

Gaussian Light Transport

PATRICK ATTIMONT, INRIA - Grenoble University, France
KARTIC SUBR, University of Edinburgh, United Kingdom
CYRIL SOLER, INRIA - Grenoble University, France

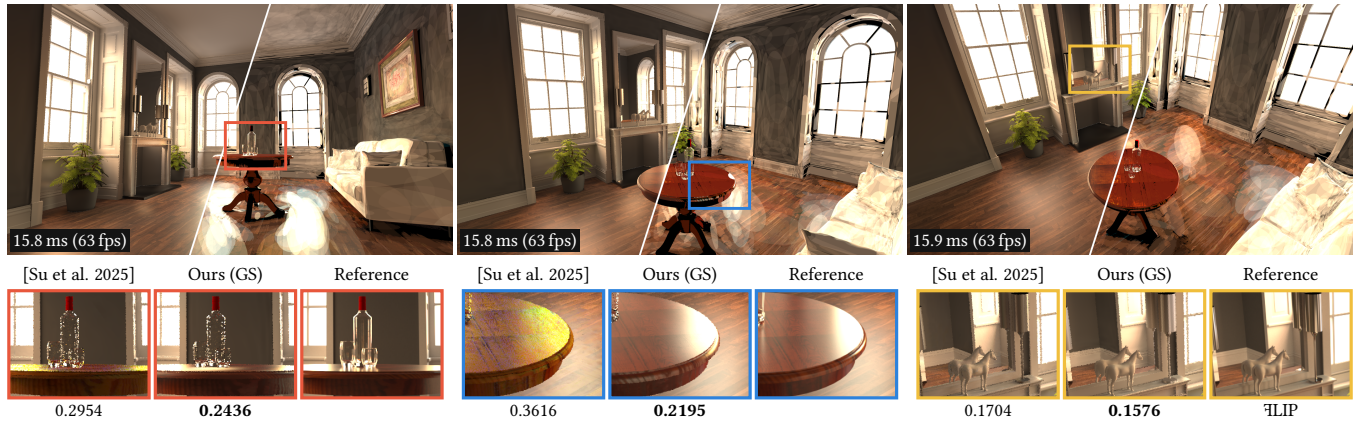


Fig. 1. Our method represents global illumination as an explicit mixture of 13D Gaussians optimized directly against the residual of the rendering equation. On LIVING ROOM, we train for 11m30s and render from arbitrary viewpoints at 1920×1080 , 1 spp, 63 fps. We achieve lower error and render $3.7\times$ faster than Vertex Features [Su et al. 2025] which we train for 4h29m. Each image above is split between the rendered Gaussians and their flattened visualization along the scene geometry.

We present a novel method for computing global illumination by expressing the solution to the light transport equation as a 13D Gaussian mixture model over positions, directions, surface normals, and material properties. We show that including scene properties in the Gaussian representation drastically reduces the number of functions and speeds up evaluation. As opposed to traditional light transport methods based on Neumann series, the parameters of our model are directly estimated by minimizing the residual of the rendering equation. While both optimization and rendering require repeated evaluations of a linear combination of high-dimensional Gaussian functions, we introduce an efficient culling strategy to keep the optimization tractable and produce renderings in real time. Our representation enables to render fast, view-independent solutions to the light transport equation, achieving rendering times on the order of milliseconds, with a fraction of the memory requirements of conventional neural rendering approaches.

CCS Concepts: • **Computing methodologies** → **Ray tracing**.

Additional Key Words and Phrases: Global illumination, Gaussian mixtures

ACM Reference Format:

Patrick Attimont, Kartic Subr, and Cyril Soler. 2026. Gaussian Light Transport. In *SIGGRAPH Asia 2026 Conference Papers (SA Conference Papers '26)*,

Authors' Contact Information: Patrick Attimont, INRIA - Grenoble University, Grenoble, France, patrick.attimont@inria.fr; Kartic Subr, University of Edinburgh, Edinburgh, United Kingdom, kartic@gmail.com; Cyril Soler, INRIA - Grenoble University, Grenoble, France, cyril.soler@inria.fr.



This work is licensed under a Creative Commons Attribution 4.0 International License. *SA Conference Papers '26, Kuala Lumpur, Malaysia*
© 2026 Copyright held by the owner/author(s).
ACM ISBN 979-8-4007-2842-6/26/12
<https://doi.org/10.1145/3829340.3842373>

December 01–04, 2026, Kuala Lumpur, Malaysia. ACM, New York, NY, USA, 11 pages. <https://doi.org/10.1145/3829340.3842373>

1 Introduction

Efficient computation of global illumination in 3D environments is a long-standing problem in computer graphics. Solving the integral equation [Kajiya 1986]—which models the equilibrium of directional radiant light energy—would yield a radiance field which can be used to render photo-realistic images from any viewpoint. This remains an elusive goal, due to the recursive nature of the equation which unravels as infinite-dimensional integrals. In this paper, we represent the solution to the equation as a sum of Gaussians and optimize the parameters of the Gaussians to approximately satisfy the equation.

Previous approaches have uncovered trade-offs between *View-Independent Rendering* by solving the global radiance field vis-a-vis view-dependent solutions. Galerkin methods (such as radiosity [Goral et al. 1984]) yield global solutions but are impractical for representing high-frequency effects due to reflections, caustics, shadows, etc. On the other hand, naive path tracing [Kajiya 1986] tends to require heavy computation for smooth effects due to global illumination. The *de facto* approach for contemporary light transport solutions is Monte Carlo integration of the Neumann series expansion of the rendering equation, yielding a general and effective solution. View-dependent methods require recomputation for novel views within a scene, but are usually justified when there are dynamic elements. We propose a method for view-independent rendering that is effective for multi-view rendering (e.g. fly-throughs) of complex, static scenes. In other words, “view-independent” means

that the calculation is not tied to the parameters of a camera (as path tracing would), and computes the entire light field that solves the transport equation. View-dependent images are thereafter produced by slicing this light field for a given camera.

Several methods from machine learning have been adapted for rendering [Keller et al. 2018]. Image-based rendering approaches such as Neural Radiance Fields (NeRFs) [Mildenhall et al. 2020] and Gaussian Splatting [Kerbl et al. 2023] aim to hypothesize arbitrary views given multi-view images as input. Neural radiosity optimizes a neural field, defined on a discrete grid, to minimize the residual of the rendering equation [Hadadan et al. 2021; Su et al. 2024].

We propose a new view-independent method for solving the light transport equation by leveraging the versatile and adaptive nature of Gaussian kernels. We represent the solution to the light transport equation as the sum of 13D Gaussians over the full spaces of positions, directions, normals, and material properties, and optimize these parameters to minimize the residual of the light transport equation. Our Gaussian kernels are therefore not restricted to remain on surfaces and inherently handle local variations in normals and materials. This allows us to compute a view-independent approximation to the solution to the rendering equation at a fraction of the memory cost of neural methods. We exploit the local support of Gaussians to design a culling strategy for a fast evaluation of the model, enabling efficient optimization (in a matter of minutes) and real-time rendering (see Figure 2). In summary,

- we propose a novel approach to solving the light transport equation by directly optimizing Gaussians, to obtain a view-independent solution;
- the complexity of our model adapts to the solution and is not constrained by the geometry of the scene;
- we show that this approach matches or improves on existing neural methods in quality, while training 10–23× faster and rendering 2–4× faster, with a memory footprint of just a few megabytes.

2 Related Work

2.1 Global illumination methods

Galerkin and Collocation Methods. Galerkin radiosity [Goral et al. 1984; Zatz 1993] and directional finite element methods [Heckbert and Winget 1991] project solutions onto structured basis functions defined over the scene geometry. They involve solving a linear system and provide view-independent solutions with theoretical guarantees. However, the need to parameterize geometry and directions makes them poorly suited to complex scenes with intricate geometry or highly glossy materials. Representing radiance as a 5D function leads to a prohibitive memory cost for fine basis functions that are required to capture phenomena such as caustics and glossy reflections. Collocation methods, including meshless formulations [Christensen 2008; Lehtinen et al. 2008; Wang et al. 2015], alleviate some discretization constraints but do not fundamentally resolve these scalability issues.

Monte Carlo and Path Tracing Methods. Monte Carlo methods avoid explicit function representations by estimating the solution

through stochastic sampling, typically via a Neumann series expansion of the rendering equation. Path tracing and its extensions—including bidirectional path tracing [Veach 1997], photon mapping [Hachisuka et al. 2008; Jensen 1996], and Metropolis Light Transport [Veach and Guibas 1997]—are highly effective at handling complex light transport phenomena. They are flexible, but can exhibit noise from high variance when long light paths are involved, making them computationally expensive and poorly suited to view-independent global illumination. Path re-using methods [Ouyang et al. 2021], temporal reprojection [Schied et al. 2017] and gradient-domain path tracing [Kettunen et al. 2015] mitigate this problem by making the best of already computed sub-parts of the light transport for new views in their own specific ways.

