Foveated RTX Ray Tracing in Virtual Reality

Uroš Šmajdek[*]
Supervised by: Ciril Bohak[†]

Faculty of Computer and Information Science
University of Ljubljana
Ljubljana / Slovenia

Abstract

In this paper, we present a foveated ray tracing method for enhancing the viewing experience of virtual reality environments featuring dense molecular structures. Achieving immersion in virtual reality requires both high-resolution and high frame-rate rendering combined with the ability to generate realistic images. Our approach combines the power of the Nvidia RTX framework with a dynamic rendering rate that is based on the direction of the user's gaze. Foveated rendering is applied in the ray-generation stage of the ray-tracing pipeline, in order to reduce the ray-tracing rate depending on the distance from the center of the user's gaze on the image. This takes advantage of the way the human eye perceives visual detail, allowing us to spare processing power on areas that are largely imperceptible. The nature of the ray-tracing pipeline also allows us to easily apply any post-processing effects, such as screen-space ambient occlusion and temporal anti-aliasing, in a matter similar to the standard deferred rendering pipeline. We demonstrate our technique by rendering a model of SARS-CoV-2 on Meta Quest Pro in $3776 \times 3904 \times 2$ stereo resolution, where we observed a higher performance in comparison to the non-foveated counterpart of our method.

Keywords: Dense environments, ray tracing, foveated rendering, virtual reality

1 Introduction

Virtual reality (VR) is one of the key factors driving innovation in modern computer graphics. Development of immersive VR technologies has begun to encompass various fields from medicine [5, 10] to entertainment [30] and education [29], and as such, there is an increasing need for interactivity and real-time rendering. Additionally, over the last two decades, the pixel densities of high-quality head-mounted displays, also known as VR headsets, have dramatically increased from $263 \times 480 \times 2$ in 1997 (Forte VFX 3D) to $2448 \times 2448 \times 2$ in 2021 (HTC Vive Pro 2). To achieve the visual quality of a human eye, however, a display would require a resolution of about $32k \times 24k$ [12], making such achievement far beyond the reach of the current hardware and software, but nonetheless ensuring it will continue for decades to come. Not only that but to achieve immersion and limit the potential motion sickness, such rendering would have to exceed 60 updates per second.

One of the recent applications for such rendering is virtual visualization of the molecular structures of biological specimens [21, 1], which aims to immerse the user in the molecular environment, using highly detailed scenes coupled with interactivity. Such software can then be used for science dissemination to the broader public, an increasingly relevant topic since the Coronavirus pandemic. It can also aid in scientific presentations, or be used as a tool to gather additional information from such structures. On the other hand, dense molecular environments provide additional challenges, as scenes contain hundreds of thousands and up to billions of objects, that form closely packed structures, representing proteins, RNA, membranes, etc.

Foveated rendering presents one of the approaches to solving this problem. It describes the dynamic adaptation of rendering based on the user's gaze, specifically by limiting the details of the scene in the peripheral visual area that are largely imperceptible. The concept of foveated displays is not new, and gaze-reactive displays were already used in building flight simulators back in 2001 [25]. Until the recent generation of VR headsets, real-time eye tracking, which is required for its interactive use, was not commercially available, thus the approach was of lesser interest to a wider computer graphics community.

Hardware-accelerated RTX Ray Tracing is a recent breakthrough that promises to revolutionize the rendering scene, traditionally dominated by rasterization techniques. With the help of other Nvidia technologies, such as DLSS, it enables higher visual fidelity without compromising performance. Past research also indicates that the computational complexity of interactive ray tracing increases logarithmically with scene complexity [33], making it a compelling choice for addressing the above-mentioned interactive molecular visualization.

The main contributions of our work are:

- combining hardware-accelerate RTX Ray Tracing with Multi-spatial resolution-based foveated render-

