Volume xx (200y), Number z, pp. 1–17

Path Guiding Using Spatio-Directional Mixture Models

A. Dodik[1] & M. Papas[2] & C. Öztireli[3] & T. Müller[4]

[1]Facebook† [2]Disney Research|Studios [3]Cambridge University [4]NVIDIA

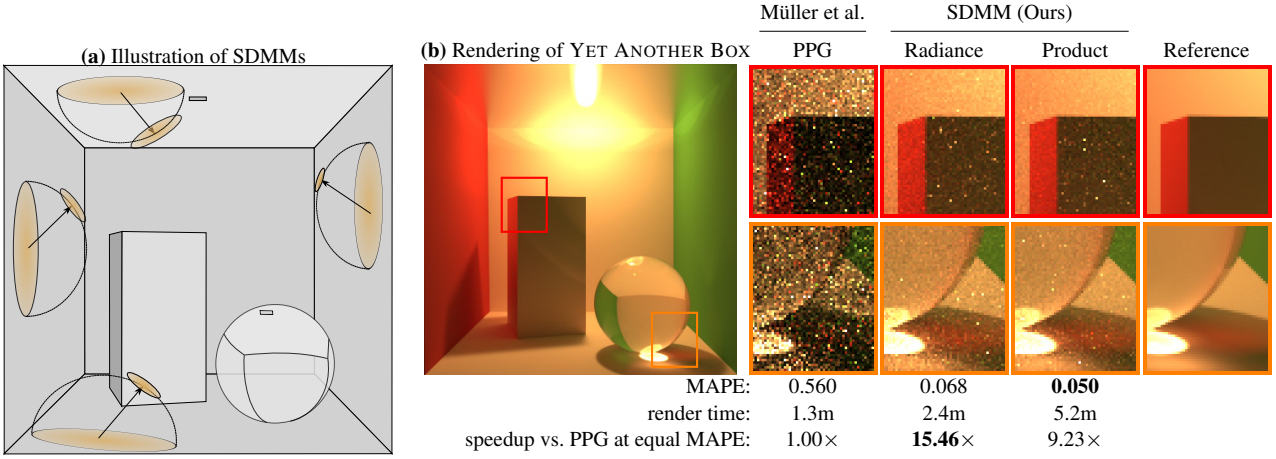


Figure 1: **(a)** Illustration of our 5D spatio-directional mixture model (SDMM) that approximates incident radiance. **(b)** Rendering of a scene with difficult spatio-directionally varying illumination: a tiny light source is shining *upwards* onto a glossy ceiling that indirectly illuminates the scene. We compare path guiding results with and without product sampling using an additional mixture that approximates the BSDF.

Abstract

We propose a learning-based method for light-path construction in path tracing algorithms, which iteratively optimizes and samples from what we refer to as spatio-directional Gaussian mixture models (SDMMs). In particular, we approximate incident radiance as an online-trained 5D mixture that is accelerated by a kD-tree. Using the same framework, we approximate BSDFs as pre-trained nD mixtures, where n is the number of BSDF parameters. Such an approach addresses two major challenges in path-guiding models. First, the 5D radiance representation naturally captures correlation between the spatial and directional dimensions. Such correlations are present in e.g. parallax and caustics. Second, by using a tangent-space parameterization of Gaussians, our spatio-directional mixtures can perform approximate product sampling with arbitrarily oriented BSDFs. Existing models are only able to do this by either foregoing anisotropy of the mixture components or by representing the radiance field in local (normal aligned) coordinates, which both make the radiance field more difficult to learn. An additional benefit of the tangent-space parameterization is that each individual Gaussian is mapped to the solid sphere with low distortion near its center of mass. Our method performs especially well on scenes with small, localized luminaires that induce high spatio-directional correlation in the incident radiance.

CCS Concepts

• ***Computing methodologies*** → ***Ray tracing; Mixture models;*** • ***Mathematics of computing*** → ***Sequential Monte Carlo methods;*** *Expectation maximization; Bayesian computation;*

1. Introduction

Path tracing is used to synthesize photorealistic images in a wide variety of settings, such as movie production, product and architecture visualization, and, more recently, video games. The efficiency of path tracing is a product of two factors: (i) the rate at which paths are traced and (ii) how well the distribution of traced paths matches the actual distribution of light throughout the virtual scene. The *key challenge* is thus to devise an algorithm that accurately samples the distribution of light while at the same time remaining performant. Adaptive importance sampling techniques (usually referred to as "path guiding" in the specific context of rendering) present a promising approach to this problem – instead of relying on heuristics,

†Research completed prior to joining Facebook.

they use information gathered during rendering to learn a sampling distribution which better suits the scene at hand.

