Foveated Ray Tracing for VR Headsets

Adam Siekawa, Michał Chwesiuk, Radosław Mantiuk(✉), and Rafał Piórkowski

West Pomeranian University of Technology, Szczecin,
al. Piastów 17, 70-310 Szczecin, Poland
rmantiuk@zut.edu.pl

Abstract. In this work, we propose a real-time foveated ray tracing system, which mimics the non-uniform and sparse characteristic of the human retina to reduce spatial sampling. Fewer primary rays are traced in the peripheral regions of vision, while sampling frequency for the fovea region traced by the eye tracker is maximised. Our GPU-accelerated ray tracer uses a sampling mask to generate a non-uniformly distributed set of pixels. Then, the regular Cartesian image is reconstructed based on the GPU-accelerated triangulation method with the barycentric interpolation. The temporal anti-aliasing is applied to reduce the flickering artefacts. We perform a user study in which people evaluate the visibility of artefacts in the peripheral region of vision where sampling is reduced. This evaluation is conducted for a number of sampling masks that mimic the sensitivity to contrast in the human eyes but also test different sampling strategies. The sampling that follows the gaze-dependent contrast sensitivity function is reported to generate images of the best quality. We test the performance of the whole system on the VR headset. The achieved frame-rate is twice higher in comparison to the typical Cartesian sampling and cause only barely visible degradation of the image quality.

1 Introduction

Rendering algorithms use sampling in the regular Cartesian coordinates. Since the rendered image is supposed to be displayed on a flat and rectangular display it is an intuitive choice of sample distribution for raster images. With the increasing popularity of Virtual Reality (VR) and the head-mounted displays (HMD), often called *VR headsets*, non-uniform sample distribution strategies are applicable. HMDs use a spherically distorted image to compensate for the optical distortion of the lenses. It suggests that the sample distribution in HMDs can be combined with the *foveated rendering*, in which the number of samples is reduced in the peripheral regions of vision. The image is rendered with the highest sampling rate in the surrounding of the observer's gaze point but sampling is reduced with eccentricity (i.e. distance from the fovea). This degradation of the image quality is unnoticeable for the human observer because the human visual system (HVS) has a lower resolution at peripheral angles of view [23].

I. Kompatsiaris et al. (Eds.): MMM 2019, LNCS 11295, pp. 106–117, 2019.
https://doi.org/10.1007/978-3-030-05710-7_9

Foveated rendering is crucial for future VR headsets because their current resolution is well below the resolution of the human retina. For example, resolution of the HTC Vive headset is less than 5 cycles-per-degree (cpd) while the resolution of the human retina in fovea is almost 60 cpd [15]. Contemporary computer graphics technologies are not ready for a 12x increase in image resolution without significant degradation of the graphics quality [8]. VR headsets start to be equipped with the *eye trackers* that capture the gaze direction of the observer (e.g. FOVE, HTC VIVE with the SMI eye tracker, Oculus Rift with the Pupil Labs eye tracker). This information combined with the head position captured by the *head tracker* delivers an accurate location of the gaze point.

In this work, we present a foveated rendering system based on the ray tracing technique. We use information about the gaze direction captured by the eye tracker to render an image with spatially varying sample distribution. In the region surrounding the gaze point, the rays are traced for each pixel in the display but in the peripheral regions, some pixels are skipped. During reconstruction, vertices of the triangles are placed at corresponding sample positions in the screen space. Then, this triangle mesh is rendered using GPU and the sampling holes are filled by barycentric interpolation of the surrounding pixels.

Ray tracing generates high-quality images with accurate reflections, refractions and shadows, which results in photorealistic appearance and high emersion in the virtual environment. In this work, we propose our custom implementation of the ray tracer, which, due to the reduced number of samples, works in real time even for high-resolution display in the VR headset.

We perform an experiment, in which the Cartesian ray tracing is replaced with the foveated rendering in real time during the free-viewing task. Four different sampling scenarios have tested that increase the number of samples in fovea or at the periphery of vision. The results of the experiment show that more than the double reduction of samples can be acceptable for the human observers, even in the easy to notice the case of a dynamic change in the sampling method.

In Sect. 2, we present previous work on the foveated rendering. Section 3 describes our gaze-depended ray tracer - we present sampling and reconstruction techniques as well as details on the implementation of the ray tracer. Section 4 presents performed experiments that evaluate visible deterioration of the image quality caused by the reduced sampling in the peripheral regions of vision. The paper ends with conclusions and future work in Sect. 5.