[*]umajdek@gmail.com
[†]ciril.bohak@kaust.edu.sa

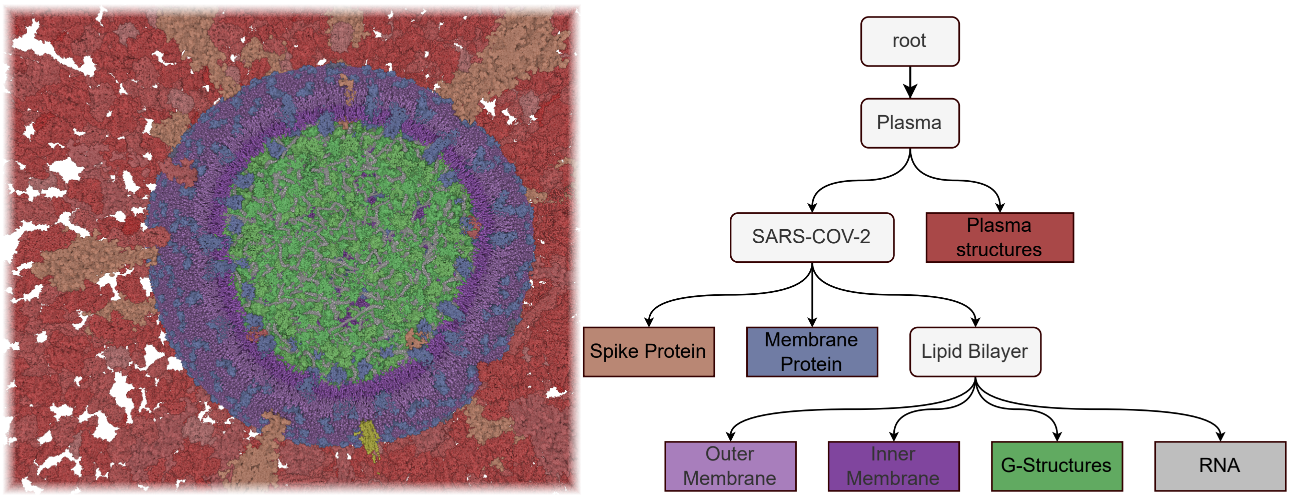

Figure 1: A visualization of dense molecular environment depicting SARS-COV-2 using our rendering technique (left), with individual structure labels and hierarchy (right).

ing,

- developing a pipeline that uses the technique for real-time visualization of dense molecular environments in virtual reality, and
- comparison and evaluation with the non-foveated ray tracing.

In section 2, we present the previous work on molecular visualization and resolution-based foveated ray tracing and differentiate our contributions from it. In section 3, we give a brief overview of the molecular environment we use as a basis. In section 4, we present our approach. The results and evaluation of our method are presented in sections 5 and 6. In section 7, we present the conclusions and give possible extensions as part of further work.

2 Related Work

Molecular Visualization: There are various tools for smaller-scale molecular visualization, such as VMD [11], Mol* [32], or PyMOL [31] which are unsuited for larger data sizes exceeding tens of millions of individual atoms. Over the years, numerous solutions to interactive molecular visualization have been proposed; glyph-based rendering [9], mapping structures onto a mesh geometry [36], using meshlets with probabilistic occlusion culling [13], instancing to repeat the recurring proteins in the scene [22, 6]. In 2015 Le Muzic et al. introduced cellVIEW [26], which depends on the LoD scheme and dynamic sphere primitives injection into primitives to be able to render 250 copies of the HIV virus model in blood plasma (16 billion atoms) at 60 FPS on NVIDIA GTX Titan. All of the above-mentioned techniques use procedural impostors to simplify the geometry and accelerate the rendering. Recently Alharbi et. al. [2] devised a solution that relies on hardware ray tracing instead and is able to render trillions of atoms using on-the-fly data construction and parallel rendering. In this paper, we expand on this paradigm but move the focus from scalability to higher performance, suitable for interactive VR visualization, which requires stereo rendering with a high and steady framerate. To this end, we supplement the hardware ray tracing with DLSS-enhanced foveated rendering but limit ourselves to data scales that directly fit into the GPU memory. To the best of our knowledge, there are no alternative foveated rendering techniques of dense molecular environments.

Multi-spatial resolution-based foveated ray tracing: Koskela et al. [20] proposed a theoretical estimate in 2016 that by integrating foveated rendering with ray tracing, 94% of the rays could potentially be omitted. This is based on the fact that ray tracing inherently supports spatial multi-resolution rendering, as it has the capacity to effortlessly adjust the number of rays emitted from a single pixel. Fujita and Harada [8] first implemented the foveated rendering system based on ray tracing, which relied on sparse sampling and kNN for image reconstruction. The system did not consider the eye sensitivity to contrast and lacked pertinent input from relevant user studies. To address these challenges, Weier et al. [38] combined ray tracing-based foveated rendering with reprojection rendering, using information from the previous frame to reduce the sampling rays for new frames. From there on out, different sampling models were proposed; Molenaar [24] traced rays based on the visual acuity fall-off model, Kim et al. [17] proposed a perceptually efficient pixel sampling method suitable for HMD ray tracing, which combined the Jin et al. [16] selective oversampling technique with the foveated rendering scheme and Koskela et al. [19] traced rays and denoised in Visual-Polar space and subsequently mapped the results to the screen space. Hybrid approaches also emerged with Blackmon et al. [4] using ray tracing to render the foveal region and rasterization to render the peripheral region. To address the lack of consideration for the displayed content, as opposed to the eccentricity, when