Our method belongs to the class of path-guiding techniques which represent factors of the rendering integrand using mixture models. In contrast to prior works which use 2D directional mixture models [VKŠ*14; HEV*16], our method includes additional parameters of the rendering equation as further dimensions of the mixture model. Specifically, we propose representing

- incident radiance (L_i) using a 5D Gaussian mixture model, and
- BSDFs (f_s) using nD Gaussian mixture models,

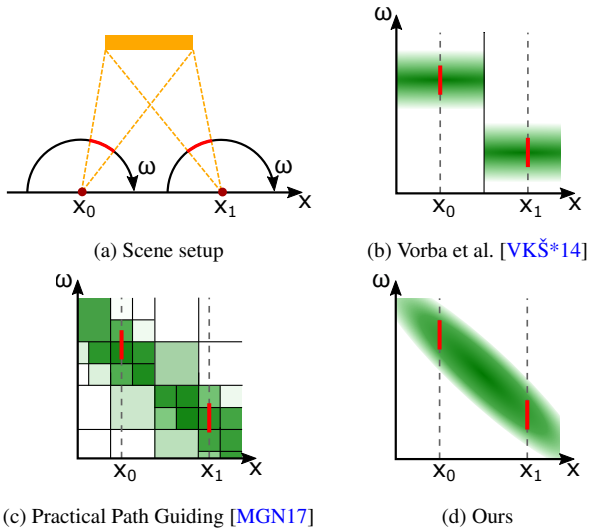
where n is the number of parameters of the BSDF.

The first benefit of such an aproach is that a 5D mixture model naturally captures spatio-directional correlation (e.g. parallax) of the incident radiance $L_\mathrm{i}(\omega_\mathrm{i}, \mathbf{x})$. While recent work focuses on parallax as a specific instance of spatio-directional correlation, it effetively demonstrates the importance of accurately capturing it during path tracing [RHL20]. Additionally, we found nD mixture models to be well-suited for compactly representing parametric BSDFs, as they both vary smoothly as a function of their parameters in most cases [HES*18]. Such a parametric BSDF representation is desirable, because it avoids the need of pre-computing multiple distinct BSDF models for every encountered material configuration. Lastly, after both models have been trained, we can perform closed-form *product sampling*.

In this paper, we describe solutions to overcome three challenges that would otherwise make the use of higher-dimensional mixture models difficult:

Product sampling with mixed-orientation Gaussians. In order to compute the product of Gaussian distributions, they must live in the same coordinates. Herholz et al. [HEV*16] achieve this by parameterizing incident radiance L_i as well as the BSDFs f_s as 2D directional distributions in the shading frame, where the surface normal always points towards $(0,0,1)$. This approach is not possible for us, because L_i is approximated by a global 5D spatio-directional mixture model, where all mixture components must share the same (global) coordinate frame. For product sampling, we must therefore rotate the mixture into the shading frame on-the-fly for every sampling decision. Such a rotation inherently introduces distortion due to the non-linear change of variables induced by changing the 2D parameterization of the solid sphere. To minimize this distortion near the center of mass of each Gaussian, we (i) adopt a tangent-space parameterization [Pen06; STM14] as a substitute for commonly used parameterizations (e.g. cylindrical), and (ii) we propose to transform covariance matrices via a first-order Taylor approximation of the change of variables to further reduce the remaining distortion.

Training of conditional distributions. In path guiding, we are interested in 2D importance-sampling distributions. Thus, when representing incident radiance $L_\mathrm{i}(\omega_\mathrm{i}, \mathbf{x})$ as a 5D distribution $p(\omega_\mathrm{i}, \mathbf{x})$, we ultimately only make use of the 2D *conditioned* distribution $p(\omega_\mathrm{i} \mid \mathbf{x})$. The same holds true for BSDFs. Unfortunately, the optimization of conditional distributions is highly non-linear and difficult [SPK05]. Instead of attacking this difficulty head-on, we side-step it by optimizing our high-dimensional *joint* distributions using the expectation maximization (EM) algorithm and only conditioning on-the-fly for each sampling decision.



(a) Scene setup (b) Vorba et al. [VKŠ*14]

(c) Practical Path Guiding [MGN17] (d) Ours

Figure 2: Comparison of our representation with previous work. In (a) we visualize a 2D incident radiance field with one spatial and one angular dimension (x, ω). The two points x_0 and x_1 are directly illuminated by an area light. (b) to (d) visualize how the incident radiance field is represented as a function of x and ω in previous and our work. In this setup, (b) approximate the angular variation of incident radiance with two 1D GMMs at discrete spatial positions, (c) discretize both dimensions in a 2D adaptive hierarchical data structure and finally, our method in (d) represents the entire spatio-angular domain with a 2D GMM which captures the correlation between the dimensions.

Scaling towards thousands of mixture components. The aforementioned conditioning is the conceptual equivalent to looking up a 2D distribution in a higher-dimensional cache such as done by Vorba et al. [VKŠ*14]. When done with brute force, its cost scales linearly with the number of mixture components. It is therefore an expensive operation that needs acceleration through an appropriate data structure, especially when the high-dimensional mixture consists of thousands of components, which is needed to represent a function as complicated as the incident radiance. We use a continually evolving kD-tree over the conditional dimensions (e.g. the 3 spatial dimensions of incident radiance) to accelerate the on-the-fly conditioning operation to practical speed.