One other limitation of path tracing methods is their inherent ignorance of the spatial and angular bandwidth of the reconstructed signal, causing high variance in low frequency parts of the image. Frequency-aware splatting [Belcour et al. 2013] and adaptive low-frequency methods [Jarosz et al. 2011] partially address these issues.

2.2 View-independent precomputed radiance

PRT and Radiance caching. Precomputed Radiance Transfer (PRT) methods compute light transport matrices that allow scene elements such as view points and light [Ng et al. 2004; Sloan et al. 2023], scene geometry [Iwasaki et al. 2007] and materials [Sun et al. 2007] to be changed interactively when rendering novel views. More recently, Guo et al. [2024] combine precomputed radiance transport with 3D Gaussian splatting to interactively generate images with indirect illumination—as opposed to global illumination. PRT methods struggle to locally adapt to the radiance field, leading to heavy memory requirements. This is further exacerbated by the basis functions (wavelets, spherical harmonics, etc.) being tied to the geometry.

Radiance caching (and irradiance caching) [Křivánek et al. 2008] accelerates global illumination for non-Lambertian scenes by pre-computed storage of radiance in a world-space grid. This can be stored on surfaces [Tatzgern et al. 2024] or in a volumetric grid [Xing et al. 2024]. Novel views are rendered by simply convolving the BRDFs with incoming radiance from the cache. Storage requirements of these methods are high (gigabytes) but rendering can be an order of magnitude faster (10 ms) than PRT. These methods make use of their precomputed cache to retrieve indirect illumination only, while our evaluated radiance fields include both direct and indirect illumination as a proxy.

Alternative representations. Several approaches trade physical rigor for efficiency and compactness. Data-driven bases [Belcour et al. 2022] model the space of solutions using learned low-dimensional representations, enabling efficient relighting. Point-based and volumetric approaches such as point-based radiance fields [Zhang et al. 2023], Plenoxels [Yu et al. 2021], and Gaussian splatting [Kerbl et al. 2023] optimize for visual metrics. Neural radiance fields (NeRF) [Mildenhall et al. 2020] learn continuous volumetric representations of radiance from images and render novel views via ray integration. Recent methods use 3D Gaussian Splatting for radiance caching [Bauer et al. 2026], and Gaussian mixtures for path guiding [Vorba et al. 2019]. Diolatzis et al. [2024]

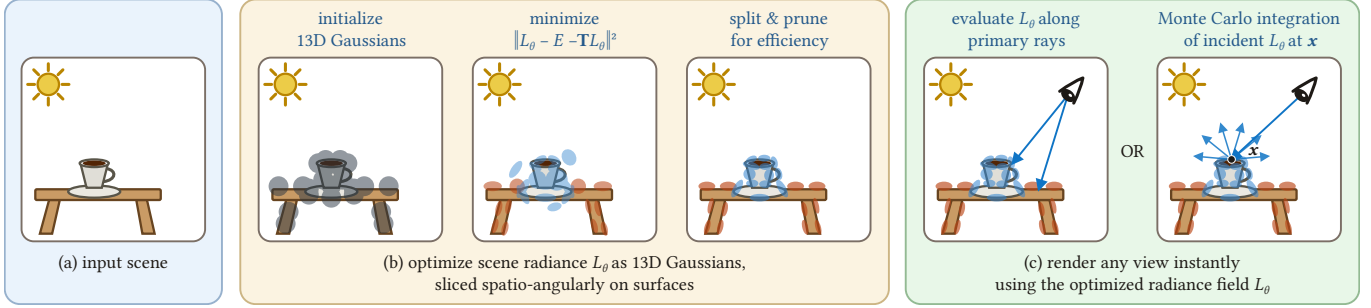


Fig. 2. Flatland illustration of our method. We represent the solution to the light transport equation as a sum of 13D Gaussians over positions, directions, normals, and material properties—only the spatial slice is shown. Given an input scene (a), we (b) optimize the global scene radiance and (c) render multi-view images from it. Gaussians are initialized on scene surfaces and fitted to minimize the residual of the rendering equation. We render novel views either by evaluating the optimized radiance at the primary hit (c. left) or by tracing secondary rays and using the model as the incident radiance at $\mathbf{x}$ (c. right). Whether fitting or rendering, the model is only ever queried at ray-traced points, so only the first visible surface contributes and visibility needs no explicit treatment.

fit rendered light fields with Gaussians, conditioned on G-Buffers, using geometry, material, and normalized scene-variable information to efficiently model high-quality, view- and scene-dependent appearance. These approaches have triggered numerous extensions including for dynamic scenes [Coomans et al. 2024]. While highly effective for view synthesis, these methods typically approximate observed radiance distributions (i.e. pre-acquired or pre-rendered data) and are not physically based nor solve the rendering equation.

Neural Approaches. Neural radiosity [Hadadan et al. 2021] represents radiance using a neural network trained to minimize the residual of the rendering equation over a dense grid of points that is unaware of the frequency content of the represented light field. Its extension to dynamic scenes has also been proposed [Su et al. 2024]. A key limitation of these approaches is their lack of adaptivity to scene content, resulting in a large memory footprint due to uniform discretizations (e.g., 5D grids). Recent work mitigates this issue by attaching neural representations to the scene geometry [Su et al. 2025], improving efficiency but still tying the representation to the underlying mesh and material properties, or by using multiresolution hash encoding [Müller et al. 2022].

3 Technical Description

Our goal is to optimize a radiance field to solve the rendering equation in a synthetic scene. The equilibrium of light transport in a scene is governed by the rendering equation [Kajiya 1986]

$$L(\mathbf{x}, \omega) = E(\mathbf{x}, \omega) + \int_{\Omega} L_i(\mathbf{x}, \omega') f(\mathbf{x}, \omega, \omega') \mathbf{n} \cdot \omega' d\omega', \quad (1)$$

where Ω is the upper hemisphere of directions, $L_i(\mathbf{x}, \omega')$ is the incident radiance at $\mathbf{x}$ along ω' , f is the reflectance function, $\mathbf{n}$ is the normal at $\mathbf{x}$, and E is emitted radiance. In operator form, with the linear light transport operator $\mathbf{T}$, this equation is [Arvo 1995]:

$$L = E + \mathbf{T}L. \quad (2)$$

We seek to solve Equation 2 with a parameterized radiance function $\theta \mapsto L_\theta$ with $\theta \in \mathbb{R}^n$, by minimizing its residual:

$$\theta^* = \operatorname{argmin}_{\theta} \|r_\theta\|^2 \quad \text{with} \quad r_\theta = L_\theta - E - \mathbf{T}L_\theta.$$

In contrast to the Neural Radiosity framework that represents L using a neural network [Hadadan et al. 2021], we use an explicit sum of localized Gaussian kernels, which allows adapting the location and number of functions to the spatial and angular frequency content of the solution L_θ , leading to lower memory requirements.

Our training objective is the squared L_2 norm of the residual of the rendering equation over the surface area and angular domains:

$$\mathcal{L}(\theta) = \|r_\theta\|^2 = \int_S \int_{\Omega} r_\theta(\mathbf{x}, \omega)^2 d\omega d\mathbf{x}. \quad (3)$$

We compute L_θ by combining a classical optimization loop with additional steps to optimize the number of kernels—and therefore the memory cost—of our representation of the actual solution to equation 2. The remainder of this section describes our function space and optimization procedure.

Parameterized radiance space. We represent L_θ as a weighted sum of M (variable) 13D-Gaussians over position $\mathbf{x}$, direction ω , normal $\mathbf{n}$, material attributes (albedo a and roughness r):

$$L_\theta(\mathbf{x}, \omega) = \sum_{k=1}^M c_k e^{-(\mathbf{q} - \mathbf{m}_k)^\top \Sigma_k^{-1} (\mathbf{q} - \mathbf{m}_k)}, \quad (4)$$

where $\mathbf{q} = (\mathbf{x}, \omega, \mathbf{n}, a, r) \in \mathbb{R}^{13}$, $\mathbf{m}_k$ and Σ_k are the mean and covariance matrix of the k^{th} Gaussian and c_k is its radiance scaling coefficient. While albedo and positions are naturally 3D, normals and directions are plunged in $\mathbb{R}^3$, and roughness is unidimensional. This approach, similar to the one of Diolatzis et al. [2024], enables sharing Gaussian contributions between nearby points in the scene even when the points do not belong to one contiguous region. For instance, surfaces with spatially-varying materials could share a Gaussian across regions of identical albedo and roughness. See Section 4 for a discussion on replacing Σ with a block-diagonal matrix. Note that this set of variables does not limit the range of materials that can be used in a scene, but simply facilitates a better adaptation of the Gaussians to variations in the illumination. We initialize using 10K Gaussians centered at randomly sampled surface points and directions, with low-intensity colors before optimization.

