Radiance probes for global illumination in volume rendering

Manca Drašček, Ciril Bohak, Matija Marolt, Žiga Lesar

University of Ljubljana, Faculty of Computer and Information Science
E-mail: md21577@student.uni-lj.si, {ciril.bohak, matija.marolt, ziga.lesar}@fri.uni-lj.si

Abstract

Global illumination is critical for producing visually realistic volume renderings but is often computationally expensive, particularly when using path tracing. To address this, we implement and evaluate a radiance probe-based approach that approximates indirect lighting by precomputing radiance at discrete points in a volume. Built upon an existing WebGPU-based path tracer, our system uses spherical harmonics (SH) to compactly encode directional lighting at each probe and enables efficient trilinear interpolation during rendering. The implementation offers interactive control over probe density, sampling parameters, and visualization options. We compare the radiance probe approach to full path tracing outputs both qualitatively and quantitatively. While it achieves comparable visual quality in many scenarios, performance gains were limited due to buffer management inefficiencies. Despite current technical constraints, our results demonstrate that radiance probes are a promising strategy for accelerating volume rendering in interactive applications, especially when high-frequency lighting precision is not critical.

1 Introduction

Accurate and efficient computation of global illumination is a significant challenge in volume rendering. The appearance of materials is heavily influenced by the surrounding environment, making the quality of rendering depend on effective environment sampling. While path tracing can produce accurate results, it is often too slow for real-time execution. A common practical solution is to precompute radiance at fixed points in space known as radiance probes or light probes and use these values during rendering to avoid costly computation for every light ray.

In this work, we explored the radiance probe approach to global illumination in volume rendering and implemented it in a prototype application, building upon an existing WebGPU-based path tracer framework and extending it with radiance-probe-based illumination, providing interactive control over probe density, sampling parameters, and visualization options. We evaluated the results both qualitatively and quantitatively and compared them to path tracing.

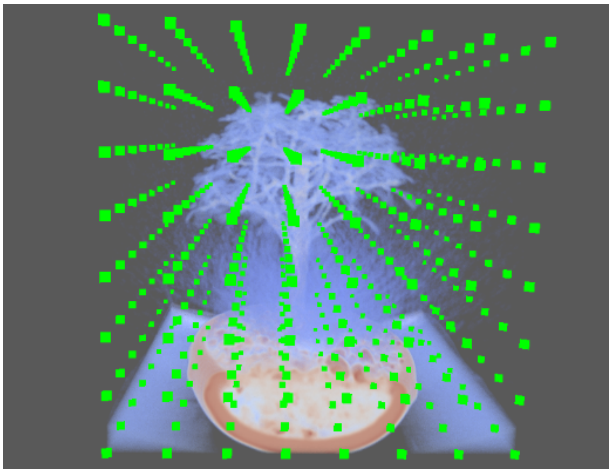

Figure 1: Radiance probe grid of size 8×8×8.

2 Related work

Traditional volume rendering with path tracing simulates the transport of light rays as they scatter and absorb within the volume [1]. Rays are cast from the camera through the volume, and at each step, absorption and scattering events are probabilistically sampled using techniques like Woodcock tracking or delta tracking [5, 2, 6] to model volumetric extinction and scattering.

Key steps in volume path tracing include calculating ray-volume intersections to determine the volume segment the ray traverses, sampling absorption and scattering events along the ray using stochastic methods, and estimating in-scattered radiance by recursively tracing light rays toward light sources at each scattering event. While this approach can produce physically accurate results, it requires tracing many samples per pixel to converge to a noise-free image, making it computationally too expensive for real-time applications. In-scattered radiance estimation is typically the most computationally intensive part of the algorithm, while being crucial for accurate representation of global illumination. There are several web-based implementations, such as [7].

Several approaches exist to accelerate volume rendering while maintaining reasonable visual quality. These include photon mapping [3], which is used to precompute partial light paths, irradiance caching [4] for storing

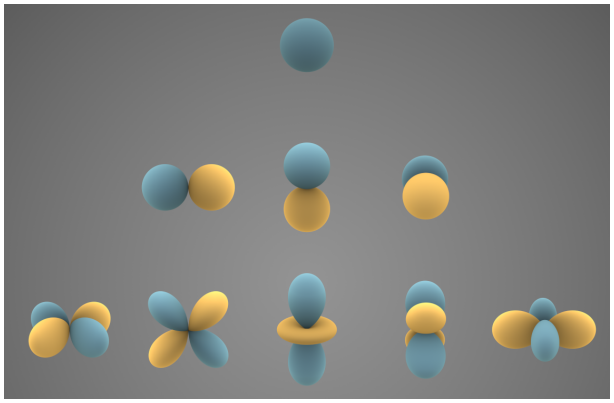

Figure 2: 2nd order spherical harmonics basis functions. Image by Inigo Quilez, CC BY-SA 3.0.