To summarize, the contributions of this paper include

- the use of spatio-directional mixture models for approximating the incident radiance and BSDFs in product path guiding, which is made practical by
- a directional tangent-space parameterization that admits low distortion under rotation, and
- a kD-tree acceleration data structure for efficient on-the-fly conditioning and EM training.

2. Related Work

Existing approaches to importance sampling the rendering integral range from analytic techniques, such as bi-directional path construction [LW93], Markov chain sampling [Vea97], multiple importance sampling [VG95], and BSDF models [HD14; WMLT07], to data-driven techniques, such as path guiding.

Path guiding. "Path guiding" techniques apply principles of adaptive importance sampling by iteratively

1. tracing an initial set of paths,
2. fitting an approximate radiance or product model to these paths,
3. and then using the model to importance sample further paths.

Such techniques were pioneered by Jensen [Jen95], who rasterized a photon map into hemispherical grids for importance sampling, and Lafortune and Willems [LW95], who quantized radiance estimates into a spatio-directional 5D tree that can be used both as a control variate and for importance sampling. More recently, path-guiding techniques surged in popularity after Vorba et al. [VKŠ*14] demonstrated that unidirectional path guiding can match bi-directional techniques in terms of sample quality—a boon to production rendering, which heavily relies on unidirectional tracing. Consequently, many production renderers support path guiding nowadays [VHH*19].

In our work, we adopt several breakthroughs of recent works on path guiding. We perform reinforcement-learning-style iterated rendering [DK18; MGN17], where the rendering iterations are combined with an inverse-variance weighting [Mül19]. Furthermore, we use a spatial acceleration tree structure that adapts to the sample density [HZE*19], and we train Gaussian mixture models to approximate incident radiance [VKŠ*14] within the leaves of that spatial tree. Lastly, we train another Gaussian mixture model to approximate BSDFs and we perform closed-form sampling of the product between the BSDF and the radiance models [HEV*16].

Spatio-directional effects. Initial path-guiding approaches modeled spatio-directional correlations implicitly by subdividing space [LW95; Jen95; VKŠ*14; HP02; DK18; MGN17]. More explicit modeling of such correlations came with primary sample space [ZZ19; MMR*19; GBBE18] and path space approaches [RHJD18], reprojection-based techniques [RHL20] as well as attempts at directly representing the full 5D radiance or 7D product using neural networks [MMR*19; MRNK20]. Our approach is most similar to the latter category—we also explicitly model the 5D radiance field—but rather than using neural networks we use a mixture of 5D Gaussians. It is worth contrasting our data-driven 5D mixtures to the reprojected 2D mixtures of Ruppert et al. [RHL20]. Reprojection constitutes an accurate closed-form solution when its underlying assumptions are valid, i.e. when the perceived origin is near-diffuse and the intermediate transport is spatially linear. This is the case with emitters that are for example directly visible from the shading location or hidden behind flat mirrors or thin sheets of dielectrics. However, when these assumptions are violated, e.g. in caustics caused by curved specular reflectors, the data-driven approach is the more general one (see Figure 12). Admittedly, such cases are rare in practice and do not result in complete failure—additional mixture components can compensate for erroneous reprojection—yet we believe the two approaches can be benefitially combined in the future.

Mixture models. Mixture models have a long and successful history for various uses in computer graphics. Thus, for brevity, we will focus only on work that utilizes mixture models within the context of path guiding. Hey et al. [HP02] were the first to propose a mixture model for path guiding: a mixture of cone-shaped distributions seeded by a photon map. Vorba et al. [VKŠ*14] proposed to use a Gaussian mixture due to its ability to approximate sparse radiance estimates using the expectation maximization (EM) optimization algorithm. They used an appproach similar to the general adaptive importance sampling algorithm called *mixture population Monte Carlo*, by Cappé et al. [CDG*08]. Later approaches exploited the closed-form product of Gaussian mixtures to perform product sampling with the BSDF, where the BSDF is approximated by an auxiliary Gaussian [HEV*16] or skewed Gaussian [HES*18] mixture. Alternatively, mutivariate Gaussians can also parameterize the distribution of entire paths [RHJD18], elegantly capturing spatio-directional correlations. However, the high-dimensional space of paths is difficult to cover densely, constraining the approach to those paths whose interactions are near-specular and thereby have a low *effective* dimensionality.

Similarly to Gaussian mixtures, mixtures of von Mises-Fisher distributions can also represent incident radiance, the BSDF (or phase function), and their product [HZE*19; RHL20]. Von Mises-Fisher distributions have the key advantage that their domain is the solid sphere $\mathbb{S}^2$ rather than Euclidean space, enabling their use without a spherical parameterization that could introduce unwanted distortion. For path guiding, an additional crucial benefit of living in $\mathbb{S}^2$ is that they can be trivially rotated, permitting product sampling between a world-space radiance representation and a local-space BSDF (or phase function) representation.