Gradient estimation. The loss in Equation 3 is estimated using Monte Carlo integration and gradients of the loss with respect to θ

are estimated using stochastic gradient descent. Surface points $\mathbf{x}_j$, outgoing (resp. incoming) directions ω_j (resp. ω'_j) are generated following adapted probability density functions $p(\mathbf{x}, \omega)$. The gradients of the loss are thereby estimated as:

$$\nabla_{\theta} \mathcal{L}(\theta) \approx \frac{1}{N} \sum_{j=1}^N \frac{2r_{\theta}(\mathbf{x}_j, \omega_j) \nabla_{\theta} r_{\theta}(\mathbf{x}_j, \omega_j)}{p(\mathbf{x}_j, \omega_j)}. \quad (5)$$

Ideally, unbiased gradient estimates would require the inner integrals in $r_{\theta}(\mathbf{x}_j, \omega_j)$ and $\nabla_{\theta} r_{\theta}(\mathbf{x}_j, \omega_j)$ to be evaluated using independent samples. Following the semi-gradient formulation of Cho and Cho [2024], we instead drop the gradient of the inner integral entirely, backpropagating only through $r_{\theta}(\mathbf{x}_j, \omega_j)$, trading a small amount of bias for a significant reduction in gradient variance.

Densification and pruning. We adapt the density of Gaussians where the residual is large, improving the efficiency of the Gaussian mixture L_{θ} via splitting/spawning and pruning steps.

We split Gaussians that exhibit large gradients within specific subspaces since this indicates under-resolved variations [Kerbl et al. 2023]. We split them along the principal axis of the corresponding subspace. While splitting refines the existing structure, it cannot introduce primitives in regions with no coverage. Thus we additionally perform residual-driven spawning: while computing the residual $r(\mathbf{x}_j, \omega_j)$ in Equation 3, we select the top- k samples with the highest contribution, and initialize new Gaussians centered at the corresponding locations $\mathbf{q}(\mathbf{x}_j, \omega_j)$. Their initial radiance coefficient c_k is set proportional to the estimated scattered radiance at these samples given by the right-hand side of the residual $c_k = E(\mathbf{x}, \omega) + \text{TL}_{\theta}(\mathbf{x}, \omega)$, which naturally estimates $L_{\theta}(\mathbf{x}, \omega)$ for subsequent optimization.

We prune Gaussians based on their relative normalized contributions, r_k for each Gaussian G_k , w.r.t. the global radiance field L_{θ} . We restrict the integration domain to the effective support $\mathcal{D}_k$ of each Gaussian, i.e. the region of $\mathcal{S} \times \Omega$ where the exponent of G_k exceeds a minimum value $\beta = -14$:

$$r_k = \int_{\mathcal{D}_k} \frac{c_k G_k(\mathbf{q}(\mathbf{x}, \omega) - \mathbf{m}_k)}{L_{\theta}(\mathbf{x}, \omega)} d\mathbf{x} d\omega. \quad (6)$$

We estimate this integral via the Monte-Carlo estimator

$$r_k \approx \frac{1}{N} \sum_{j=1}^N \frac{c_k G_k(\mathbf{q}(\mathbf{x}_j, \omega_j) - \mathbf{m}_k)}{L_{\theta}(\mathbf{x}_j, \omega_j)}. \quad (7)$$

Gaussian kernels whose contribution r_k falls below a threshold τ are considered negligible and are removed from the representation. We use $\tau = 8 \times 10^{-3}$ for all experiments. Thus, we prune based on relative energy contribution rather than absolute magnitude, allowing both bright and low-intensity regions to be treated consistently.

3.1 Model evaluation and rendering

We produce images from the learned model using two alternative ways: (1) GS which simply evaluates L_{θ} for every visible point $\mathbf{x}$ in the scene along directions ω pointing toward the camera; or (2) GS + MC, where one bounce of rays are traced from the camera and the right-hand side of Equation 1 is evaluated by integrating the incident illumination L_i at this point using $L_i(\mathbf{x}, \omega') = L_{\theta}(\mathbf{y}, \omega'')$

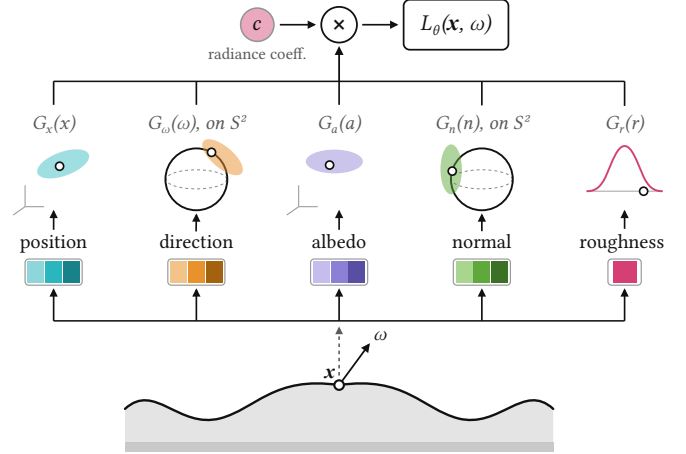


Fig. 3. A visualization of our Gaussian representation. Albedo, normal and roughness are extracted from query point $\mathbf{x}$ and serve as additional selection features for the model. Note that Gaussians in albedo and normal have no rotation because of their diagonal representation.

denoting $(\mathbf{y}, \omega'')$ as the coordinates of the point seen from $\mathbf{x}$ in direction ω' .

Both optimization and rendering queries necessitate the evaluation of contributions from potentially large numbers of Gaussian kernels. While queries generated during GS rendering exhibit strong spatial and directional coherence, this is not the case during training or when rendering with GS + MC, where samples are drawn randomly over both position and direction, which precludes screen-space tiling strategies, such as those used in Kerbl et al. [2023].

For efficient evaluation of L_{θ} , we introduce a simple yet effective world space tile-based culling strategy that drastically reduces the number of Gaussian functions to evaluate per query. Each query is assigned a 64-bit Morton code based on the index of the shape it lies on and its 3D position. Queries are then sorted according to their Morton codes and grouped into tiles by truncating the code to a fixed prefix length. This grouping ensures spatial locality while improving memory coherence during evaluation. For each tile, we compute conservative bounding ranges across all 13 input dimensions. These bounds are obtained via parallel reductions over the queries assigned to each tile and are used to perform Gaussian/tile intersection tests to cull irrelevant primitives. If the expected exponent falls below a predefined threshold, the Gaussian contribution to the tile is considered negligible and is discarded, effectively truncating each kernel to a finite region of support. This hierarchical culling strategy drastically reduces the number of Gaussian kernel evaluations per query. Typically on the BEDROOM scene this strategy allows us to only evaluate 76 Gaussian kernels on average per pixel, among a total of 22K.

4 Implementation

We implemented our optimization and rendering steps within the Mitsuba / Python framework, which provides easy management of scenes and lighting primitives [Jakob et al. 2022]. The Gaussian representation and training pipeline are implemented using PyTorch.

To achieve high efficiency during both training and inference, model evaluation and analytic derivative computations are implemented through custom Taichi kernels [Hu et al. 2019].

Covariance matrices. In the naïve version of our algorithm, covariance matrices are represented using their Cholesky decomposition $\Sigma = \mathbf{L}^\top \mathbf{L}$, where $\mathbf{L}$ is a lower 13×13 triangular matrix. With this choice we have

$$\mathbf{v}^\top \Sigma^{-1} \mathbf{v} = \|\mathbf{L}^{-1} \mathbf{v}\|^2,$$

which means that computing each Gaussian term in Equation 4 boils down to solving a single 13D triangular linear system. We found that an interesting trade-off is to turn the full 13-dimensional Gaussian representations into a product of lower-dimensional Gaussians, thereby neglecting the possible correlation between some of the dimensions. Each Gaussian G_k is modeled as a product of independent Gaussians over these subspaces. Although this factorization limits the ability to capture cross-subspace correlations, we find that it has a relatively low impact on the quality of the converged models, while significantly improving performance by maintaining spatial locality in 3D. In all results shown in this paper, position, direction, material and normals are treated as separate subspaces. Following Kerbl et al. [2023], the covariance matrices of the position and direction components are decomposed into a scaling matrix $\mathbf{S}$ and a rotation matrix $\mathbf{R}$:

$$\Sigma = \mathbf{R} \mathbf{S} \mathbf{S}^\top \mathbf{R}^\top,$$

making the evaluation of $\mathbf{v}^\top \Sigma^{-1} \mathbf{v}$ even faster.