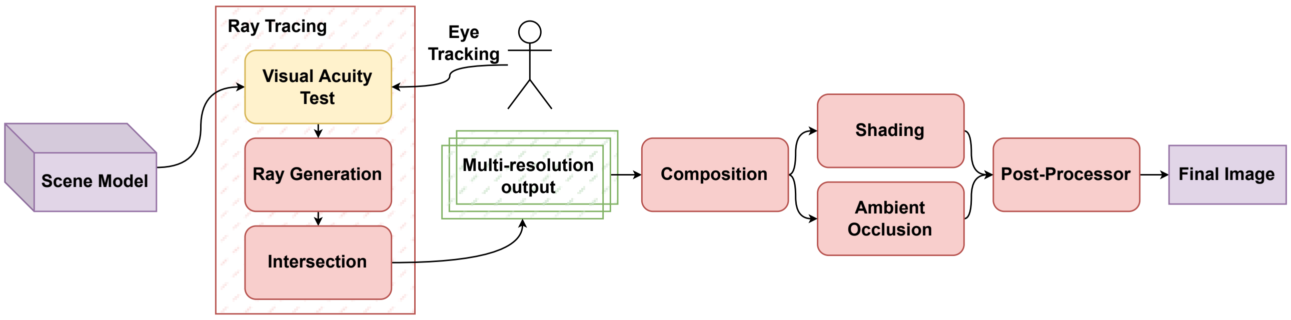

Figure 2: An overview of our technique for the foveated rendering of dense molecular environments in VR.

determining the number of rays emitted, Tursun et al. [35] proposed a new luminance-contrast-aware foveated ray tracing technique. The technique significantly reduced the number of traced rays but required a low-quality image to be generated for each frame, indicating areas with different luminances. Similarly, Yang et al. [39] introduced a trio of simple policies to achieve a variable ray tracing rate. The method integrates the foveate perspective, material, and depth information in order to dynamically reduce the number of tracing rays in certain steps in the ray tracing pipeline. Recently Wang et al. [37] compiled an in-depth overview of state-of-the-art foveated methods for rasterization, ray tracing, and volumetric rendering techniques. In our approach, we implement a low-overhead spatial policy to determine the local number of rays, similar to the one introduced in [39], but we further supplement it using a hardware-accelerated Nvidia Ray Tracing framework [18]. Additionally, we use traditional spatial and temporal anti-aliasing techniques in favor of Nvidia DLSS [23]. While there is no substantial contribution from individual methods, their combination, however, is a novelty. Moreover, some components of the pipeline utilize methods that we were unable to find elsewhere, including a single multi-resolution ray tracing pass and blending between foveal regions.

3 Molecular Environment

As input, we used the atom-based hierarchical 3D models of biological mesoscale organisms, such as viruses. The model may consist of an arbitrary number of instances, each composed of densely packed atoms, with the rest of the space implicitly filled with water molecules. We choose not to render those molecules as they are of less interest to the viewer and would render the scene completely packed. Those spaces are instead treated as empty, allowing the user to focus and travel between important structures found inside a virus. An example of a model's hierarchy can be seen in Figure 1. Additionally, employing such a hierarchical structure allows us to dynamically highlight different levels of structures based on distance and type, further enhancing the user experience.

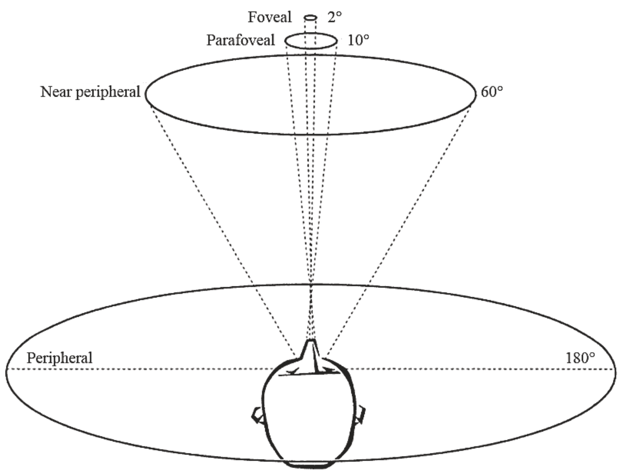

Figure 3: Schematic depiction of foveal/peripheral vision eccentric angles and regions by Ivanvcic et al. [14].

4 Foveated Ray Tracing

In this section, we describe our proposed foveated rendering system. An overview of the rendering pipeline can be seen in Figure 2, and we describe the individual stages in the following subsections. The pipeline is modular, the only static part being the visual acuity test and the hardware-accelerated ray tracing. Other parts can be easily swapped out or new ones added, which is made even easier by its strong resemblance to a classic deferred rendering pipeline.

