HSIN-I, CHEN, MING-SHIUAN, et al. "High-resolution 360 Video Foveated Stitching for Real-time VR". *Computer Graphics Forum* 36.7 (2017), 115–123. DOI: https://doi.org/10.1111/cgf.13277 4.

[LDC06] LONGHURST, PETER, DEBATTISTA, KURT, and CHALMERS, ALAN. "A gpu based saliency map for high-fidelity selective rendering". *Proceedings of the 4th international conference on Computer graphics, virtual reality, visualisation and interaction in Africa*. 2006, 21–29 3.

[LHNW00] LUEBKE, DAVID, HALLEN, BENJAMIN, NEWFIELD, DALE, and WATSON, BENJAMIN. *Perceptually driven simplification using gaze-directed rendering*. Tech. rep. Tech. Rep. CS-2000-04, Department of Computer Science, University of . . ., 2000 3.

[LKC08] LEE, SUNGKIL, KIM, GERARD JOUNGHYUN, and CHOI, SEUNGMOON. "Real-time tracking of visually attended objects in virtual environments and its application to LOD". *IEEE Transactions on Visualization and Computer Graphics* 15.1 (2008), 6–19 3.

[LM00] LOSCHKY, LESTER C and MCCONKIE, GEORGE W. "User performance with gaze contingent multiresolutional displays". *Proceedings of the 2000 symposium on Eye tracking research & applications*. 2000, 97–103 3.

[LPB01] LEE, SANGHOON, PATTICHIS, M.S., and BOVIK, A.C. "Foveated video compression with optimal rate control". *IEEE Transactions on Image Processing* 10.7 (2001), 977–992. DOI: 10.1109/83.931092 4.

[LPB02] LEE, SANGHOON, PATTICHIS, M.S., and BOVIK, A.C. "Foveated video quality assessment". *IEEE Transactions on Multimedia* 4.1 (2002), 129–132. DOI: 10.1109/6046.985561 4.

[MDT09] MURPHY, HUNTER A, DUCHOWSKI, ANDREW T, and TYRRELL, RICHARD A. "Hybrid image/model-based gaze-contingent rendering". *ACM Transactions on Applied Perception (TAP)* 5.4 (2009), 1–21 2, 3.

[MDV20] MENG, XIAOXU, DU, RUOFEI, and VARSHNEY, AMITABH. "Eye-dominance-guided Foveated Rendering". *IEEE Transactions on Visualization and Computer Graphics* 26.5 (2020), 1972–1980. DOI: 10.1109/TVCG.2020.2973442 3.

[MDZV18] MENG, XIAOXU, DU, RUOFEI, ZWICKER, MATTHIAS, and VARSHNEY, AMITABH. "Kernel foveated rendering". *Proceedings of the ACM on Computer Graphics and Interactive Techniques* 1.1 (2018), 1–20 2, 3.

[MIGS22] MOHANTO, BIPUL, ISLAM, ABM TARIQUL, GOBBETTI, ENRICO, and STAADT, OLIVER. "An integrative view of foveated rendering". *Computers & Graphics* 102 (2022), 474–501. ISSN: 0097-8493. DOI: https://doi.org/10.1016/j.cag.2021.10.010 7.

[MS47] MANDELBAUM, JOSEPH and SLOAN, LOUISE L. "Peripheral Visual Acuity*: With Special Reference to Scotopic Illumination". *American Journal of Ophthalmology* 30.5 (1947), 581–588 6.

[MST*20] MILDENHALL, BEN, SRINIVASAN, PRATUL P., TANCIK, MATTHEW, et al. "NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis". *ECCV*. 2020 2.

[OPG21] OECHSLE, MICHAEL, PENG, SONGYOU, and GEIGER, ANDREAS. "UNISURF: Unifying Neural Implicit Surfaces and Radiance Fields for Multi-View Reconstruction". *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. Oct. 2021, 5589–5599 2.

[OYT96] OHSHIMA, TOSHIKAZU, YAMAMOTO, HIROYUKI, and TAMURA, HIDEYUKI. "Gaze-directed adaptive rendering for interacting with virtual space". *Proceedings of the IEEE 1996 Virtual Reality Annual International Symposium*. IEEE. 1996, 103–110 3.

[PCPM21] PUMAROLA, ALBERT, CORONA, ENRIC, PONS-MOLL, GERARD, and MORENO-NOGUER, FRANCESC. "D-NeRF: Neural Radiance Fields for Dynamic Scenes". *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2021, 10318–10327 2.

[PKM21] POLYCHRONAKIS, ANDREAS, KOULIERIS, GEORGE ALEX, and MANIA, KATERINA. "Emulating Foveated Path Tracing". *Motion, Interaction and Games*. MIG '21. Virtual Event, Switzerland: Association for Computing Machinery, 2021. ISBN: 9781450391313. DOI: 10.1145/3487983.3488295 4, 6, 7, 10.

[PSK*16] PATNEY, ANJUL, SALVI, MARCO, KIM, JOOHWAN, et al. "Towards Foveated Rendering for Gaze-tracked Virtual Reality". *ACM Trans. Graph.* 35.6 (Nov. 2016), 179:1–179:12. ISSN: 0730-0301. DOI: 10.1145/2980179.2980246 3, 7.

[RPLG21] REISER, CHRISTIAN, PENG, SONGYOU, LIAO, YIYI, and GEIGER, ANDREAS. "Kilonerf: Speeding up neural radiance fields with thousands of tiny mlps". *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2021, 14335–14345 2.