Our method seeks the same property for product sampling: the ability to rotate the mixture model freely in $\mathbb{S}^2$. However, we also desire the ability to model anisotropy with few mixture components as well as the ability to condition on additional Euclidean dimensions, e.g. the 3 spatial dimensions in the 5D spatio-directional radiance distribution. These requirements disqualify the isotropic von Mises-Fisher distributions, as well as other possible spherical candidates, such as the Bingham distribution [Bin74] (undesirable symmetry) or anisotropic spherical Gaussians [XSD*13] (difficult conditioning). Instead, we rely on a *tangent-space* Gaussian mixture model [Pen06; STM14], which we will describe in detail in Section 3.1.

3. Methodology

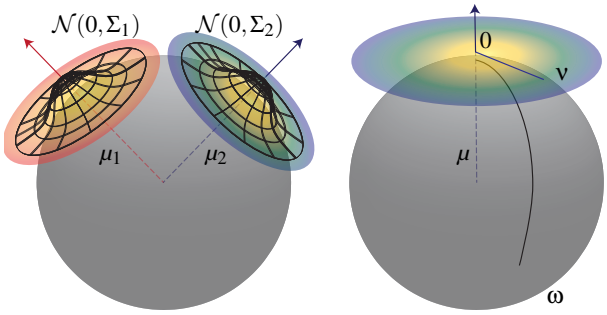
We are interested in using Monte Carlo integration to solve the reflection integral

$$L_s(\omega_o, \mathbf{x}) = \int_{\mathbb{S}^2} L_i(\omega_i, \mathbf{x})\, f_s(\omega_i, \omega_o, \phi(\mathbf{x}))\cos(\gamma)\, d\omega_i\,, \qquad (1)$$

where ω_o is the reflected direction, ω_i is the direction of incident radiance, γ is its angle with the surface normal, $\mathbf{x}$ is the shading location, and $\phi(\mathbf{x})$ are the parameters of the BSDF at that location.

In order to improve the efficiency of said Monte Carlo integration, we would like to importance sample according to a probability density function (PDF) that is approximately proportional to the integrand, i.e. our goal is to find

$$p(\omega_i \,|\, \omega_o, \mathbf{x}) \propto L_i(\omega_i, \mathbf{x})\, f_s(\omega_i, \omega_o, \phi(\mathbf{x}))\cos(\gamma)\,. \qquad (2)$$



(a) Mixture setup (b) Tangent space parameterization

Figure 3: **(a)** A 2-component example of a tangent-space Gaussian mixture model. Each Gaussian component is parameterized by a unit-length mean vector μ_k, pointing to its position on the sphere, and a covariance matrix Σ_k describing the Gaussian's shape in the corresponding *tangent space* of the sphere. **(b)** The μ-centered tangent space is a circular 2D parameterization of the surface of the sphere. World-space directions ω are transformed to tangent-space directions ν and back via $\nu = \log_\mu(\omega)$ and $\omega = \exp_\mu(\nu)$.

We approach this goal by decomposing it into two sub-goals: (i) learning the incident radiance L_i as a 5D Gaussian mixture model and (ii) learning the BSDFs f_s as nD Gaussian mixture models, where n is the total number of dimensions of ω_i, ω_o, and ϕ:

$$p_{L_i}(\omega_i, \mathbf{x}) \propto L_i(\omega_i, \mathbf{x}) \tag{3}$$

$$p_{f_s}(\omega_i, \omega_o, \phi) \propto f_s(\omega_i, \omega_o, \phi(\mathbf{x}))\cos(\gamma)\,. \tag{4}$$

We then approximately sample according to Equation (2) by conditioning the approximations p_{L_i} and p_{f_s} on $(\omega_o, \mathbf{x}, \phi(\mathbf{x}))$ and then computing their closed-form product distribution, i.e.

$$p(\omega_i \,|\, \omega_o, \mathbf{x}) \propto p_{L_i}(\omega_i \,|\, \mathbf{x})\, p_{f_s}(\omega_i \,|\, \omega_o, \phi(\mathbf{x}))\,. \tag{5}$$

Next, we will discuss each of these steps in detail.

3.1. Tangent-Space Gaussian Mixtures

Since we represent incident radiance using a spatio-directional 5D mixture model as opposed to caches that are associated with geometric surfaces, we are forced to adopt a global "world-space" parameterization of the incident radiance. In order to compute product sampling between world-space radiance and a locally parameterized BSDF, we must be able to rotate the mixtures to the same coordinate frame. To this end, we build upon spherical *tangent-space* Gaussian mixture models [Pen06; STM14]. For example, in a tangent-space model on a sphere with K mixture components, the k-th Gaussian is parameterized by a 3D world-space unit-length mean vector $\mu_k \in \mathbb{S}^2$ and a 2×2 tangent-space covariance matrix $\Sigma_k \in \mathbb{R}^{2\times2}$. The mean vector μ_k *determines* the tangent space that the covariance matrix Σ_k lives in; see Figure 3a for an illustration.