Gaussian colors and covariance scales both use an exponential activation, since radiance and scale values lie in the range $[0, \infty)$.

Loss function. Because radiance is optimized in linear space, applying the naïve loss function of Equation 3 would prioritize learning bright regions at the expense of darker ones. To mitigate this, we found very effective to normalize the loss by the current prediction of the model:

$$\mathcal{L}(\theta) = \left\| \frac{r_\theta}{\nabla_0(L_\theta) + \epsilon} \right\|^2, \quad (8)$$

where $\nabla_0(L_\theta)$ is in effect L_θ , but considered constant when differentiating.

Additionally, we observed that regions containing near-coincident surfaces—such as object contact areas or thin geometric interfaces—present sharp discontinuities that the optimization tries to resolve by strongly elongating the Gaussian kernels along their positional components. We mitigate this by introducing an anisotropy penalty term $\mathcal{P}(\theta)$ in the loss function for the three positional components. Let $\mathbf{s}_i \in \mathbb{R}^3$ denote the logarithmic scaling parameters of Gaussian i , with components $\mathbf{s}_i = (s_{i,1}, s_{i,2}, s_{i,3})$. We define the mean logarithmic scale as

$$\bar{s}_i = \frac{1}{3} \sum_{n=1}^3 s_{i,n}. \quad (9)$$

The anisotropy regularization term is defined over all Gaussians as

$$\mathcal{P}(\theta) = \frac{1}{N} \sum_{i=1}^N \text{ReLU} \left(\sum_{n=1}^3 (s_{i,n} - \bar{s}_i)^2 - \epsilon \right), \quad (10)$$

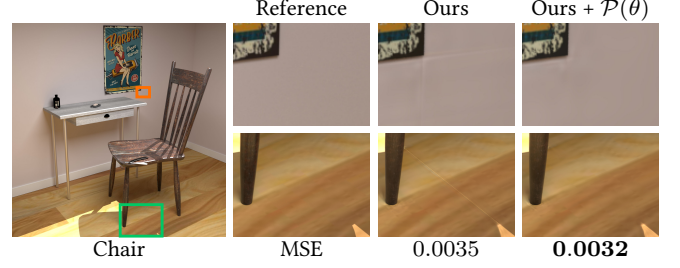


Fig. 4. Adding the extra penalty term $\mathcal{P}(\theta)$ in the loss to control the anisotropy of Gaussian functions allows to avoid artifacts due to overly elongated Gaussians such as the thin bright stripe in the middle inset.

where ϵ controls the tolerated level of anisotropy. In our implementation, we use $\epsilon = 2.5$. Figure 4 illustrates the visual improvement brought by this anisotropy regularization term.

In summary, the full expression of the loss function we are using during the optimization step is

$$\mathcal{L}(\theta) = \left\| \frac{r_\theta}{\nabla_0(L_\theta) + \epsilon} \right\|^2 + \lambda_{\text{aniso}} \mathcal{P}(\theta),$$

where λ_{aniso} monitors the compromise between the shape of the Gaussian in position and the resulting fit of Equation 1. We used $\lambda_{\text{aniso}} = 0.008$ in all our examples.

Optimization. During the estimate of gradients (Equation 5) we sample Q incident directions and Q explicit light samples in r_θ for each outgoing direction ω_j , so as to reduce variance due to the generally significant contribution of direct illumination. In all our examples, we used $Q = 32$.

Since the evaluation of r_θ requires L_θ to be computed twice, we concatenate the queries of both evaluations into a single tensor that is evaluated once, making the economy of one spatial reordering and culling per iteration. Additionally, culling is reused during the backward pass to speed up backpropagation.

During the Gaussian splitting phase, each candidate Gaussian is split along a single subspace at a time with priority given to spatial dimensions, in order to ensure that the child Gaussians accurately approximate their parent, and to preserve continuity during optimization. The child Gaussians are initialized by offsetting them in opposite directions along the principal axis of the parent, at a distance $d = 0.5$ times the principal scale, and scaling their covariance by a factor $k = 0.8$ along this axis.

We use the Adam optimizer [Kingma and Ba 2015]. All runs use the same hyperparameters, and learning rates for $(\mathbf{m}, \mathbf{R}, \mathbf{S}, c)$ are set to $(3 \times 10^{-4}, 1 \times 10^{-3}, 5 \times 10^{-3}, 1 \times 10^{-2})$. Gaussians are split every 400 iterations, and 500 new Gaussians are spawned every 500 iterations. Both splitting and spawning stop at iteration 15000. Gaussians are pruned every 2000 iterations throughout training.

Rendering. We exploit the separability of our Gaussian functions to derive an efficient separable culling criterion. Specifically, we evaluate a Mahalanobis distance between the Gaussian and the tile bounds independently in each subspace. Because the Gaussian exponent decomposes additively across decorrelated subspaces, these

per-subspace distances can be accumulated to form a global bound. We first apply this test to the 1D subspaces, which provides a cheap and effective early rejection mechanism, then a similar test is performed for the 3D spatial and directional components using the tile’s bounding box. For 3D subspaces, a good conservative Mahalanobis distance between a tile bounding box and a Gaussian can be computed as follows: we first compute the Mahalanobis distance from the Gaussian to the center of the bounding box, from which we subtract the maximum distance from the center of the bounding box to one of its corners. Similarly to Hadadan et al. [2021], the model does not directly learn light distributions over perfectly specular surfaces. Instead, during both rendering and optimization, paths are extended until the first smooth surface.

5 Discussion

Our representation shares similarities with N-dimensional Gaussians [Diatz et al. 2024]: by centering Gaussians on scene features (albedo, normals), high-frequency content is encoded directly in the input space rather than learned. Their method fits pre-generated path-traced images, solving transport beforehand, whereas we solve the rendering equation itself. We keep fewer principal component axes than their full representation (see Section 4). Dropping this generality allows us to derive a more efficient culling strategy well suited for real-time rendering, while reducing the memory footprint of our representation.

Our method belongs to the larger family of kernel-based approximations. For instance, our adaptive Gaussian mixture is equivalent to constructing a Gaussian process with a location-dependent covariance function, i.e. a nonstationary kernel where the notion of distance changes across space. Similarly, neural network approximations (e.g. Neural Radiosity [Hadadan et al. 2021]) implicitly define a kernel, but use fixed positions on a predefined grid. As a comparison, our model not only learns kernel weights but also positions $\mathbf{m}_k$ in 13D, shape (covariance matrices Σ_k) and number of nodes to use. Finally, our model can also be understood as a learned radial basis function neural network, that implicitly constructs a nonstationary Gaussian process kernel. In different domains Gaussian processes and kernel methods have been applied to solve partial differential equations [Chen et al. 2021; Xu et al. 2025], while sparsity-promoting techniques improve efficiency [Vakili et al. 2022].

One advantage of our adaptive representation is that it avoids the pitfalls of bandwidth-limited constructions [Soler et al. 2022].

Since L_θ appears twice in the expression of the residual r_θ , the model must be evaluated twice: for outgoing radiance, and at the intersection points for incident radiance. This can be interpreted numerically as a recursive training approach, where the model iteratively uses its own previous state as the best estimate so far to the solution of the rendering equation.

Limitations. The calculation of r_θ is currently conducted using uniform sampling of surfaces, which favors the learning of Gaussians with a large positional footprint. We noticed that glossy reflections on curved surfaces are learned quicker with a sampling that is based on curvature. We think that sampling according to a local prediction of angular and spatial bandwidth [Belcour et al. 2013] would eventually focus the workload on regions that need it.

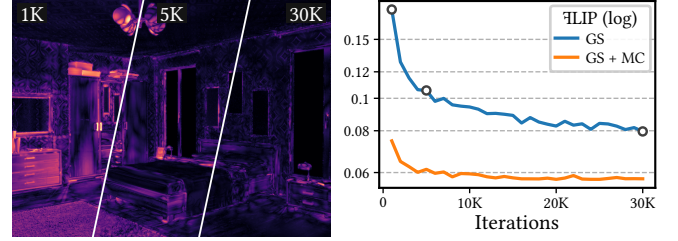


Fig. 5. *Left*: 3D LIP error maps of BEDROOM, each computed from a different checkpoint of the same run. Brighter is worse. *Right*: mean 3D LIP against training iteration for both render modes at 1920×1080 64 spp.