4.1 Visual Acuity Test and Ray Generation

Visual acuity generally refers to our ability to distinguish the shapes and details of the object we are observing. One of its key properties is the foveal/peripheral vision, which recognizes that human visual acuity is not uniform over the entire visual field. Objects in the periphery are significantly harder to recognize than those in the center of the fovea [34], as illustrated in Figure 3.

We use this foveal/peripheral property of the visual acuity to determine the resolution at which we render different parts of the final image. To achieve this, we perform a vi-

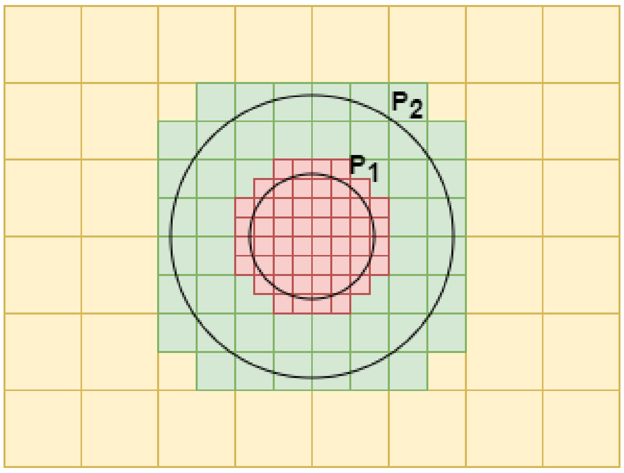

Figure 4: Visual representation of the multi-resolution rendering based on the eccentric angle from the user's gaze. The red squares in the center each cover one pixel, and larger squares denote rendering in lower resolution.

sual acuity test before the ray generation step, where we compare the eccentric angle of the final pixel relative to the center of the user's eye gaze. If we label the rendering resolution at pixel $\mathbf{x}$ as $R_\mathbf{x}$, we can parameterize the visual acuity test as:

$$R_\mathbf{x} = \begin{cases} 1 & : d_\mathbf{x} \leq P_1 \\ 1/4 & : P_1 < d_\mathbf{x} \leq P_2 \\ 1/16 & : d_\mathbf{x} > P_2 \end{cases}$$

where $d_\mathbf{x}$ denotes the eccentric angle at pixel $\mathbf{x}$, P_1 is set between foveal and parafoveal eccentric angle (2-10°), and P_2 is set between parafoveal and near peripheral eccentric angle (10-60°). The chosen resolutions conform to the standard mipmap sizes, normally used for texture rendering, which allows us to store and access the data more efficiently. A visual representation of the final image composition from different resolutions can be seen in Figure 4. Rendering resolution directly determines the number of rays generated in the ray-generation step of the ray-tracing pipeline.

4.2 Ray Tracing Pipeline

Our ray tracing pipeline heavily relies upon the Nvidia RTX ray tracing extension for Vulkan [18], which is the continuation of Nvidia OptiX [28], a general-purpose SDK for accelerating ray casting applications. Using the framework, we pack the atoms that build our model into *Accelerated Structures*, a specialized high-performance spatial structure built for ray tracing. The framework then takes care of optimizing the ray collision detection using bounding volume hierarchy traversal while we still retain full control of the ray generation and after-collision events. To be able to use deferred rendering techniques later, we store not only the colors of hit objects but also the hit location, the normal of the hit surface, and the hit object's numerical identifier.

To further reduce the overhead in the rendering pipeline, we only perform the ray tracing procedure once instead of separately for each resolution we render. We achieve this by inherently rendering at full resolution and then discarding the unneeded rays in the periphery.

4.3 Composition

During composition, we combine the 2D multi-resolution output of the ray tracing procedure. Since rendering at different resolutions results in spatial discontinuities between the rendering regions, we blend different resolutions at their border, as opposed to simply taking the highest resolution present at that location, which results in smoother transitions. To that end, we employ an alpha-blending technique to reduce the computational overhead of the procedure.

4.4 Shading

As our model does not depict an environment in which realistic lighting plays a role, we can simplify the illumination in ray tracing algorithm to focus solely on local illumination with Phong shading. This optimization not only improves performance by reducing the number of traced rays but also eliminates the probabilistic computations typically found in ray tracing algorithms, resulting in immediate convergence of the final result.