2 Previous Work

A known approach is to use an eye tracker to reduce the computational complexity of the image synthesis.

An example is the gaze-driven level-of-detail (LOD) technique, in which simplification of the object geometry is driven by the angular distance from the object to gaze direction [14].

Watson et al. [24] studied a possible spatial and chrominance complexity degradation with the eccentricity in the screen space rather than the object space. They used different high detail inset sizes to generate a high-resolution inset within a low-resolution display field. The perception of a target object among distractors was tested for different peripheral resolutions. Experiments performed using head-mounted display revealed that the complexity can be reduced by almost half without perceivable degradation of the image quality.

Levoy and Whitaker in [9] proposed a ray tracer for volumetric data in which both distributions of rays traced through the image plane and distribution of samples along each ray are functions of local retinal acuity. As a result, the resolution of the rendered images varies locally in response to changes in the user's gaze direction. In a practical implementation, a 2D mipmap was generated by downsampling the original image. For each target pixel, which size varies according to the distance to the gaze position, rays are cast from four corners of each pixel from two MIP-map levels - falling just above and just below the desired target pixel size. A single colour is computed for the pixel by interpolating between colours returned by all rays. Traditional rendering of volume data involves the accumulation of voxel information along with a ray cast into the data set. In their ray tracer, the volume data was structured in a 3D mipmap. A sample for one ray was computed by interpolating between two adequate levels of this 3D mip-map (more precisely between the nearest eight voxels from each level). Again, the size of the 3D levels depends on the distance between the ray and gaze position.

For the ray casting, the gaze-dependent sampling was proposed by Murphy et al. [13]. A similar solution was used to accelerate the ambient occlusion algorithm [12]. The reduced number of rays was traced to approximate the occlusion coefficients in the peripheral regions. Günter et al. [7] proposed a rendering engine, which generates three low-resolution images corresponding to the different fields of view. Then, the wide-angle images are magnified and combined with the non-scaled image of the area surrounding the gaze point. Thus, the number of processed pixels can be reduced by 10–15 times, while ensuring the deterioration of image quality invisible for the observer. Another technique proposed by Stengel et al. [21] aims to reduce shading complexity in the deferred shading technique [2]. The spatial sampling is constant for the whole image but the material shaders are simplified for peripheral pixels. According to the authors, this technique reduces the shading time up to 80%. Programmable control of the shading rate, which enables efficient shading for the foveated rendering was also proposed in Vaidyanathan et al. [22]. In Patney et al. [16], a postprocess contrast enhancement in the peripheral region was introduced to reduce a sense of tunnel vision and further reduce the number of samples. They noticed that people tolerated up to 2x larger blur radius before detecting differences from a non-foveated ground truth. A novel multi-resolution and saccade-aware temporal antialiasing algorithm were also proposed.

A simple gaze-dependent ray tracer was presented at the non-peer-reviewed student conference [19]. In this ray tracer, spatial sampling of the primary rays is based on the shape of the gaze-dependent contrast sensitivity function. A similar approach was presented by Fujita and Harada [5]. More recently, Weier et al. [25] proposed combining foveated rendering based on the ray tracing with reprojection rendering using previous frames in order to reduce the number of new image samples per frame. In their work, the reprojection is also used to reconstruct the Cartesian image. Our solution has similar functionality but uses a much simpler approach. We reduce reconstruction to a simple barycentric interpolation.

3 Foveated Rendering

In this section, we present our gaze-dependent rendering system (see Fig. 1). The non-uniform sampling mask is used to trace primary rays with varying spatial distribution. Location of the mask centre is changed according to the gaze direction captured by the eye tracker. Rays are shot through vertices of the mask and traced using our real-time foveated ray tracer. Finally, the Cartesian RGB image is reconstructed from randomly and non-linearly distributed samples. This image is displayed in the VR headset. The camera is changed according to the head movement of the observer.

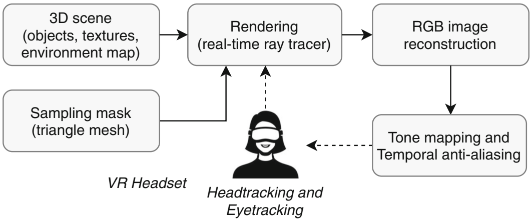
3D scene
(objects, textures, environment map)
Rendering
(real-time ray tracer)
RGB image reconstruction
Sampling mask
(triangle mesh)
Tone mapping and Temporal anti-aliasing
VR Headset
Headtracking and Eyetracking

Fig. 1. High-level architecture of our gaze-dependent rendering system.