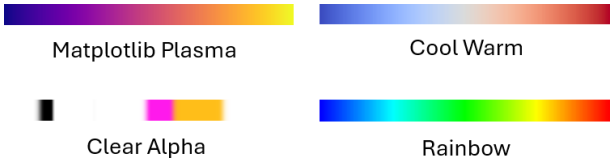

Figure 3: Transfer functions used for testing.

preintegrated in-scattered radiance at predefined points in space, and radiance probes [8], which store radiance at predefined points in space.

3 Methods

Our work focuses on radiance probes to accelerate indirect illumination estimation. We place radiance probes at fixed points on a regular grid superimposed on a volume, as shown in Figure 1. Radiance probes store directional lighting information that can be reused during rendering to avoid expensive recursive path tracing at every scattering event during light transport simulation. Instead of tracing many rays per pixel, the renderer interpolates lighting from nearby probes, enabling faster computation. Radiance probes thus approximate the spatial distribution of incident radiance, allowing efficient global illumination approximation without full path tracing for every frame.

Since radiance is a directional quantity, the challenge with radiance probes lies in compactly representing directional lighting information. A straightforward approach stores radiance for many discrete directions (like cube maps), but this is memory-intensive. SH provide an elegant solution by projecting the directional radiance onto a set of orthogonal basis functions defined on the sphere. SH coefficients succinctly encode smooth lighting distributions, capturing low-frequency lighting variations efficiently. For volume rendering, typically, low-order SH (e.g., 2nd order, 9 coefficients) are sufficient to represent diffuse illumination at each probe [9]. Using SH, each probe stores a small vector of coefficients per color channel, which can be quickly evaluated for any lighting direction by computing a weighted sum of the basis functions. This approach balances accuracy and storage and enables fast interpolation for lighting evaluation.

3.1 Radiance probes

Our implementation places radiance probes on a uniform 3D grid throughout the volume, with user-controllable resolution. During precomputation, we compute radiance for each probe by casting multiple light rays in random directions sampled uniformly over the unit sphere. Each light ray is traced into the volume using delta tracking to generate scattering events. At each scattering point, direct illumination from the light source is evaluated using shadow ray transmittance estimation. The algorithm is similar to path tracing but focused on estimating the incident radiance from all directions at the radiance probe position. We take multiple samples per probe to reduce noise in the radiance estimates.

The estimated radiance is then projected onto SH coefficients for compact storage and efficient evaluation during rendering. We use 2nd order SH representation, therefore the SH basis functions we use consist of 1 constant term, 3 linear terms, and 5 quadratic terms, shown in Figure 2. Each radiance probe stores 9 coefficients that correspond to the basis functions per color channel, totaling 27 floating-point values per radiance probe.

During rendering, when we need radiance estimates at locations between probes, we perform trilinear interpolation of the SH coefficients from the 8 nearest probes. This allows smooth transitions between probe points while maintaining directional lighting information.

To visualize single-channel data values such as density or radiance magnitude, we use one-dimensional transfer functions stored as one-dimensional textures that map single-valued volume samples to RGB colors. Transfer functions used for testing were based on popular color maps from the scientific visualization field to provide intuitive and perceptually meaningful visuals (see Figure 3).

3.2 Implementation details

Our implementation builds upon an existing WebGPU-based volume rendering framework by Will Usher.[1] We extended this framework with radiance probe functionality while maintaining the original path tracing capabilities for comparison. Our enhanced application supports choosing different transfer functions and volumetric models, adjusting radiance probe density and lighting setup (direction and intensity), controlling the number of samples for radiance probe precomputation, and toggling between the display of radiance probes or the final rendered volume. The extended application is available publicly on GitHub.[2]

The system works in two phases:

1. **Radiance probe precomputation.** Before rendering begins, we compute radiance at all probe positions using a compute shader. This also happens on every parameter change (e.g., volumetric model, transfer function, radiance probe density, lighting, etc.).