Crucially, this representation can be rotated into any local shading frame for product sampling: one simply rotates the mean vectors μ_k to the local frame and applies the azimuthal part of the rotation to the covariance matrices Σ_k.

Tangent spaces. We define the tangent space at some mean vector μ, as a circular 2D parameterization of the surface of the sphere (see Figure 3b): one can map from the μ-centered tangent space to the sphere and back, using the so-called log and exp maps, which are each other's inverse:

$$\nu = \log_\mu(\omega)\,, \quad \omega = \exp_\mu(\nu)\,. \tag{6}$$

In the following we will assume that $\omega \in \mathbb{S}^2$ is a direction vector and $\nu \in \mathbb{R}^2$ is a coordinate in a tangent space with $\|\nu\| < \pi$. In this case, we can use the *azimuthal equidistant projection* to define our log and exp maps, same as Simo-Serra et al. [STM14]:

$$\log_\mu(\omega) = \left(\frac{\omega_x^{\circlearrowleft}}{\operatorname{sinc}(\cos^{-1}(\omega_z^{\circlearrowleft}))}\,, \frac{\omega_y^{\circlearrowleft}}{\operatorname{sinc}(\cos^{-1}(\omega_z^{\circlearrowleft}))} \right)^\intercal, \qquad \omega^{\circlearrowleft} = R_\mu \omega\,, \tag{7}$$

$$\exp_\mu(\nu) = R_\mu^{-1}\left(\nu_u \operatorname{sinc}(\|\nu\|), \nu_v \operatorname{sinc}(\|\nu\|), \cos(\|\nu\|)\right)^\intercal, \tag{8}$$

where R_μ is the rotation matrix that rotates μ to $(0,0,1)$ and sinc is the *unnormalized* sinc function.

The crucial property that justifies the use of exponential and logarithm maps is that the length of the shortest path (i.e. shortest geodesic) connecting μ with any ω along the manifold (e.g. sphere) equals the distance between the origin of the tangent space at μ and the corresponding ν [†]. This property enables the optimization of tangent-space Gaussians using the expectation maximization (EM) algorithm [STM14], which we will describe in Section 3.2.

Density evaluation. Given the parameters μ and Σ of a spherical tangent-space Gaussian, its density *in tangent space* is the regular bi-variate Gaussian density, centered at the tangent space's origin:

$$p^t(\nu) := e^{-\frac{1}{2}\nu^\intercal \Sigma^{-1} \nu} \Big/ \left(2\pi\sqrt{\det\Sigma}\right). \tag{9}$$

Its density with respect to the *solid-angle measure*, which we need for importance sampling of directions in path tracing, is in turn obtained through a change of variables:

$$p^\Omega(\omega) = p^t(\nu)\sqrt{|\det G|}\,, \tag{10}$$

where G is the matrix representation of the metric tensor of the tangent space. Specifically, it is the 2×2 matrix $G = J^\intercal J$, where J is the 3×2 Jacobian matrix of the the exponential map, $\exp_\mu$, evaluated at ν. [‡] We provide the closed form of the Jacobian matrix in Appendix A.

Thus, the solid-angle density of a *mixture* of K tangent-space Gaussians is

$$\boxed{p(\omega) := \sum_{k=1}^{K} \pi_k\, p_k^\Omega(\omega)\,,} \tag{11}$$

† Note that, for spheres, this property holds as long as $\|\nu\| < \pi$, or, more generally, ν does not cross the *cut locus* at μ

‡ The use of the metric tensor in a change of variables generalizes the well-known use of the absolute determinant of the Jacobian matrix to the case of transforming between manifolds; in our case, we are mapping from a 2D tangent space to 3D unit vectors.

where π_k is the weight of the k-th mixture component, such that $\sum_k \pi_k = 1$, and $p_k^\Omega(\omega)$ is its solid-angle density.

Sampling. Sampling of a tangent-space mixture is simple compared to evaluating its density. It consists of three steps: (i) sample the mixture component index k proportional to π_k, then (ii) sample the tangent-space direction $\nu \propto p_k^t(\nu)$, and lastly (iii) convert the sampled ν to world space by evaluating $\omega = \exp_{\mu_k}(\nu)$.

Special care must be taken in step (ii), since the bijectivity of the azimuthal equidistant projection breaks down at the radius π: due to the equidistance property, the projection reaches the antipodal point of μ_k at the radius π. Therefore, for the purpose of Monte Carlo integration using the density defined in Equation (11), if the sampled ν lies outside the radius π, the sample must be discarded.

Note that all bounded parameterizations of the sphere (e.g. cylindrical coordinates) require discarding samples that lie outisde of their domain. In fact, the tangent space formulation has an advantage in this regard, because the center of mass of each Gaussian is located in the center of its tangent space, leading to only a vanishingly small number of discarded samples in practice (on average 0.021%).