Another challenge that the molecular environment presents is the discrepancy between the macro-level and micro-level perspectives. At the macro level, we are interested in the various structures that make up our model, such as membranes or RNA, while at the micro level, we focus on individual atoms that comprise those structures. To seamlessly bridge these two levels, we modulate the color of each atom by interpolating between its inherent color, c_a, and the color of its corresponding structure, c_s, based on the atom's distance from the camera, d:

$$c = \begin{cases} (1 - \frac{d}{D_{max}}) \cdot c_a + \frac{d}{D_{max}} \cdot c_s & : d < D_{max} \\ c_s & : d \geq D_{max} \end{cases} .$$

Any atom located beyond a pre-determined distance, D_{max}, is assigned the color of its structure, making it easier to differentiate between the structures from a distance. The colors are user-defined.

4.5 Screen-Space Ambient Occlusion

As we are dealing with primarily mono-colored, locally illuminated base shapes of the same size, an ambient occlusion technique is needed to help the user distinguish between them and also improve depth perception. Screen Space Ambient Occlusion (SSAO) is well suited for this task, as its complexity does not scale with the number of objects in the scene and is generally better in dense environments due to the improved locality of GPU memory accesses. Since our model only contains spheres, we use the hemisphere sampling variant of the technique, first described by Fillion and McNaughton [7].

4.6 Post processing

In the post-processing stage, we draw atom contours in order to ease the distinction between neighboring objects. This is especially important when working in dense molecular environments, as the high number of small objects is inherently difficult to distinguish. To determine whether to draw a contour at a pixel or not, we use previously computed object identifiers to check if all neighboring pixels belong to the same object. If they do not we draw a semi-transparent contour and blend it with the pixel's color.

5 Results

We evaluated the proposed technique on the SARS-CoV-2 model [27], which consists of 74,724,996 atoms. Experiments were rendered to a Desktop application window with a rendering resolution of 3840×2160, and the Oculus Pro VR headset with a stereo rendering resolution of $3776 \times 3904 \times 2$. All rendering was done using a Nvidia GeForce RTX 4090 graphics card with 24 GB of VRAM, an AMD Ryzen Threadripper PRO 3995WX 64-core processor, and 256 GB of RAM.

To discern the impact of the inclusion of foveated rendering, we measured the number of rendered frames per second (FPS). We performed 4 different tests for different combinations of the inclusion of foveated rendering and SSAO. During each experiment, we measured the average framerate over a duration of 30 seconds. To better represent a realistic usage scenario we slowly moved the camera around for the entire duration. As the framerate is restricted to 72 FPS on Meta Quest Pro, we chose the scene that closely matches this framerate to minimize the impact on final results. The results can be seen in Table 1. The addition of foveated rendering almost doubled the performance both on Desktop and in VR as long as SSAO is not used. Enabling the SSAO resulted in a massive performance drop, especially when foveated rendering was also enabled. For qualitative evaluation, we present renderings of all four states in Figure 5. The two left segments show the rendering without foveation and the two right segments with foveation. The top two segments show the result with SSAO disabled, and the bottom two segments show the result with SSAO enabled.

Display	SSAO	Foveated	FPS	rFPS
Desktop	No	No	288.3	-
		Yes	553.0	92%
	Yes	No	126.6	-
		Yes	159.0	26%
Oculus Pro	No	No	39.9	-
		Yes	71.8	80%
	Yes	No	24.6	-
		Yes	36.1	47%

Table 1: Performance evaluation of the proposed foveated rendering pipeline for dense molecular data. For reference, we rendered the same model without foveated rendering and also repeated both experiments without the inclusion of SSAO. The maximum refresh rate on Oculus Pro is 72 Hz, which represents the maximum measured framerate on the system. rFPS is defined as relative improvement using foveation on the specific display and SSAO setting.

6 Discussion

Our tests have revealed that foveated rendering by itself massively improves the performance (see Table 1). The results are less significant when the technique is used simultaneously with SSAO. We believe this is due to the lower resolution producing higher spatial discontinuities between neighboring pixels in low-resolution regions, resulting in a higher number of GPU cache misses. By itself, SSAO also massively reduces performance, as observed by the 38%-56% framerate drop when applied without foveation. This likely signifies that SSAO is not a suitable candidate for rendering dense molecular data in high resolutions.

From Figure 5, we conclude that the visual differences between the foveated and non-foveated results are rather minimal, especially close to the gaze direction. Internal user testing has shown that during movement, the aliasing in the distant object in the peripheral regions can be distracting. To address this issue, a performance-friendly solution would be to apply a blur filter to both peripheral regions in the post-processing stage. On the other hand, the figure also showcases the importance of ambient occlusion, without which it becomes difficult to distinguish between the objects, but we do not believe it justifies the performance loss in terms of user experience.