[1] https://github.com/Twinklebear/webgpu-volume-pathtracer

[2] https://github.com/mancadra/NRG-RadianceProbes

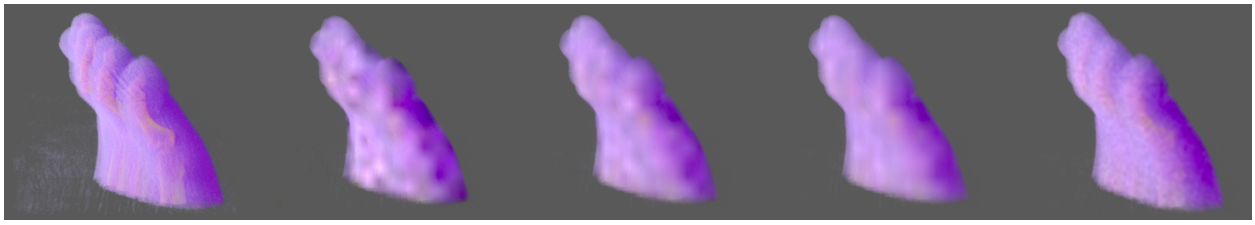

Figure 4: Rendering of a CT scan of a foot with path tracing and with radiance probes, using the Matplotlib Plasma transfer function. From left to right: path tracing, 16×16×16 probes with 16 samples, 16×16×16 probes with 128 samples, 16×16×16 probes with 512 samples, and 64×64×64 probes with 512 samples.

Table 1: Render time and initialization time measurements for path tracing and radiance probes.

Method	Render time [ms]			Initialization time [ms]		
	Fuel (64×64×64)	Hydrogen (128×128×128)	Foot (256×256×256)	Fuel (64×64×64)	Hydrogen (128×128×128)	Foot (256×256×256)
Path tracing	1.221	2.748	3.167	-	-	-
Radiance Probes						
Samples per probe: 16						
8×8×8	1.913	5.448	5.196	0.300	5.800	0.699
16×16×16	1.098	5.728	4.920	0.300	0.600	0.699
32×32×32	1.432	4.913	4.774	0.389	0.900	0.500
64×64×64	1.965	3.561	5.855	0.399	0.800	0.600
Samples per probe: 64						
8×8×8	1.322	6.567	3.327	0.300	0.400	2.565
16×16×16	1.574	5.477	1.244	0.400	0.400	0.300
32×32×32	1.751	7.896	4.371	0.400	0.399	4.000
64×64×64	1.098	9.350	6.777	0.299	0.500	3.899
Samples per probe: 128						
8×8×8	1.657	5.308	5.789	0.500	0.800	0.500
16×16×16	1.615	3.909	1.232	0.500	0.500	0.600
32×32×32	2.718	6.336	1.149	0.300	0.601	0.599
64×64×64	6.665	9.931	8.426	0.300	0.400	0.600
Samples per probe: 256						
8×8×8	4.771	4.800	6.398	0.600	0.699	0.600
16×16×16	6.623	2.072	1.917	1.099	0.800	0.500
32×32×32	3.301	4.709	6.057	0.599	0.800	0.799
64×64×64	4.913	10.608	5.415	3.100	3.899	5.000

2. **Rendering.** During rendering, we use the precomputed radiance probes to illuminate the volume. This eliminates the need for recursive light ray generation at every scattering location, as the radiance can be directly sampled from the radiance probe grid.

The probe precomputation uses a parallel compute shader that distributes work across the GPU. Each thread handles one radiance probe, sampling random directions and accumulating radiance into SH coefficients. After computation, the radiance probe data (SH coefficients) are copied to a read buffer for use during rendering.

During rendering, we evaluate the SH coefficients at each sample point to get directional lighting, blending between nearby radiance probes using trilinear interpolation.

4 Results

We evaluated our implementation both qualitatively and quantitatively against regular volumetric path tracing. Qualitatively, the radiance probe method produces visually similar results to path tracing, especially with sufficient probe density and samples, as shown in Figures 4 to 6. The main differences appear in areas of high-frequency lighting or sharp shadows, where the probe interpolation can cause blurring. Additionally, some noticeable artifacts are visible if the probe density is lower, where the probe placement is visible in the gradient of shadows. This is caused by linear interpolation of radiance where such a simplification is not suitable due to sharp changes in the radiance field. However, for a relatively smooth volume, such as the one shown in Figure 5, even a low-resolution radiance probe grid produces adequate results.

Table 1 lists the time measurements for rendering three test volumes with path tracer and radiance probes with different probe density and samples per probe configurations. All measurements were performed on a AMD Ryzen 5 7530U CPU with 16GB RAM and integrated AMD Radeon Graphics GPU. The render times for radiance probes remain approximately equal across test cases, while the time of the initialization of probes increases with density and sampling per probe. Initialization measurements are in most cases below 1 ms, where high measurement accuracy is not achievable due to browser limitations. However, noticeably elevated initialization times have been measured for radiance probe resolutions

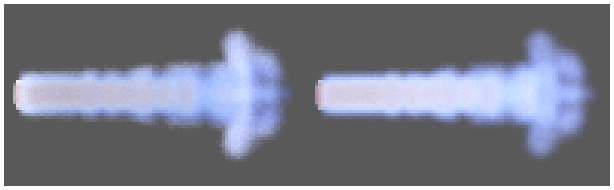

Figure 5: Rendering of a fuel injection simulation with path tracing (left) and radiance probes (right) of resolution 8×8×8 and 64 samples per probe. We used the Cool Warm transfer function.