Non-uniform Sampling. The sampling mask is delivered as a mesh of triangles with samples located in the vertices of the triangles. Spatial distribution of the samples is defined based on the characteristic of the human retina. The mask is larger than the observer's field of view because it must compensate for the eye movements. The peripheral regions of the mask are uncovered when observers shift her/his eyes to the borders of the visual field. The centre of the mask is always located in the gaze position.

The eyes resolution at different viewing angles is measured using Gabor pattern (sinusoidal grating in the Gaussian envelope) presented to human observers in various eccentricities [17]. Observers are asked to guess the orientation of the stimulus (horizontal or vertical), while the contrast threshold (the contrast between light and dark bars of the sinusoidal grating) is decreased in the consecutive steps of the experiment. The threshold contrast sensitivity is indicated by the inability to distinguish the orientation of the stimulus. We repeated this experiment following the methodology presented in Chwesiuk and Mantiuk [3]. As explained in Loschky et al. [10], the resulting contrast thresholds can be expressed as the cut-off spatial frequency (see Fig. 2). This cut-off frequency defines the maximum number of samples visible for the human observer for a given eccentricity. We use this representation to create the sampling masks with varying spatial distribution of samples.

A mask prepared for HTC Vive display is presented in Fig. 3 (left). Because of the limited resolution of the headset display, there is a white spot in the centre of the mask, which covers all the available pixels.

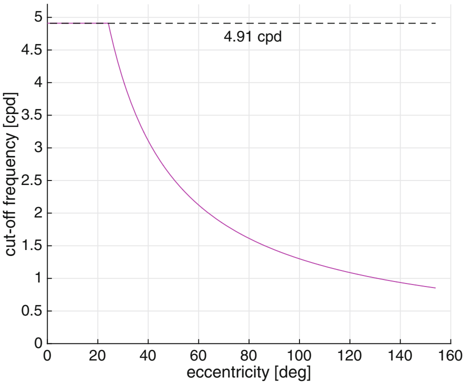

Fig. 2. Cut-off spatial frequency for human observer as the function of the eccentricity. Magenta line shows measured human sensitivity to contrast. The dashed horizontal line depicts the maximum frequency of the display (HTC Vive headset). (Color figure online)

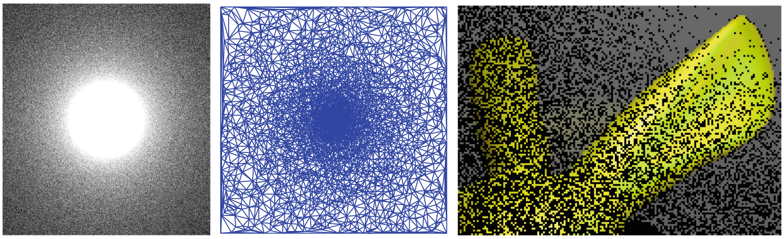

Fig. 3. Left: sampling mask corresponding to the function from Fig. 2. Centre: example triangle mesh used during rendering (for clarity of presentation, the number of vertices was reduced to below seven thousand). Right: inset of the RGB image consisting of the quasi-randomly distributed samples (the black areas indicate pixels for which no rays have been traced). (Color figure online)

Implementation Details. The basic assumption of the foveated rendering systems is real-time work. To accomplish this goal we implemented a custom ray tracer executed by a GPU. The ray tracer uses a non-uniform distribution of the primary rays depending on the observer's gaze direction, which changes continuously while using the VR headset. Actual ray tracing is performed by the OpenCL kernels and RadeonRays [1] routines. The sparse set of samples is converted into the RGB image during the reconstruction phase implemented in OpenGL. The final image is displayed by OpenVR [4]. To speed up rendering, we compute only one ray bounce.

The VR headset visualisation suffers from strong temporal aliasing, which is particularly visible in areas with a reduced number of samples. Therefore, we implemented a naïve temporal anti-aliasing (TAA) based on the depth information from both current and previous frames. The previous frame is accumulated, then, the TAA algorithm looks for pixel coordinates in this accumulation buffer that correspond to the same pixel in the current buffer. The colours from previous and current buffers are averaged.