7 Conclusions

In this paper, we present a multi-spatial resolution-based foveated ray tracing method for the visualization of dense molecular environments in virtual reality. We evaluate the technique, both with and without the addition of ambient occlusion, showing a moderate to large increase in performance, depending on whether SSAO was used or not. We

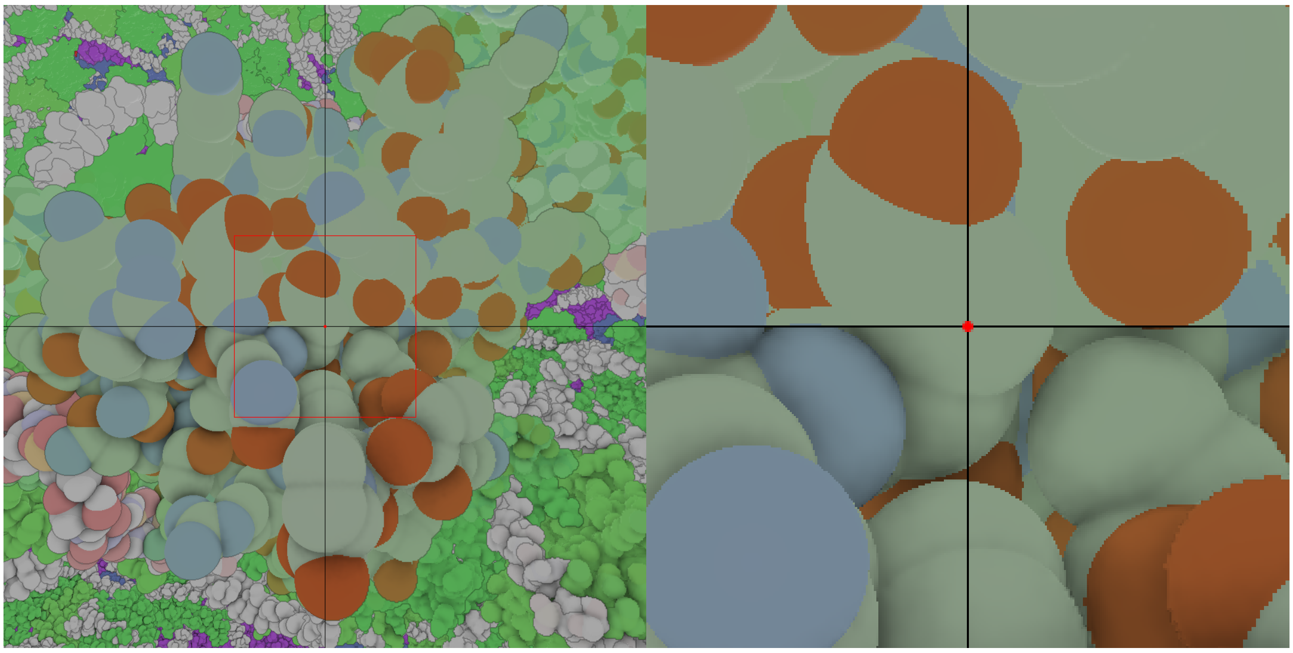

Figure 5: Individual image groups show a comparison between the non-foveated (left) and foveated (right) rendering and between no-SSAO (top) and with SSAO (bottom). The gaze direction is marked with a red dot. The right image group shows the magnification of the left one, denoted by the red square.

also showcase the need for ambient occlusion in terms of being able to tell the objects apart in a dense molecular environment.

As a future extension of this work more performant ambient occlusion techniques could be employed [3, 15], as the usage of SSAO resulted in a massive performance loss. Additionally, the method could be expanded by also considering the context of the scene during the visual acuity test, and thus being able to modulate the rendering resolution of different structures depending on other criteria, such as their depth and importance to the user.

References

[1] Ruwayda Alharbi, Ondrej Strnad, Laura Luidolt, Manuela Waldner, Ciril Bohak, David Kouril, Tobias Klein, Eduard Gröller, and Ivan Viola. Nanotilus: Generator of immersive guided-tours in crowded 3d environments. *IEEE Transactions on Visualization and Computer Graphics*, PP, 12 2021.

[2] Ruwayda Alharbi, Ondřej Strnad, Tobias Klein, and Ivan Viola. Nanomatrix: Scalable construction of crowded biological environments, 2022.

[3] Louis Bavoil. Deinterleaved Texturing for Cache-Efficient Interleaved Sampling. Technical report, NVIDIA Corporation, 2010.

[4] Steven Blackmon, Luke T. Peterson, Cuneyt Ozdas, and Steven J. Clohset. Foveated rendering, US Patent App. 15/372,589, 2017.