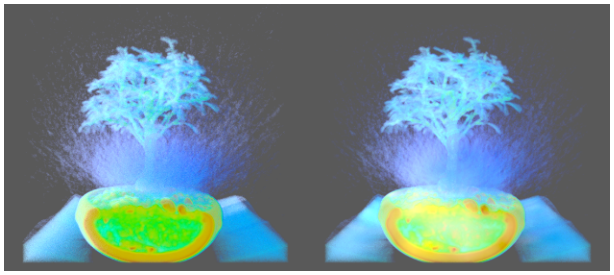

Figure 6: Rendering of a CT scan of a bonsai tree with path tracing (left) and radiance probes (right) of resolution 16×16×16 and 64 samples per probe. We used the Rainbow transfer function.

of 64×64×64 and above.

Rendering times for both path tracing only and path tracing with radiance probes stay well within real-time limits. However, rendering times for volumes using radiance probes have not improved as expected. We attribute this outcome to inefficient buffer storage and access, since we perform trilinear interpolation in software within a shader. A single radiance sample requires 72 memory accesses (9 coefficients and 8 neighbors). We expect using storage textures would accelerate sampling, but we are instead faced with prohibitive limitations on the API side – most current hardware allows only 4 or 8 storage texture bindings, with texture formats limited to common color formats, which is insufficient to store all SH coefficients.

Convergence, on the other hand, depends strongly on the volume itself and the transfer function. For a highly scattering medium, we expect path tracing with radiance probes to converge much faster to the correct result, since naive path tracing requires sampling many light paths for a result that can be obtained with a single sample from the radiance probes.

5 Conclusion

The radiance probe method provides an effective compromise between rendering quality and performance for volume visualization. Potential improvements for future work include adaptive probe placement based on volume features or higher-order SH for better directional lighting accuracy, and using storage textures to optimize radiance sampling. Overall, radiance probes offer a practical solution for interactive volume rendering with low-frequency global illumination effects, suitable for applications where real-time performance is more critical than absolute physical accuracy. Therefore, this approach is not suitable for medical usage where details are the key.

Acknowledgment

This research was conducted as part of the basic research project *Cell visualization of unified microscopic data and procedurally generated sub-cellular structures* [project number J2-50221], funded by the Slovenian Research and Innovation Agency (Javna agencija za znanstvenoraziskovalno in inovacijsko dejavnost RS) from the state budget.

References

[1] J. Fong, M. Wrenninge, C. Kulla, and R. Habel. Production volume rendering. In *ACM SIGGRAPH 2017 Courses*, SIGGRAPH '17, New York, NY, USA, 2017. Association for Computing Machinery.

[2] M. Galtier, S. Blanco, C. Caliot, C. Coustet, J. Dauchet, M. El Hafi, V. Eymet, R. Fournier, J. Gautrais, A. Khuong, B. Piaud, and G. Terrée. Integral formulation of null-collision monte carlo algorithms. *Journal of Quantitative Spectroscopy and Radiative Transfer*, 125:57–68, 8 2013.

[3] H. W. Jensen. *Realistic image synthesis using photon mapping*. AK Peters/crc Press, 2001.

[4] J. Křivánek and P. Gautron. *Practical global illumination with irradiance caching*, volume 10. Morgan & Claypool Publishers, 2009.

[5] T. Kroes, F. H. Post, and C. P. Botha. Exposure render: An interactive photo-realistic volume rendering framework. *PLoS ONE*, 7:e38586, 7 2012.

[6] P. Kutz, R. Habel, Y. K. Li, and J. Novák. Spectral and decomposition tracking for rendering heterogeneous volumes. *ACM Transactions on Graphics*, 36:1–16, 8 2017.

[7] Ž. Lesar, C. Bohak, and M. Marolt. Real-time interactive platform-agnostic volumetric path tracing in webgl 2.0. In *Proceedings of the 23rd International ACM Conference on 3D Web Technology*, Web3D '18, 2018.

[8] P. Stadlbauer, W. Tatzgern, J. H. Mueller, M. Winter, R. Stojanovic, A. Weinrauch, and M. Steinberger. Adaptive multi-view radiance caching for heterogeneous participating media. In *Computer Graphics Forum*, page e70051. Wiley Online Library, 2025.

[9] J. Stam. Multiple scattering as a diffusion process. In Patrick M. Hanrahan and Werner Purgathofer, editors, *Rendering Techniques '95*, pages 41–50, Vienna, 1995. Springer Vienna.