Image Reconstruction. Ray tracing is not limited to the uniform sampling schema, because one has a full control over the ray origin and direction. This allows rendering images based on the non-uniform sampling algorithms with a negligible impact on the rendering performance. Since the ray tracing performance depends on the number of traced rays, a sample distribution that reduces the overall number of rays will benefit in the increased performance. However, the next step is required to transform the spatially non-uniform samples to the Cartesian coordinates, which can be displayed on the screen. The goal is to use a reconstruction technique, which introduces the lowest possible distortions to the original signal and does not affect the overall performance significantly.

The non-uniform map of samples can be triangulated and rendered using the standard forward rendering [20]. The triangulation is a time-consuming process, which is hard to execute in real time. Therefore, we generate the triangle mesh in the preprocessing and then this mesh is applied for image reconstruction during actual rendering.

The sampling mask is converted to the triangle mesh using the Delaunay triangulation technique. Each sample in the map becomes a vertex in the mesh (see example in Fig. 3 (centre)). This mesh is read from the file during initialisation of our real-time ray tracer. Ray tracer traces rays passing through the vertices of the mesh and stores colours of the corresponding pixels. During the actual triangle mesh rendering, colours inside the triangles are interpolated in screen space using the barycentric interpolation.

We also tested more complex reconstruction techniques: the push-pull reconstruction technique introduced by Gortler et al. [6] and the cell maps described in [20]. However, the mentioned techniques do not improve the image quality significantly but introduce a significant performance overload.

4 Experimental Evaluation

We performed perceptual experiments to explore whether the visible quality of foveated rendering and the quality of the full-resolution rendering are similar.

More precisely, the goal of the experiments was to find the larges sampling reduction that would be acceptable for people watching the rendered animation in VR headset.

Stimuli. We prepared two scenes - Air Shed and Bunny Box - that were displayed in subsequent sessions of the experiment. More realistic Air Shed scene (see Fig. 4, left) consisted of complex models and photorealistic textures. Bunny Box (see Fig. 4, right) was an artificial scene with simple objects and textures that do not mask deteriorations caused by the reduced sampling.

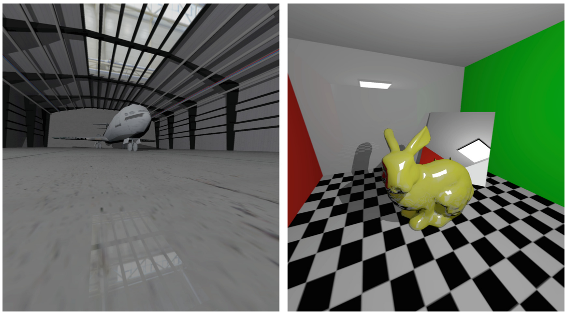

Fig. 4. Example rendering of Air Shed (left) and Bunny Box (right) scenes.

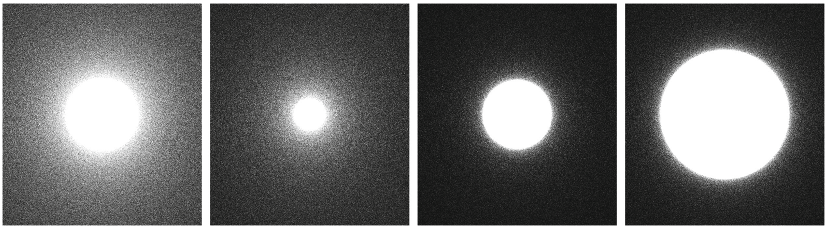

Fig. 5. From left: sampling masks with 54%, 27%, 18%, and 41% of samples.

During the experiment session, the spatial distribution of samples (i.e. primary rays) was modified using four different sampling masks. We also used the reference sampling mask with one ray per pixel, which did not require the reconstruction phase. The sampling masks presented in Fig. 5, correspond to functions plotted in Fig. 6. The 54% mask was created based on the measured gaze-dependent contrast sensitivity function. This function is plotted as the magenta line in Fig. 6. A value of 54% means that the mask defines 54% of the total number of the primary rays required for the full resolution/reference rendering. This 54% mask almost by half reduces the number of primary rays. The second mask defines 27% of samples, which means that the number of samples has been reduced more than three times. There are fewer samples in both fovea and peripheral regions (see blue line in Fig. 6). For the 18% mask, the number of samples has been reduced more than five times but the number of samples in the fovea is almost the same as for the 54% mask (see green line in Fig. 6). In the 41% mask, the densely sampled centre region was extended, while the number of samples in peripheral regions was strongly reduced (see black line in Fig. 6). These four masks were chosen to test different cases of the sampling distributions. Especially, different size of the fovea region was evaluated.