[5] Cléber Gimenez Corrêa, Fátima L. S. Nunes, Adriano Bezerra, and Paulo M. Carvalho. Evaluation of vr medical training applications under the focus of professionals of the health area. In *Proceedings of the 2009 ACM Symposium on Applied Computing*, SAC '09, page 821–825, New York, NY, USA, 2009. Association for Computing Machinery.

[6] Martin Falk, Michael Krone, and Thomas Ertl. Atomistic Visualization of Mesoscopic Whole-Cell Simulations Using Ray-Casted Instancing. *Computer Graphics Forum*, 32, 2013.

[7] Dominic Filion and Rob McNaughton. Effects & techniques. In *ACM SIGGRAPH 2008 Games*, SIGGRAPH '08, page 133–164. Association for Computing Machinery, New York, NY, USA, 2008.

[8] Masahiro Fujita and Takahiro Harada. Foveated real-time ray tracing for virtual reality headset. *Light Transport Entertainment Research*, 2014.

[9] Sebastian Grottel, Michael Krone, Christoph Müller, Guido Reina, and Thomas Ertl. MegaMol—A Prototyping Framework for Particle-Based Visualization. *IEEE Transactions on Visualization and Computer Graphics*, 21(2):201–214, 2015.

[10] Min-Chai Hsieh and Jia-Jin Lee. Preliminary study of vr and ar applications in medical and healthcare education. *Journal of Nursing and Health*, 3, 2018.

[11] William Humphrey, Andrew Dalke, and Klaus Schulten. VMD: Visual molecular dynamics. *Journal of Molecular Graphics*, 14(1):33–38, 1996.

[12] Warren Hunt. Virtual reality: The next great graphics revolution. *Keynote Talk HPG*, pages 1–2, 2015.

[13] Mohamed Ibrahim, Peter Rautek, Guido Reina, Marco Agus, and Markus Hadwiger. Probabilistic Occlusion Culling using Confidence Maps for High-Quality Rendering of Large Particle Data. *IEEE Transactions on Visualization and Computer Graphics (Proceedings IEEE VIS 2021)*, 28(1):573–582, 2022.

[14] Snježana Ivančić Valenko, Vladimir Cviljušac, Sanja Zlatić, and Damir Modrić. The impact of physical parameters on the perception of the moving elements in peripheral part of the screen. *Tehnički vjesnik*, 26(5):1444–1450, 2019.

[15] Jorge Jiménez, Xianchun Wu, Angelo Pesce, and Adrian Jarabo. Practical real-time strategies for accurate indirect occlusion. *SIGGRAPH 2016 Courses: Physically Based Shading in Theory and Practice*, 2016.

[16] Bongjun Jin, Insung Ihm, Byungjoon Chang, Chanmin Park, Wonjong Lee, and Seokyoon Jung. Selective and adaptive supersampling for real-time ray tracing. In *Proceedings of the Conference on High Performance Graphics 2009*, HPG '09, page 117–125, New York, NY, USA, 2009. Association for Computing Machinery.

[17] Youngwook Kim, Yunmin Ko, and Insung Ihm. Selective foveated ray tracing for head-mounted displays. In *2021 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, pages 413–421, 2021.

[18] Daniel Koch. Vulkan ray tracing final specification release. https://www.khronos.org/blog/vulkan-ray-tracing-final-specification-release, 11 2020. Accessed: 2023-02-11.

[19] Matias Koskela, Atro Lotvonen, Markku Mäkitalo, Petrus Kivi, Timo Viitanen, and Pekka Jääskeläinen. Foveated Real-Time Path Tracing in Visual-Polar Space. In Tamy Boubekeur and Pradeep Sen, editors, *Eurographics Symposium on Rendering - DL-only and Industry Track*. The Eurographics Association, 2019.

[20] Matias Koskela, Timo Viitanen, Pekka Jääskeläinen, and Jarmo Takala. Foveated path tracing. In George Bebis, Richard Boyle, Bahram Parvin, Darko Koracin, Fatih Porikli, Sandra Skaff, Alireza Entezari, Jianyuan Min, Daisuke Iwai, Amela Sadagic, Carlos Scheidegger, and Tobias Isenberg, editors, *Advances in Visual Computing*, pages 723–732, Cham, 2016. Springer International Publishing.

[21] David Kouril, Ondrej Strnad, Peter Mindek, Sarkis Halladjian, Tobias Isenberg, Eduard Gröller, and Ivan Viola. Molecumentary: Adaptable narrated documentaries using molecular visualization. *IEEE Transactions on Visualization and Computer Graphics*, PP:1–1, 11 2021.