Our current implementation runs on static scenes only. We think that if single objects are moved or scene parameters such as reflectance properties or lighting are changed, it would be highly beneficial to start from the previous solution and continue optimizing from there. Another possible avenue for predefined changes would be to add an extra temporal dimension to Gaussian functions.

Our culling approach relies on the assumption that queries grouped within a tile are coherent both spatially and angularly. While this assumption holds well for primary camera rays during GS rendering, it becomes weaker during training and GS + MC rendering, where queries exhibit reduced spatio-directional correlation. Consequently, directional culling becomes less effective, resulting in a 1.5× inference slowdown in the worst case. Future work could investigate direction-aware tiling or clustering schemes to better preserve spatio-directional locality under these workloads.

Like any Gaussian-based representation, ours is smooth by construction and resolves a discontinuity only where an input announces it. Albedo, normal and roughness jump at material and geometric edges, so boundaries aligned with them stay sharp (Fig. 6). Position and direction vary continuously, so radiance discontinuous in those alone is interpolated and softened. Near-specular reflections show it most: the chimney’s dark base in LIVING ROOM, and the bright highlight along the seat edge in CHAIR (Fig. 8).

6 Results

A typical example of exploring the solution to the light transport equation in a complex scene showing glossy and specular materials, complex geometry and textures is shown in Figure 1. Once the solution is optimized, all three images are obtained by simply slicing the Gaussians w.r.t. the scene geometry and materials, and the camera direction. For each viewpoint we show the rendered vs. flattened Gaussians, the directional component of which makes their visualized support ellipsoids move along with the camera.

Figure 5 illustrates the convergence of our method. We use the HDR-3D LIP error metric [Andersson et al. 2021] for quantitative comparisons. Although the limited total number of Gaussians used to represent the solution inherently biases the result, visual and 3D LIP comparisons with the path traced reference prove very satisfactory.

In Table 1 we present an ablation study that illustrates the relative impact of Gaussian dimensionality, spawning, splitting and regularization term on the mean square error for all scenes presented in this paper. Except for the RINGS scene where these stages have little

Table 1. MSE comparison of ablation runs (lower is better). When the full model is not ranked first (e.g., RINGS), the performance gap is marginal and within run-to-run stochastic variance. Colors mark the **best**, **second-best**, and **third-best** result per column.

Ablation	BEDROOM	CHAIR	DINING ROOM	LIVING ROOM	RINGS	STAIRCASE	VEACH AJAR
No material (9D)	0.02313	0.00188	0.00205	0.02368	0.02001	0.00759	0.01440
No normal (10D)	0.01137	0.00215	0.00259	0.04286	0.02005	0.00463	0.01179
No Split	0.03354	0.00348	0.00324	0.06272	0.02016	0.00353	0.02557
No Spawn	0.01503	0.00167	0.00204	0.03452	0.02001	0.00239	0.00591
No $\mathcal{P}(\theta)$	0.01339	0.00130	0.00207	0.02820	0.02000	0.00126	0.00538
Full	0.00636	0.00116	0.00189	0.02203	0.02001	0.00131	0.00444

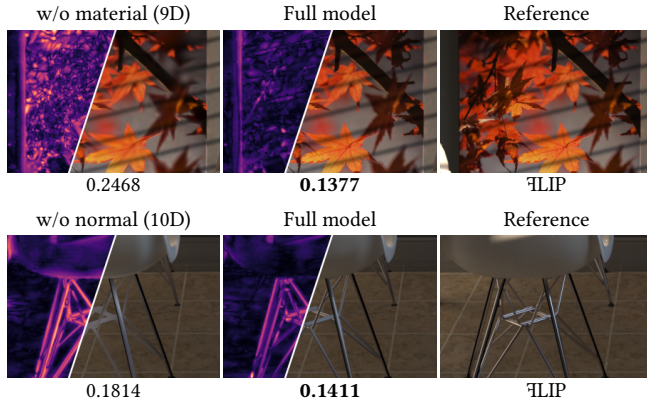


Fig. 6. The extra dimensions of the Gaussians allow accommodation of local variation. For example, the Gaussian visualizations show that not including albedo in the Gaussian dimensions requires more (scene total 37.9K) Gaussians in textured regions than with albedo (scene total 35.2K Gaussians).

to no impact, the improvement brought by these steps in terms of image quality stands out clearly. When splitting is disabled, as it is normally the main source of added detail, we increase the number of initial Gaussians to 30K for fair comparison. In Figure 6 we illustrate the visual impact of not including material and normal dimensions within the 13 dimensions of the space of Gaussians.

We compare against Neural Radiosity [Haddadan et al. 2021], extended with two feature encodings: the multi-resolution hash grid of Müller et al. [2022] and the vertex features of Su et al. [2025]. We use the official implementation of Su et al. with their default settings (four learnable parameters per vertex) and adapt it to support hash grids: eight resolution levels, a coarsest resolution of 4, a scale factor of 2 between successive levels, and eight learnable parameters per voxel. Both baselines are trained for the same number of iterations and with the same network (four hidden layers of 256 neurons) so that they differ only in how the input is encoded. Training and rendering are performed on an NVIDIA RTX 4080 SUPER for all methods.

Table 2 reports training and rendering cost against the stronger of the two baselines, vertex features. Our models train for a fixed 30K iterations (7–12 minutes), whereas the baselines were trained until their artifacts cleared, 40K–100K iterations (1.5–4.5 hours): a 10.0–23.4× reduction. At one sample per pixel we render 2.2–4.5× faster, and reach a lower FLIP on five of the six scenes.

Table 2. Training and rendering cost against Neural Radiosity with vertex features, NR (VF). Model is the size of the trained representation on disk. Render is the frame time at 1920 × 1080, 1 spp.

Scene	Method	Iter.	Train	Model	Render	FLIP
BEDROOM	NR (VF)	40K	2h10m	26.4 MB	70.4 ms	0.0774
	Ours	30K	10m48s	4.4 MB	18.1 ms	0.0824
CHAIR	NR (VF)	40K	1h42m	1.9 MB	52.3 ms	0.0602
	Ours	30K	10m07s	4.3 MB	16.1 ms	0.0486
DINING ROOM	NR (VF)	40K	1h54m	6.0 MB	60.2 ms	0.1012
	Ours	30K	9m17s	5.4 MB	13.5 ms	0.0831
LIVING ROOM	NR (VF)	100K	4h29m	5.3 MB	61.5 ms	0.1362
	Ours	30K	11m30s	5.8 MB	16.5 ms	0.1177
RINGS	NR (VF)	40K	1h30m	0.9 MB	44.0 ms	0.0509
	Ours	30K	7m37s	1.0 MB	19.9 ms	0.0425
VEACH AJAR	NR (VF)	60K	2h25m	6.5 MB	46.5 ms	0.1499
	Ours	30K	12m08s	4.1 MB	13.0 ms	0.0641
Average	NR (VF)	53K	2h22m	7.8 MB	55.8 ms	0.0960
	Ours	30K	10m15s	4.2 MB	16.2 ms	0.0730

Figure 7 provides a visual comparison of the three methods at equal sample count. Additional viewpoints in LIVING ROOM are provided at 1 spp in Figure 1. Model size (the trained representation as stored on disk) behaves differently for the two encodings: the hash grid is a fixed 35.7 MB irrespective of the scene, while vertex features mostly scale with the vertex count of the geometry and range from 0.9 MB to 26.4 MB. Our representation is bounded by the number of Gaussians rather than by either grid resolution or tessellation, and stays between 1 MB and 5.9 MB on all scenes. The peak GPU memory of the process—including the renderer’s own allocations—is 3.1–4.3 GB during training and 3.0–4.0 GB when rendering in GS, rising to 3.9–6.1 GB in GS + MC, regardless of the sample count.

Table 3 connects rendering mode, sample count, error, and frame time for our method. Since GS mode queries the representation at the primary hit, samples only buy antialiasing and noise reduction on traced specular paths. GS + MC takes one Monte Carlo bounce first, and so is noisier at one sample and more accurate once converged, overtaking GS at 1024 samples on every scene.

Finally, in Figure 8 we show that our method can successfully handle very different situations typical of global illumination: geometric complexity, caustics, glossy and specular surfaces, textures. Gaussian components naturally decompose into diffuse + glossy components when BSDFs contain multiple lobes.