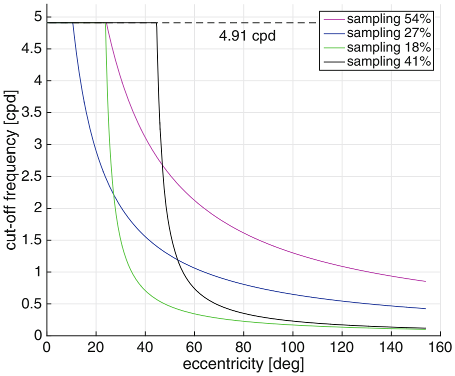

Fig. 6. Cut-off spatial frequency for human observer as the function of eccentricity. Magenta, blue, green, and black lines present simulated sensitivity for 54%, 27%, 18%, and 41% of samples, respectively. The dashed horizontal line depicts the cut-off frequency of the display (HTC Vive headset). (Color figure online)

Figure 7 shows example renderings of Air Shed and Bunny Box scenes for different sampling masks. As can be seen in the insets, the non-uniform distribution of samples causes visible degradation of the image quality increasing with the eccentricity.

Procedure and Participants. We asked observers to wear VR headset and freely look around the scene. At the beginning of the experiment, the reference image was displayed. After 20 s the reference image was replaced with the image generated using a randomly selected mask. The masks were changed at random intervals of about 5 s to the end of the session lasting 180 s. The observer task was to press the mouse button as soon as she/he noticed the change of the image quality caused by the change of the mask.

The experiment was performed on a group of 6 volunteer observers (age between 20 and 24 years, 4 males and 2 females). They declared normal or corrected to normal vision and correct colour vision. The participants were aware that the visualisation quality is tested, but they were naïve about the purpose of the experiment.

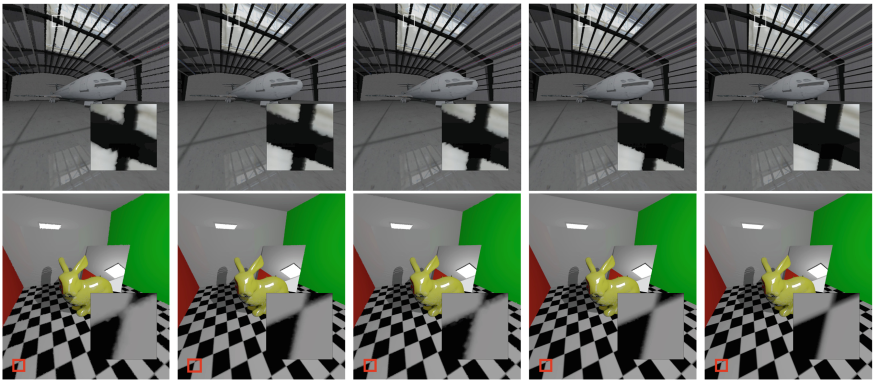

Fig. 7. Example images rendered for varying sampling distribution (from left: 18%, 27%, 41%, 54%, and 100% of the samples). The insets show magnification of the image regions depicted by the red rectangle. (Color figure online)

Apparatus and Performance Results. We used HTC Vive VR headset connected to PC computer with NVIDIA Geforce GTX 1080 GPU. This setup allows rendering two frames of 1512 × 1680 pixels resolution required by HTC Vive display in 66 ms for Air Shed scene, and 49.5 ms for Bunny Box.

Table 1 shows the average frame rendering times for each sampling mask. In the last column, the achieved increase in performance in comparison to the reference sampling is specified.

Table 1. Rendering performance.

Scene	Sampling mask	Rendering time [ms]	Speed-up
Air Shed	18%	18.1	3.7x
	27%	23.7	2.8x
	41%	37.5	1.8x
	54%	29.8	2.2x
Bunny Box	18%	15.0	3.3x
	27%	17.9	2.8x
	41%	27.6	1.8x
	54%	24.1	2.1x

We achieved more than 4-times speed-up for 18% sampling, however, this reduction of the samples is noticeable to observers (see Sect. 4). Acceptable quality was obtained for 41% and 54% sampling masks with double rendering time reduction.