[RWH*16] ROTH, THORSTEN, WEIER, MARTIN, HINKENJANN, ANDRÉ, et al. "An analysis of eye-tracking data in foveated ray tracing". *2016 IEEE Second Workshop on Eye Tracking and Visualization (ETVIS)*. IEEE. 2016, 69–73 2, 3.

[RYS*16] RYOO, JIHOON, YUN, KIWON, SAMARAS, DIMITRIS, et al. "Design and Evaluation of a Foveated Video Streaming Service for Commodity Client Devices". *Proceedings of the 7th International Conference on Multimedia Systems*. MMSys '16. Klagenfurt, Austria: Association for Computing Machinery, 2016. ISBN: 9781450342971. DOI: 10.1145/2910017.2910592 4.

[SCMP19] SIEKAWA, ADAM, CHWESIUK, MICHAŁ, MANTIUK, RADOSŁAW, and PIÓRKOWSKI, RAFAŁ. "Foveated Ray Tracing for VR Headsets". *MultiMedia Modeling*. Ed. by KOMPATSIARIS, IOANNIS, HUET, BENOIT, MEZARIS, VASILEIOS, et al. Cham: Springer International Publishing, 2019, 106–117. ISBN: 978-3-030-05710-7 3, 7.

[SGEM16] STENGEL, MICHAEL, GROGORICK, STEVE, EISEMANN, MARTIN, and MAGNOR, MARCUS. "Adaptive Image-Space Sampling for Gaze-Contingent Real-time Rendering". *Computer Graphics Forum* 35.4 (2016), 129–139. DOI: https://doi.org/10.1111/cgf.12956 4, 8.

[SK97] SEMWAL, SUDHANSHU KUMAR and KVARNSTROM, HAKAN. "Directed safe zones and the dual extent algorithms for efficient grid traversal during ray tracing". *Graphics Interface*. Vol. 97. 1997, 76–87 3.

[SN10] SINGH, JAG MOHAN and NARAYANAN, P. J. "Real-Time Ray Tracing of Implicit Surfaces on the GPU". *IEEE Transactions on Visualization and Computer Graphics* 16.2 (2010), 261–272. DOI: 10.1109/TVCG.2009.41 3, 6.

[SRJ11] STRASBURGER, HANS, RENTSCHLER, INGO, and JÜTTNER, MARTIN. "Peripheral vision and pattern recognition: A review". *Journal of vision* 11.5 (2011), 13–13 3.

[SSWR08] SUNDSTEDT, VERONICA, STAVRAKIS, EFSTATHIOS, WIMMER, MICHAEL, and REINHARD, ERIK. "A psychophysical study of fixation behavior in a computer game". *Proceedings of the 5th symposium on Applied perception in graphics and visualization*. 2008, 43–50 3.

[TAW*19] TURSUN, OKAN TARHAN, ARABADZHIYSKA-KOLEVA, ELENA, WERNIKOWSKI, MAREK, et al. "Luminance-Contrast-Aware Foveated Rendering". *ACM Trans. Graph.* 38.4 (July 2019). ISSN: 0730-0301. DOI: 10.1145/3306346.3322985 2, 4, 8.

[TTD22] TARIQ, TAIMOOR, TURSUN, CARA, and DIDYK, PIOTR. "Noise-based Enhancement for Foveated Rendering". *arXiv preprint arXiv:2204.04455* (2022) 2, 4.

[Van] VANCO, MATEJ. *Unity - Raymarcher* 2.

[WBSS04] WANG, ZHOU, BOVIK, A.C., SHEIKH, H.R., and SIMONCELLI, E.P. "Image quality assessment: from error visibility to structural similarity". *IEEE Transactions on Image Processing* 13.4 (2004), 600–612. DOI: 10.1109/TIP.2003.819861 9.

[WDF*21] WALTON, DAVID R, DOS ANJOS, RAFAEL KUFFNER, FRISTON, SEBASTIAN, et al. "Beyond blur: Real-time ventral metamers for foveated rendering". *ACM Transactions on Graphics* 40.4 (2021), 1–14 4.

[WRHS18] WEIER, MARTIN, ROTH, THORSTEN, HINKENJANN, ANDRÉ, and SLUSALLEK, PHILIPP. "Foveated Depth-of-Field Filtering in Head-Mounted Displays". *ACM Trans. Appl. Percept.* 15.4 (Sept. 2018). ISSN: 1544-3558. DOI: 10.1145/3238301 3.

[WRK*16] WEIER, MARTIN, ROTH, THORSTEN, KRUIJFF, ERNST, et al. "Foveated Real-Time Ray Tracing for Head-Mounted Displays". *Computer Graphics Forum* 35.7 (Oct. 2016), 289–298. DOI: 10.1111/cgf.13026 2–4, 6, 8.

[YEP14] YOU, JUNYONG, EBRAHIMI, TOURADJ, and PERKIS, ANDREW. "Attention Driven Foveated Video Quality Assessment". *IEEE Transactions on Image Processing* 23.1 (2014), 200–213. DOI: 10.1109/TIP.2013.2287611 4.

[ZRL*08] ZHOU, KUN, REN, ZHONG, LIN, STEPHEN, et al. "Real-time smoke rendering using compensated ray marching". *ACM SIGGRAPH 2008 papers*. 2008, 1–12 2.