Additional dimensions. When we want a tangent-space Gaussian mixture to admit additional, Euclidean dimensions, e.g. the position $\mathbf{x}$ when approximating incident radiance as $p_{L_i}(\omega_i, \mathbf{x}) \propto L_i(\omega_i, \mathbf{x})$, the above equations stay almost exactly the same.

In this case, Equation (9) becomes the general multi-variate Gaussian distribution, and the exponential and logarithm maps of the Euclidian dimensions are defined as *translations* by their mean vector. For example, the 3D world-space position is mapped to its tangent space as $\log_\mu(\mathbf{x}) := \mathbf{x} - \mu^{\mathbf{x}}$, where $\mu^{\mathbf{x}}$ is the mean of the Gaussian distribution in 3D world space. By representing Euclidean dimensions in their own tangent space, all dimensions—regardless of whether they are spherical or Euclidean—can be treated using the same tangent-space formulae.

3.2. Learning of Conditional Mixture Models using EM

We are ultimately interested in obtaining the conditional 2D importance-sampling distributions $p_{L_i}(\omega_i \,|\, \mathbf{x})$ and $p_{f_s}(\omega_i \,|\, \omega_o, \phi)$. However, directly optimizing the conditional distributions turns out to be difficult [SPK05]. Therefore, we instead optimize the corresponding joint distributions $p_{L_i}(\omega_i, \mathbf{x})$ and $p_{f_s}(\omega_i, \omega_o, \phi)$, and subsequently condition them on-the-fly during rendering using standard closed-form Gaussian conditioning. To optimize the joint distributions, we utilize the well established expectation maximization (EM) algorithm. The following paragraphs briefly introduce EM and then focus on our adaptations to make it compatible with tangent-space Gaussian mixtures.

Mini-batch expectation maximization. We use the mini-batch variant of EM [Dod20; NFM20] due to its online training capability[§]. Mini-batch EM operates by repeatedly sampling a mini-batch of M samples from a Gaussian mixture parameterized by (π_k, μ_k, Σ_k) and then computing from that mini-batch a triplet of sufficient statistics $(S_k^{(0)}, S_k^{(1)}, S_k^{(2)})$ for each mixture component k. From these sufficient statistics we then compute the updated mixture parameters, $(\hat{\pi}_k, \hat{\mu}_k, \hat{\Sigma}_k)$.

Given a mini-batch of samples $x_1, \ldots, x_M \in \mathbb{R}^D$ with corresponding Monte Carlo weights $w_1, \ldots, w_M \in \mathbb{R}$, the sufficient statistics are computed as weighted sums

$$S_k^{(0)} = \frac{\sum_i^M w_i \mathcal{S}_{ki}}{\sum_i^M w_i}, \quad S_k^{(1)} = \frac{\sum_i^M w_i \mathcal{S}_{ki} x_i}{\sum_i^M w_i}, \quad S_k^{(2)} = \frac{\sum_i^M w_i \mathcal{S}_{ki} x_i x_i^\mathsf{T}}{\sum_i^M w_i},$$
$$\mathcal{S}_{ki} := \frac{\pi_k p_k^t(x_i)}{p^t(x_i)}, \tag{12}$$

and, thereafter, the updated parameters $(\hat{\pi}_k, \hat{\mu}_k, \hat{\Sigma}_k)$ are computed as

$$\hat{\pi}_k = \frac{S_k^{(0)}}{\sum_i^K S_i^{(0)}}, \quad \hat{\mu}_k = \frac{S_k^{(1)}}{S_k^{(0)}}, \quad \hat{\Sigma}_k = \frac{S_k^{(2)}}{S_k^{(0)}} - \hat{\mu}_k \hat{\mu}_k^\mathsf{T}. \tag{13}$$

Unfortunately, a derivation of the above equations is beyond the scope of this paper. We refer to detailed descriptions of mini-batch EM based on sufficient statistics [Dod20; NFM20].

Sufficient statistics in tangent spaces. When the mixture components are defined in tangent space, special care must be taken when computing the first- and second-moment sufficient statistics $S_k^{(1)}$ and $S_k^{(2)}$. Given a mini-batch of *direction vectors* $\omega_i \in \mathbb{S}^2, i \in \{1, \ldots, M\}$, the first-moment sufficient statistic $S_k^{(1)}$ of the k-th component must be computed in the corresponding μ_k-centered tangent-space coordinates $\nu_i = \log_{\mu_k}(\omega_i)$ and, subsequently, the updated mean vector $\hat{\mu}_k$ must be computed by translating back from the μ_k-centered tangent-space to world space:

$$S_k^{(1)} = \frac{\sum_i^M w_i \mathcal{S}_{ki} \nu_i}{\sum_i^M w_i}, \quad \hat{\mu}_k = \exp_{\mu_k}\left(S_k^{(1)} / S_k^{(0)}\right). \tag{14}$$