Results. Figure 8 presents plots of the normalised detection rate for each sampling mask. The detection rate equal to one means that every observer managed to detect the change of the mask in all cases (i.e. pressed the mouse button at the right moment). The detection rate of zero means that observers noticed the change in 50% of cases, while −1 means that the change of the mask was unnoticed.

As can be seen in Fig. 8 (left), 18% and 27% sampling are above the zero detection threshold for both scenes, while 41% and 54% are below this threshold. It is worth noting that the experiment was performed for conservative assumptions because it is much simpler to notice image deterioration while the mask is replaced in real-time. It will be much harder to see image deteriorations if the same mask is used for the whole animation.

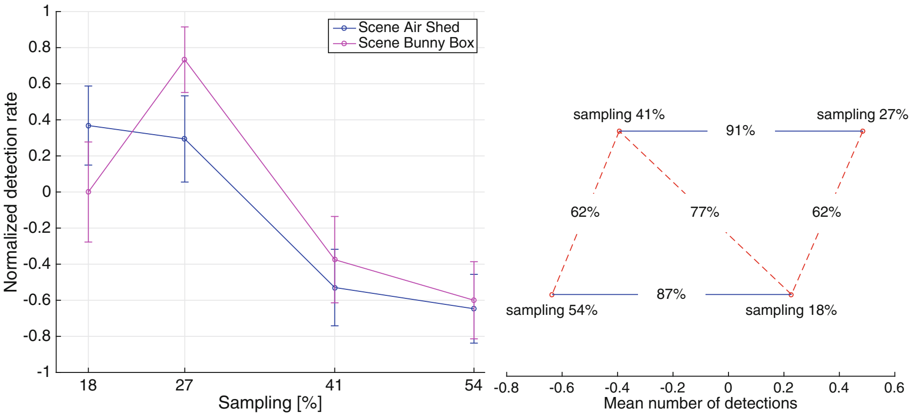

Fig. 8. Left: Normalized detection rate, the error bars depict the standard error of mean. Right: Ranking graph illustrating the statistical significance of the results achieved in the experiment. (Color figure online)

The tested the statistical significance of the achieved results using the multiple-comparison test, which identifies the statistical difference in ranking tests [12]. Figure 8 (right) presents a ranking of the mean detection rates for four tested sampling masks. They are ordered according to the detection rate, with the lowest detection on the left. The percentages indicate the probability that an average observer will choose the sampling on the right as worse than the sampling on the left. If the line connecting two samplings is red and dashed, it indicates that there is no statistical difference between this pair of samplings. The probabilities close to 50% usually result in the lack of statistical significance. For higher probabilities, the dashed-lines will start to be replaced by the blue lines. Figure 8 (right) shows that the ranking between 54% and 41% samplings cannot be trusted. However, 54% sampling generates significantly better results (fewer detections) than 18% sampling, and 41% sampling is better than 27% sampling.

5 Conclusions and Future Work

We have presented an efficient ray tracing system that renders the complex scenes on VR headset in real-time. The performance improvement was achieved with the use of non-uniform sample distribution, which by reducing the number of traced rays significantly decrease rendering time. Experiments that have been performed show that the reduction of the spatial sampling for the peripheral region of the image is barely noticeable, especially for the sampling mask, which mimics human sensitivity to contrast.

We did not manage to completely eliminate flickering in the periphery of vision. This flickering is caused by the temporal aliasing strengthened by the non-uniform and sparse sampling. In future work we plan to implement better anti-aliasing technique, e.g. the multi-sampling presented in [16]. We assume that the flickering can be further reduced by advanced fixation techniques that analyze the gaze data captured by eye tracker in a way more suitable for the foveated rendering. The essential work on this topic was done by Mantiuk et al. [11]. Recently, interesting findings were published by Roth et al. [18] and Weier et al. [25].

Acknowledgments. The project was funded by the Polish National Science Centre (decision number DEC-2013/09/B/ST6/02270).