Table 3. Rendering mode vs. sample count for our method. Eval is the model evaluation alone, so the gap to 1 spp frame time is renderer overhead. On RINGS secondary rays leave the scene almost everywhere, so GS + MC evaluates the model far less than GS and ends up cheaper.

Scene	Mode	1 spp			64 spp			1024 spp		
		Eval	Render	TLIP	Render	TLIP	Render	TLIP	Render	TLIP
BED-ROOM	GS	10.5 ms	18.1 ms	0.101	1.15 s	0.082	18.03 s	0.082		
	GS + MC	14.3 ms	31.1 ms	0.276	1.88 s	0.058	30.02 s	0.027		
CHAIR	GS	11.2 ms	16.1 ms	0.052	0.99 s	0.049	15.44 s	0.048		
	GS + MC	11.8 ms	19.3 ms	0.329	1.23 s	0.064	19.71 s	0.025		
DINING ROOM	GS	10.2 ms	13.5 ms	0.090	0.86 s	0.083	13.09 s	0.083		
	GS + MC	14.6 ms	21.2 ms	0.442	1.36 s	0.106	21.66 s	0.041		
LIVING ROOM	GS	10.6 ms	16.5 ms	0.125	1.02 s	0.118	16.26 s	0.117		
	GS + MC	13.9 ms	26.3 ms	0.547	1.67 s	0.140	26.51 s	0.056		
RINGS	GS	14.2 ms	19.9 ms	0.062	1.28 s	0.042	20.04 s	0.039		
	GS + MC	3.8 ms	14.7 ms	0.389	0.94 s	0.074	14.81 s	0.027		
VEACH	GS	10.1 ms	13.0 ms	0.070	0.82 s	0.064	12.83 s	0.064		
AJAR	GS + MC	14.2 ms	19.4 ms	0.350	1.24 s	0.094	19.70 s	0.038		

7 Conclusion

We presented a novel method for solving the rendering equation based on a mixture of Gaussian functions, which allows real-time exploration of the solution to the problem of global illumination in static scenes. The very low memory footprint of our representation makes it easy to embed in a viewer for instance to explore architectural constructions.

Our plans for future work include extending our method to the integro-differential equation for global illumination in participating media, and the export of light fields. The intrinsic local support of our representation in frequency space also naturally allows to isolate/scale/cancel parts of the solution to the light transport equation by their corresponding spatial and angular frequency. This could for instance be used to box-filter the signal as a post-process for artistic reasons. More generally our model could be used for baking global illumination thanks to its very economic memory footprint.

Acknowledgments

We thank the anonymous reviewers for their careful reading of our manuscript and for their valuable feedback.

We are also grateful to the artists whose scenes we use. BEDROOM by SlykDrako and VEACH AJAR by Benedikt Bitterli are released under CC0. DINING ROOM, LIVING ROOM and STAIRCASE are by Wig42 and are used under CC BY (<https://creativecommons.org/licenses/by/3.0/>). The Mitsuba versions of these five scenes are by Benedikt Bitterli. CHAIR and RINGS are courtesy of Hadadan et al. [2021], who adapted them from scenes by BlendSwap users barcin and IAMCUBEMAN, both used under CC BY.

References

Pontus Andersson, Jim Nilsson, Peter Shirley, and Tomas Akenine-Möller. 2021. Visualizing Errors in Rendered High Dynamic Range Images. In *Eurographics 2021 - Short Papers*, Holger Theisel and Michael Wimmer (Eds.). The Eurographics Association. doi:10.2312/egs.20211015

James Arvo. 1995. The Role of Functional Analysis in Global Illumination. In *Rendering Techniques '95*. Springer, Vienna, 115–126. doi:10.1007/978-3-7091-9430-0_12

David Bauer, Qi Wu, Hamid Gadirov, and Kwan-Liu Ma. 2026. GSCache: Real-Time Radiance Caching for Volume Path Tracing Using 3D Gaussian Splatting. *IEEE Transactions on Visualization & Computer Graphics* 32, 01 (Jan. 2026), 901–911. doi:10.1109/TVCG.2025.3634634

Laurent Belcour, Thomas Deliot, Wilhem Barbier, and Cyril Soler. 2022. A Data-Driven Paradigm for Precomputed Radiance Transfer. In *SIGGRAPH 2022 - Conference & Exhibition on Computer Graphics & Interactive Techniques*. Vancouver, Canada, 1–8. <https://inria.hal.science/hal-03726655>

Laurent Belcour, Cyril Soler, Kartic Subr, Nicolas Holzschuch, and Fredo Durand. 2013. 5D Covariance tracing for efficient defocus and motion blur. *ACM Trans. Graph.* 32, 3, Article 31 (July 2013), 18 pages. doi:10.1145/2487228.2487239

Yifan Chen, Bamdad Hosseini, Houman Owahdi, and Andrew M. Stuart. 2021. Solving and learning nonlinear PDEs with Gaussian processes. *J. Comput. Phys.* 447 (2021), 110668. doi:10.1016/j.jcp.2021.110668

In-Young Cho and Jaewoong Cho. 2024. Fast and Accurate Neural Rendering Using Semi-Gradients. arXiv:2410.10149 [cs.CV] <https://arxiv.org/abs/2410.10149>

Per Christensen. 2008. Point-based approximate color bleeding. *Pixar Technical Notes* 2, 5 (2008), 6.

Arno Coomans, Edoardo A. Dominci, Christian Döring, Joerg H. Mueller, Jozef Hladky, and Markus Steinberger. 2024. Real-time Neural Rendering of Dynamic Light Fields. *Computer Graphics Forum* 43, 2 (2024), e15014. arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1111/cgf.15014 doi:10.1111/cgf.15014

Stavros Diolatzis, Tobias Zirr, Alexander Kuznetsov, Georgios Kopanas, and Anton Kaplanyan. 2024. N-Dimensional Gaussians for Fitting of High Dimensional Functions. In *ACM SIGGRAPH 2024 Conference Papers (SIGGRAPH '24)*. ACM, New York, NY, USA, 1–11. doi:10.1145/3641519.3657502

Cindy M. Goral, Kenneth E. Torrance, Donald P. Greenberg, and Bennett Battaile. 1984. Modeling the interaction of light between diffuse surfaces. In *Proceedings of the 11th annual conference on Computer graphics and interactive techniques (SIGGRAPH '84)*. ACM, New York, NY, USA, 213–222. doi:10.1145/800031.808601

Yijia Guo, Yuanxi Bai, Liwen Hu, Ziyi Guo, Mianzhi Liu, Yu Cai, Tiejun Huang, and Lei Ma. 2024. PRTGS: Precomputed Radiance Transfer of Gaussian Splats for Real-Time High-Quality Relighting. In *Proceedings of the 32nd ACM International Conference on Multimedia (Melbourne VIC, Australia) (MM '24)*. ACM, New York, NY, USA, 5112–5120. doi:10.1145/3664647.3680893

Toshiya Hachisuka, Shinji Ogaki, and Henrik Wann Jensen. 2008. Progressive photon mapping. *ACM Trans. Graph.* 27, 5, Article 130 (Dec. 2008), 8 pages. doi:10.1145/1409060.1409083

Saeed Hadadan, Shuhong Chen, and Matthias Zwicker. 2021. Neural radiosity. *ACM Trans. Graph.* 40, 6, Article 236 (Dec. 2021), 11 pages. doi:10.1145/3478513.3480569

Paul S. Heckbert and James M. Winget. 1991. *Finite Element Methods for Global Illumination*. Technical Report. University of California at Berkeley, USA.

Yuanming Hu, Tzu-Mao Li, Luke Anderson, Jonathan Ragan-Kelley, and Frédo Durand. 2019. Taichi: a language for high-performance computation on spatially sparse data structures. *ACM Trans. Graph.* 38, 6, Article 201 (Nov. 2019), 16 pages. doi:10.1145/3355089.3356506

Kei Iwasaki, Yoshinori Dobashi, Fujiichi Yoshimoto, and Tomoyuki Nishita. 2007. Pre-computed radiance transfer for dynamic scenes taking into account light interreflection. In *Proceedings of the 18th Eurographics Conference on Rendering Techniques (Grenoble, France) (EGSR'07)*. Eurographics Association, Goslar, DEU, 35–44.

Wenzel Jakob, Sébastien Speierer, Nicolas Roussel, Merlin Nimier-David, Delio Vicini, Tizian Zeltner, Baptiste Nicolet, Miguel Crespo, Vincent Leroy, and Ziyi Zhang. 2022. *Mitsuba 3 renderer*. <https://mitsuba-renderer.org>.