Only after the updated mean vector $\hat{\mu}_k$ has been computed can the second-moment sufficient statistic $S_k^{(2)}$ be computed. This is the case, because the statistic $S_k^{(2)}$ determines the updated covariance matrix $\hat{\Sigma}_k$, which must be centered around $\hat{\mu}_k$, not μ_k. Formally:

$$S_k^{(2)} = \frac{\sum_i^M w_i \mathcal{S}_{ki} \hat{\nu}_i \hat{\nu}_i^\mathsf{T}}{\sum_i^M w_i}, \quad \hat{\Sigma}_k = S_k^{(2)} / S_k^{(0)} \tag{15}$$

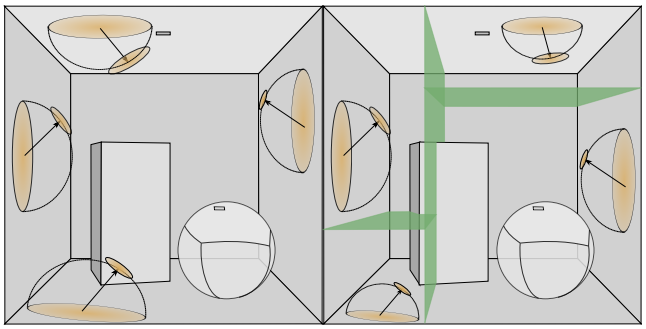
where $\hat{\nu}_i = \log_{\hat{\mu}_k}(\omega_i)$ and $\mathcal{S}_{ki}$ must use updated densities w.r.t. to the $\hat{\mu}_k$-centered tangent space. Note, that the subtraction of $\hat{\mu}_k \hat{\mu}_k^\mathsf{T}$ is missing (c.f. Equation (13)), because $S_k^{(2)}$ was computed in the already $\hat{\mu}_k$-centered tangent space.

As previously stated in Section 3.1, additional Euclidean dimensions, such as the 3D position $\mathbf{x}$, can also be treated with the new formulae if the log and exp maps map them to their respective tangent space by translating them by their mean vector.

3.3. Efficient Conditioning and EM

After the joint distribution $p_{L_i}(\omega_i, \mathbf{x})$ has been optimized using EM, we seek to efficiently condition it on $\mathbf{x}$ on-the-fly during rendering to be able to importance sample $p_{L_i}(\omega_i \,|\, \mathbf{x})$. However, the computational cost of conditioning is linear in the number of mixture

[§] We discuss an alternative option, stepwise EM [CM09; Cap11; VKŠ*14], in Section 6.



(a) Single mixture (b) Mixture partitioned by kD-tree

Figure 4: Illustrations of the spatial components of our model. In (a) we visualize a small fraction of the mixture components within a single model and their spatial overlap. Due to the high amount of overlap, the computational requirements of updating and querying the model can be linear to the total number of components. By using a kD-tree spatial subdivision scheme (b) we can significantly reduce these computational requirements while still capturing correlations within a leaf node.

components, and our 5D Gaussian mixtures require *thousands* of components to cover the incident radiance field well. Thus, when done naïvely for *all* thousands of components, conditioning is exceedingly expensive.

The key observation that makes conditioning tractable is that only those mixture components whose spatial mean component $\mu_k^{\mathbf{x}}$ is in close proximity to $\mathbf{x}$ meaningfully affect the conditional distribution $p_{L_i}(\omega_i \,|\, \mathbf{x})$. Conditioning can thus be accelerated by only operating on a local neighborhood of mixture components.

The EM algorithm, like conditioning, also suffers from an $\mathcal{O}(K)$ cost and can therefore also be accelerated by operating independently on local neighborhoods of Gaussians. At first, it may seem like restricting EM to local neighborhoods might destroy the desirable properties of spatio-directional mixtures, but this is a largely incorrect notion. It merely prevents individual mixture components from covering large spatial regions, which is rarely appropriate in high-fidelity virtual scenes. Most importantly, it does *not* remove the key ability of spatio-directional mixtures to accurately model spatio-directional correlations such as parallax *within* their neighborhoods.

To partition our 5D Gaussian mixture components into local neighborhoods, we adopt the spatial kD-tree partitioning scheme proposed by Herholz et al. [HZE*19]. We subdivide and collapse the kD-tree according to the number of observed samples within each leaf—same as Herholz et al.—and *within* each leaf, we place 16 of our 5D mixture components, as illustrated in Figure 4b. Given a query position $\mathbf{x}$, conditioning is thus only performed on the 16 Gaussians found within the kD-tree's leaf node at $\mathbf{x}$. Likewise, given a mini-batch of spatio-directional samples for EM training, the samples are partitioned into their respective kD-tree leaves and mini-batch EM is applied to each leaf independently.

In Figure 5 we demonstrate that using the kD-tree yields the desired performance benefits without compromising on the quality benefits of the spatio-directional radiance representation.