References

1. Advanced Micro Devices, Inc.: Radeon-rays library, version 2.0 (2016). http://gpuopen.com/gaming-product/radeon-rays/
2. Akenine-Möller, T., Haines, E., Hoffman, N.: Real-Time Rendering, 3rd edn. A. K. Peters Ltd., Natick (2008)
3. Chwesiuk, M., Mantiuk, R.: Measurements of contrast detection thresholds for peripheral vision using non-flashing stimuli. In: Czarnowski, I., Howlett, R.J., Jain, L.C. (eds.) IDT 2017. SIST, vol. 73, pp. 258–267. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-59424-8_24
4. V Corporation: Openvr library, version 1.0.10 (2017). https://github.com/ValveSoftware/openvr
5. Fujita, M., Harada, T.: Foveated real-time ray tracing for virtual reality headset. Technical report, Light Transport Entertainment Research (2014)
6. Gortler, S.J., Grzeszczuk, R., Szeliski, R., Cohen, M.F.: The lumigraph. In: Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques, pp. 43–54. ACM (1996)
7. Guenter, B., Finch, M., Drucker, S., Tan, D., Snyder, J.: Foveated 3d graphics. ACM Trans. Graph. **31**(6), 164:1–164:10 (2012)
8. Hunt, W.: Virtual reality: the next great graphics revolution. Keynote Talk HPG (2015)
9. Levoy, M., Whitaker, R.: Gaze-directed volume rendering. ACM SIGGRAPH Comput. Graph. **24**(2), 217–223 (1990)
10. Loschky, L., McConkie, G., Yang, J., Miller, M.: The limits of visual resolution in natural scene viewing. Vis. Cogn. **12**(6), 1057–1092 (2005)

11. Mantiuk, R., Bazyluk, B., Mantiuk, R.K.: Gaze-driven object tracking for real time rendering. Comput. Graph. Forum **32**(2), 163–173 (2013)
12. Mantiuk, R.K., Tomaszewska, A., Mantiuk, R.: Comparison of four subjective methods for image quality assessment. Comput. Graph. Forum **31**(8), 2478–2491 (2012)
13. Murphy, H.A., Duchowski, A.T., Tyrrell, R.A.: Hybrid image/model-based gaze-contingent rendering. ACM Trans. Appl. Percept. (TAP) **5**(4), 22 (2009)
14. Ohshima, T., Yamamoto, H., Tamura, H.: Gaze-directed adaptive rendering for interacting with virtual space. In: 1996 Proceedings of the IEEE Conference on Virtual Reality Annual International Symposium, pp. 103–110. IEEE (1996)
15. Palmer, S.E.: Vision Science: Photons to Phenomenology, vol. 1. MIT Press, Cambridge (1999)
16. Patney, A., et al.: Perceptually-based foveated virtual reality. In: ACM SIGGRAPH 2016 Emerging Technologies, p. 17. ACM (2016)
17. Peli, E., Yang, J., Goldstein, R.B.: Image invariance with changes in size: the role of peripheral contrast thresholds. JOSA A **8**(11), 1762–1774 (1991)
18. Roth, T., Weier, M., Hinkenjann, A., Li, Y., Slusallek, P.: An analysis of eye-tracking data in foveated ray tracing. In: IEEE Second Workshop on Eye Tracking and Visualization (ETVIS), pp. 69–73. IEEE (2016)
19. Siekawa, A.: Gaze-dependent ray tracing. In: Proceedings of CESCG 2014: The 18th Central European Seminar on Computer Graphics (Non-peer-reviewed) (2014)
20. Siekawa, A.: Image reconstruction from spatially non-uniform samples. In: Proceedings of CESCG 2017: The 21th Central European Seminar on Computer Graphics (Non-peer-reviewed) (2017)
21. Stengel, M., Magnor, M.: Gaze-contingent computational displays: boosting perceptual fidelity. IEEE Sig. Process. Mag. **33**(5), 139–148 (2016)
22. Vaidyanathan, K., et al.: Coarse pixel shading. In: Proceedings of High Performance Graphics, pp. 9–18. Eurographics Association (2014)
23. Wandell, B.A.: Foundations of Vision, vol. 8. Sinauer Associates, Sunderland (1995)
24. Watson, B., Walker, N., Hodges, L.F., Worden, A.: Managing level of detail through peripheral degradation: effects on search performance with a head-mounted display. ACM Trans. Comput.-Hum. Interact. (TOCHI) **4**(4), 323–346 (1997)
25. Weier, M., et al.: Perception-driven accelerated rendering. Comput. Graph. Forum **36**(2), 611–643 (2017)