Wojciech Jarosz, Derek Nowrouzezahrai, Robert Thomas, Peter-Pike Sloan, and Matthias Zwicker. 2011. Progressive photon beams. *ACM Trans. Graph.* 30, 6 (Dec. 2011), 1–12. doi:10.1145/2070781.2024215

Henrik Wann Jensen. 1996. Global Illumination using Photon Maps. In *Rendering Techniques '96 (Eurographics)*, Xavier Pueyo and Peter Schröder (Eds.). Springer, Vienna, 21–30. doi:10.1007/978-3-7091-7484-5_3

James T. Kajiya. 1986. The rendering equation. *ACM SIGGRAPH Computer Graphics* 20, 4 (Aug. 1986), 143–150. doi:10.1145/15886.15902

Alexander Keller, Jaroslav Krivánek, Jan Novák, Anton Kaplanyan, and Marco Salvi. 2018. Machine learning and rendering. In *ACM SIGGRAPH 2018 Courses (Vancouver, British Columbia, Canada) (SIGGRAPH '18)*. ACM, New York, NY, USA, Article 19, 2 pages. doi:10.1145/3214834.3214841

Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 2023. 3D Gaussian Splatting for Real-Time Radiance Field Rendering. *ACM Transactions on Graphics* 42, 4 (July 2023). <https://repo-sam.inria.fr/fungraph/3d-gaussian-splatting/>

Markus Kettunen, Marco Manzi, Miika Aittala, Jaakko Lehtinen, Frédo Durand, and Matthias Zwicker. 2015. Gradient-domain path tracing. *ACM Trans. Graph.* 34, 4, Article 123 (July 2015), 13 pages. doi:10.1145/2766997

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. In *International Conference on Learning Representations (ICLR)*. <https://arxiv.org/abs/1412.6980>

- Jaroslav Krivánek, Pascal Gautron, Sumanta Pattanaik, and Kadi Bouatouch. 2008. Radiance caching for efficient global illumination computation. In *ACM SIGGRAPH 2008 Classes* (Los Angeles, California) (SIGGRAPH '08). ACM, New York, NY, USA, Article 75, 19 pages. doi:10.1145/1401132.1401228
- Jaakko Lehtinen, Matthias Zwicker, Emmanuel Turquin, Janne Kontkanen, Frédo Durand, François Sillion, and Timo Aila. 2008. A Meshless Hierarchical Representation for Light Transport. *ACM Trans. Graph.* 27, 3 (2008).
- Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. 2020. NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis. In *ECCV*.
- Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. 2022. Instant neural graphics primitives with a multiresolution hash encoding. *ACM Transactions on Graphics* 41, 4 (2022), 1–15. doi:10.1145/3528223.3530127
- Ren Ng, Ravi Ramamoorthi, and Pat Hanrahan. 2004. Triple product wavelet integrals for all-frequency relighting. In *ACM SIGGRAPH 2004 Papers* (Los Angeles, California) (SIGGRAPH '04). ACM, New York, NY, USA, 477–487. doi:10.1145/1186562.1015749
- Y. Ouyang, S. Liu, M. Kettunen, M. Pharr, and J. Pantaleoni. 2021. ReSTIR GI: Path Resampling for Real-Time Path Tracing. *Computer Graphics Forum* 40, 8 (2021), 17–29. arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1111/cgf.14378 doi:10.1111/cgf.14378
- Christoph Schied, Anton Kaplanyan, Chris Wyman, Anjul Patney, Chakravarty R. Alla Chaitanya, John Burgess, Shiqiu Liu, Carsten Dachsbacher, Aaron Lefohn, and Marco Salvì. 2017. Spatiotemporal variance-guided filtering: real-time reconstruction for path-traced global illumination. In *Proceedings of High Performance Graphics* (Los Angeles, California) (HPG '17). ACM, New York, NY, USA, Article 2, 12 pages. doi:10.1145/3105762.3105770
- Peter-Pike Sloan, Jan Kautz, and John Snyder. 2023. *Precomputed Radiance Transfer for Real-Time Rendering in Dynamic, Low-Frequency Lighting Environments* (1 ed.). ACM, New York, NY, USA. https://doi.org/10.1145/3596711.3596749
- Cyril Soler, Ronak Molazem, and Kartic Subr. 2022. A Theoretical Analysis of Compactness of the Light Transport Operator. In *ACM SIGGRAPH 2022 Conference Proceedings* (Vancouver, BC, Canada) (SIGGRAPH '22). ACM, New York, NY, USA, Article 17, 9 pages. doi:10.1145/3528233.3530725
- Rui Su, Honghao Dong, Haojie Jin, Yisong Chen, Guoping Wang, and Sheng Li. 2025. Vertex Features for Neural Global Illumination. In *Proceedings of the SIGGRAPH Asia 2025 Conference Papers* (SA Conference Papers '25). ACM, New York, NY, USA, Article 10, 11 pages. doi:10.1145/3757377.3763826
- Rui Su, Honghao Dong, Jierui Ren, Haojie Jin, Yisong Chen, Guoping Wang, and Sheng Li. 2024. Dynamic Neural Radiosity with Multi-grid Decomposition. In *SIGGRAPH Asia 2024 Conference Papers* (Tokyo, Japan) (SA '24). ACM, New York, NY, USA, Article 23, 12 pages. doi:10.1145/3680528.3687685
- Xin Sun, Kun Zhou, Yanyun Chen, Stephen Lin, Jiaoying Shi, and Baining Guo. 2007. Interactive relighting with dynamic BRDFs. *ACM Trans. Graph.* 26, 3 (July 2007), 27–es. doi:10.1145/1276377.1276411
- Wolfgang Tatzgern, Alexander Weinrauch, Pascal Stadlbauer, Joerg H. Mueller, Martin Winter, and Markus Steinberger. 2024. Radiance Caching with On-Surface Caches for Real-Time Global Illumination. *Proc. ACM Comput. Graph. Interact. Tech.* 7, 3, Article 38 (Aug. 2024), 17 pages. doi:10.1145/3675382
- Sattar Vakili, Jonathan Scarlett, Da-Shan Shiu, and Alberto Bernacchia. 2022. Improved Convergence Rates for Sparse Approximation Methods in Kernel-Based Learning. In *Proceedings of the 39th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 162)*, Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato (Eds.). PMLR, 21960–21983.
- Eric Veach. 1997. *Robust Monte-Carlo Methods for Light Transport Simulation*. Ph.D. Dissertation. Stanford University.
- Eric Veach and Leonidas J. Guibas. 1997. *Metropolis Light Transport* (1 ed.). ACM, New York, NY, USA. https://doi.org/10.1145/3596711.3596744
- Jiří Vorba, Johannes Hanika, Sebastian Herholz, Thomas Müller, Jaroslav Krivánek, and Alexander Keller. 2019. Path guiding in production. In *ACM SIGGRAPH 2019 Courses* (Los Angeles, California) (SIGGRAPH '19). Association for Computing Machinery, New York, NY, USA, Article 18, 77 pages. doi:10.1145/3305366.3328091
- Beibei Wang, Xiangxu Meng, and Tamy Boubekeur. 2015. Wavelet Point-Based Global Illumination. *Computer Graphics Forum (Special Issue on EGSR 2015)* 34, 4 (2015), 143–154.
- Youxin Xing, Gaole Pan, Xiang Chen, Ji Wu, Lu Wang, and Beibei Wang. 2024. Real-time all-frequency global illumination with radiance caching. *Computational Visual Media* 10, 5, 923–936. doi:10.1007/s41095-023-0367-z
- Zhitong Xu, Da Long, Yiming Xu, Guang Yang, Shandian Zhe, and Houman Owjadi. 2025. Toward efficient kernel-based solvers for nonlinear PDEs. In *Proceedings of the 42nd International Conference on Machine Learning (Vancouver, Canada) (ICML '25)*. JMLR.org, Article 2778, 19 pages.
- Alex Yu, Sara Fridovich-Keil, Matthew Tancik, Qinhong Chen, Benjamin Recht, and Angjoo Kanazawa. 2021. Plenoxels: Radiance Fields without Neural Networks. arXiv:2112.05131 [cs.CV] https://arxiv.org/abs/2112.05131
- Harold R. Zatz. 1993. Galerkin radiosity: a higher order solution method for global illumination. In *Proceedings of the 20th Annual Conference on Computer Graphics and Interactive Techniques* (Anaheim, CA) (SIGGRAPH '93). ACM, New York, NY, USA, 213–220. doi:10.1145/166117.166145
- Qiang Zhang, Seung-Hwan Baek, Szymon Rusinkiewicz, and Felix Heide. 2023. Differentiable Point-Based Radiance Fields for Efficient View Synthesis. arXiv:2205.14330 [cs.CV] https://arxiv.org/abs/2205.14330