[22] N. Lindow, D. Baum, and H.-C. Hege. Interactive Rendering of Materials and Biological Structures on Atomic and Nanoscopic Scale. *Comput. Graph. Forum*, 31(3pt4):1325–1334, 06 2012.

[23] Edward Liu. Dlss 2.0 - image reconstruction for real-time rendering with deep learning. https://developer.nvidia.com/gtc/2020/video/s22698-vid, 2020. Accessed: 2023-02-11.

[24] Erik N Molenaar. Towards real-time ray tracing through foveated rendering. Master's thesis, University of Utrecht, 2018.

[25] Hunter A. Murphy and Andrew T. Duchowski. Gaze-contingent level of detail rendering. In *Eurographics*, 2001.

[26] Mathieu Le Muzic, Ludovic Autin, Július Parulek, and Ivan Viola. cellVIEW: a Tool for Illustrative and Multi-Scale Rendering of Large Biomolecular Datasets. *Eurographics Workshop on Visual Computing for Biomedicine*, 2015:61–70, 2015.

[27] Ngan Nguyen, Ondřej Strnad, Tobias Klein, Deng Luo, Ruwayda Alharbi, Peter Wonka, Martina Maritan, Peter Mindek, Ludovic Autin, David S. Goodsell, and Ivan Viola. Modeling in the time of covid-19: Statistical and rule-based mesoscale models. *IEEE Transactions on Visualization and Computer Graphics*, 27(2):722–732, 2021.

[28] Steven G. Parker, James Bigler, Andreas Dietrich, Heiko Friedrich, Jared Hoberock, David Luebke, David McAllister, Morgan McGuire, Keith Morley, Austin Robison, and Martin Stich. Optix: A general purpose ray tracing engine. *ACM Trans. Graph.*, 29(4), 06 2010.

[29] Maria Paola Puggioni, Emanuele Frontoni, Marina Paolanti, Roberto Pierdicca, Eva Savina Malinverni, and Michele Sasso. A content creation tool for ar/vr applications in education: The scoolar framework. In *Augmented Reality, Virtual Reality, and Computer Graphics: 7th International Conference, AVR 2020, Lecce, Italy, September 7–10, 2020, Proceedings, Part II*, page 205–219, Berlin, Heidelberg, 2020. Springer-Verlag.

[30] Cedriss Saint-Louis and Abdelwahab Hamam. Survey of haptic technology and entertainment applications. In *SoutheastCon 2021*, pages 01–07. IEEE, 03 2021.

[31] Schrödinger, LLC. The PyMOL Molecular Graphics System, Version 1.8. unpublished, 11 2015.

[32] David Sehnal, Sebastian Bittrich, Mandar Deshpande, Radka Svobodová, Karel Berka, Václav Bazgier, Sameer Velankar, Stephen K Burley, Jaroslav Koča, and Alexander S Rose. Mol* Viewer: modern web app for 3D visualization and analysis of large biomolecular structures. *Nucleic Acids Research*, 49(W1):W431–W437, 05 2021.

[33] Philipp Slusallek and I Wald. State of the art in interactive ray tracing. *STAR, EUROGRAPHICS 2001*, pages 21–42, 2001.

[34] Hans Strasburger, Ingo Rentschler, and Martin Jüttner. Peripheral vision and pattern recognition: A review. *Journal of vision*, 11:13, 05 2011.

[35] Okan Tarhan Tursun, Elena Arabadzhiyska-Koleva, Marek Wernikowski, Radosław Mantiuk, Hans-Peter Seidel, Karol Myszkowski, and Piotr Didyk. Luminance-contrast-aware foveated rendering. *ACM Transactions on Graphics (TOG)*, 38(4):1–14, 2019.

[36] Thomas Waltemate, Björn Sommer, and Mario Botsch. Membrane Mapping: Combining Mesoscopic and Molecular Cell Visualization. In *Eurographics Workshop on Visual Computing for Biology and Medicine*, 2014.

[37] Lili Wang, Xuehuai Shi, and Yi Liu. Foveated rendering: A state-of-the-art survey. *Computational Visual Media*, 9(2):195–228, 2023.

[38] Martin Weier, Thorsten Roth, Ernst Kruijff, André Hinkenjann, Arsène Pérard-Gayot, Philipp Slusallek, and Yongmin Li. Foveated real-time ray tracing for head-mounted displays. *Computer Graphics Forum*, 35:289–298, 10 2016.

[39] Jinyuan Yang, Xiaoli Li, and Abraham G. Campbell. Variable rate ray tracing for virtual reality. In *SIGGRAPH Asia 2020 Posters*, SA '20, New York, NY, USA, 2020. Association for Computing Machinery.