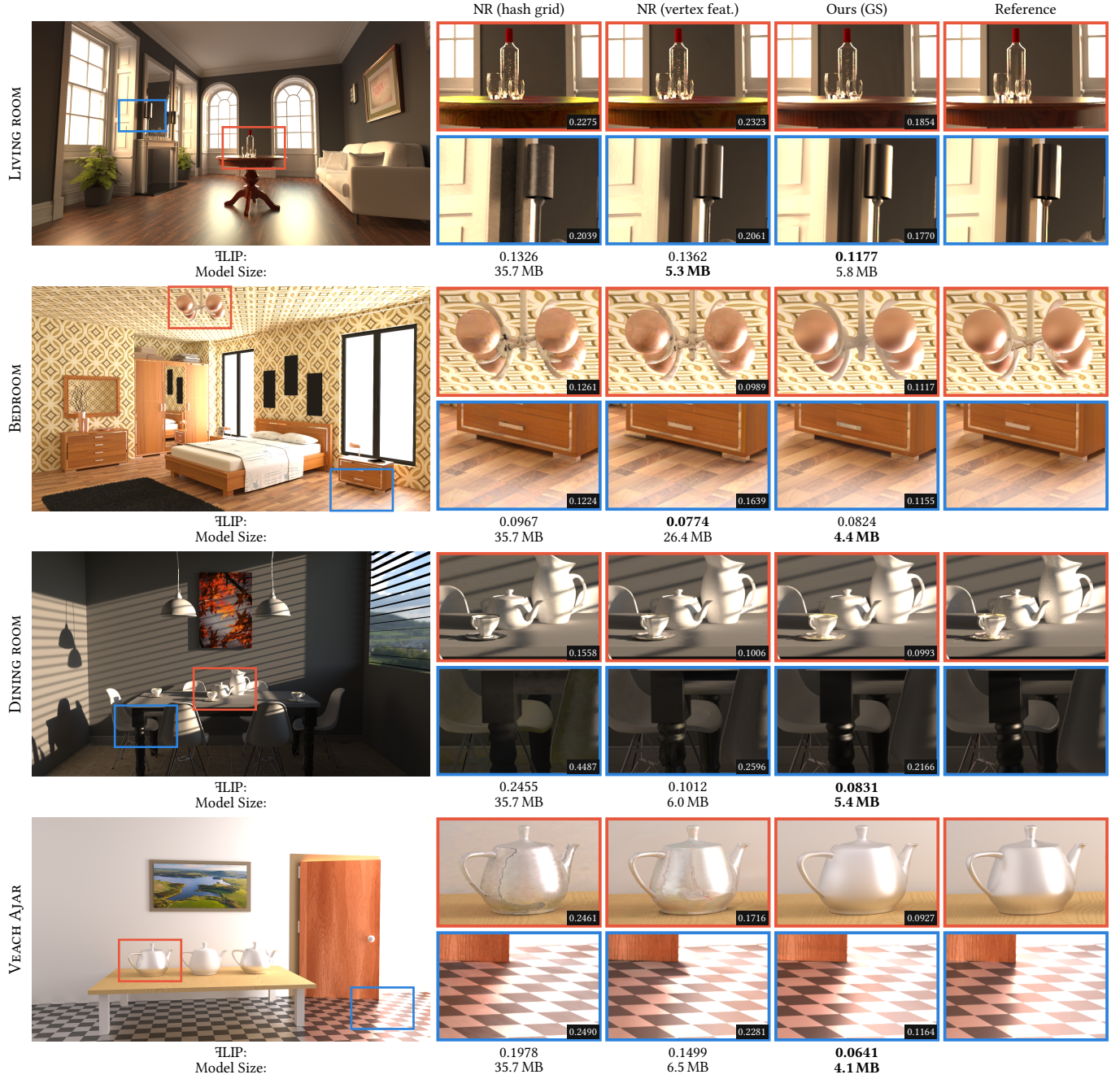


Fig. 7. Visual comparison with Neural Radiosity [Hadadan et al. 2021] extended with Hash Grids [Müller et al. 2022] and Neural Vertex Features [Su et al. 2025] on multiple test scenes. All renders have a resolution of 1920×1080 and use 64 spp. See Table 2 for training and rendering times.

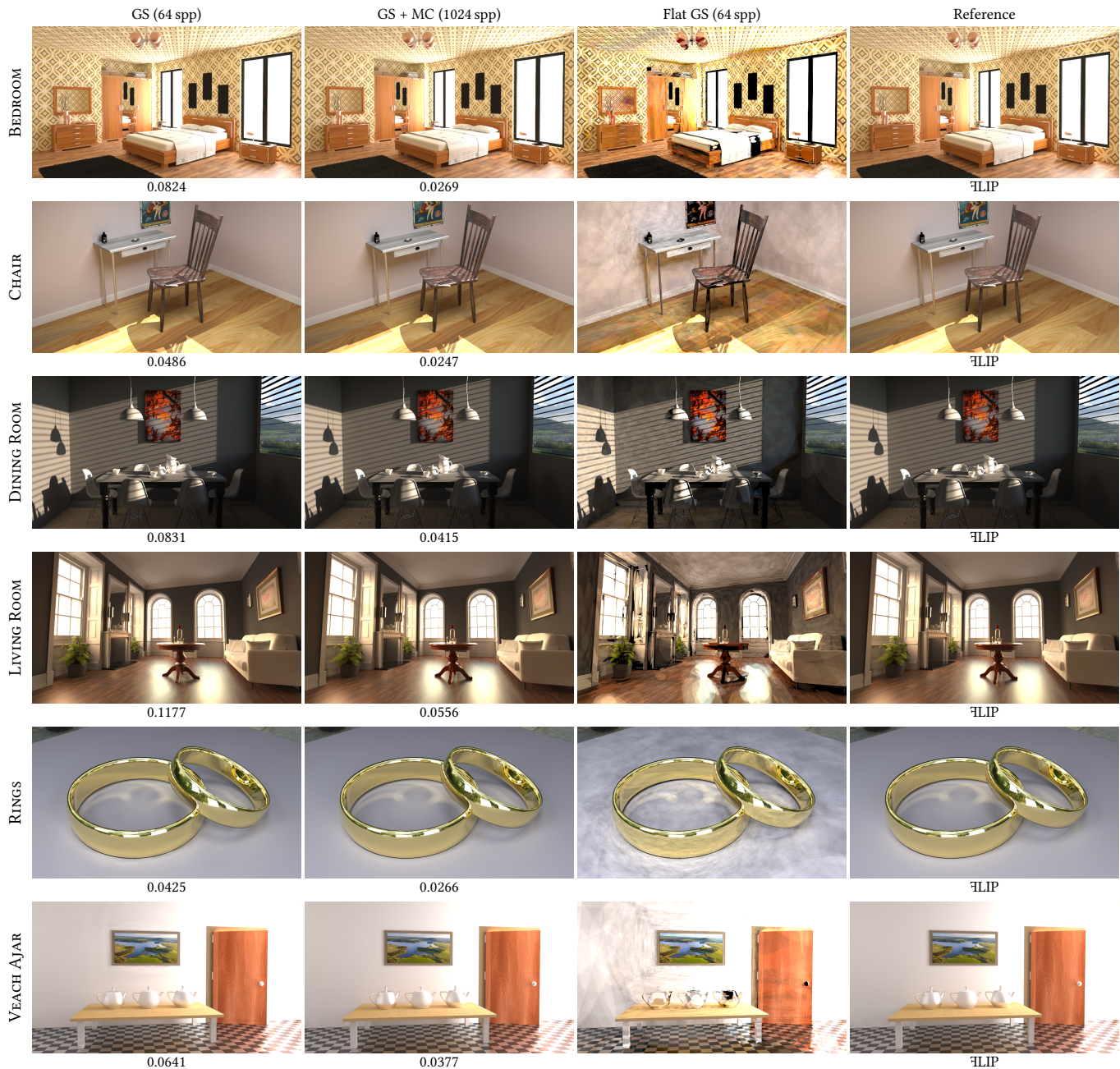


Fig. 8. At the cost of some precomputation, our method allows fast multi-view rendering, with low $\mathcal{F}$ LIP, of challenging light transport settings such as indirect illumination, caustics, high-frequency shadows and specular reflections. In all situations, our GS render (column a.) rapidly approximates the reference. In scenes with strong indirect lighting, our GS + MC render (column b.) improves the quality of the result at 1024 spp. The flat-shaded Gaussians (column c.) show adaptation of covariances to the geometry, shadows, etc. See Table 2 for computation times.