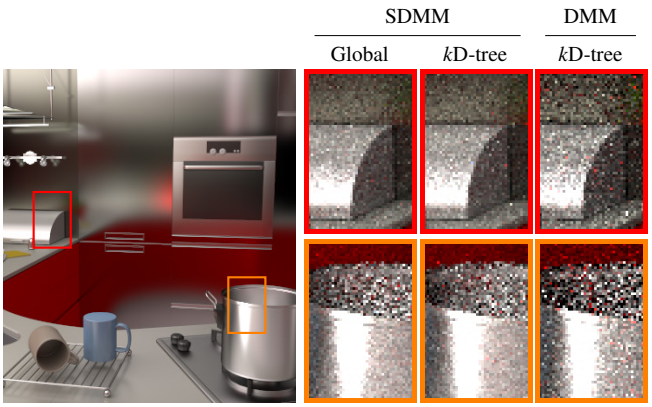


Figure 5: Subdividing our spatio-directional mixture model by a kD-tree (middle) unlocks similarly low noise as using a single global mixture (left) at a fraction of the cost—one that is comparable to placing purely directional mixtures in the same kD-tree (right). All mixture models use a total of roughly 16000 components and were rendered with 1024spp. Additional comparisons of spatio-directional versus directional mixtures are available in the supplementary material.

3.4. Product Sampling from Mixed-Orientation Gaussians

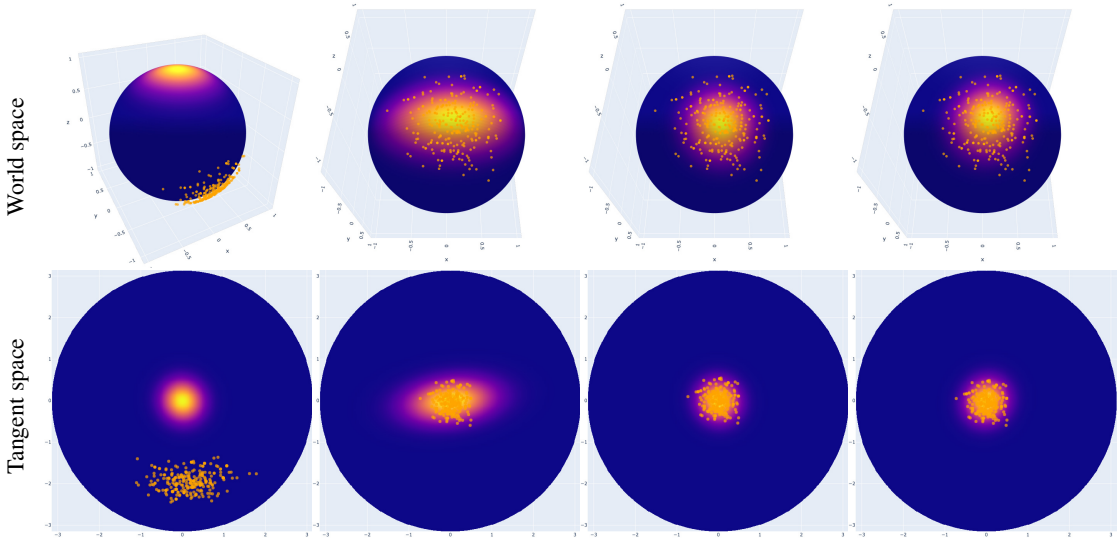
After conditioning $p_{L_i}(\omega_i \,|\, \mathbf{x})$ and $p_{f_s}(\omega_i \,|\, \omega_o, \phi)$, the last step is computing their product distribution such that it can be importance sampled (for the details of how p_{f_s} is conditioned in practice, please see Section 4.1).

To do so, we must first rotate both mixtures into the same coordinate frame, which is achieved by applying the appropriate rotation matrix to the mean vectors μ_k of either the incident-radiance or the BSDF distribution. In the following, we will assume that such a rotation has already been performed and the two distributions are in the same directional coordinate frame.

To compute the product distribution of the radiance approximation $p_{L_i}(\omega_i \,|\, \mathbf{x})$ and the BSDF approximation $p_{f_s}(\omega_i \,|\, \omega_o, \phi)$, one has to compute the product between *each pair* of mixture components. Let us consider a single such pair, where one component is parameterized by (μ_1, Σ_1) and the other component is parameterized by (μ_2, Σ_2).

To compute the product of these two components, they must be expressed within the same tangent space. Without loss of generality, we choose to parameterize the second component in the μ_1-centered tangent space.

Mapping covariances to other tangent spaces. In the μ_1-centered tangent space, the second mean vector μ_2 is easily expressed using the exp-map as $\exp_{\mu_1}(\mu_2)$, but the question of how Σ_2 should transform from one tangent space to the other is more difficult to answer. The most naïve approach would be to simply leave it unchanged, which we found to be too inaccurate; see Figure 6b. The correct approach, on the other hand, would be to re-fit the covariance matrix through EM in the new tangent space, which would be impractically slow to do for each sampling decision.



(a) Before EM (b) After EM (naïve Σ) (c) After EM (accurate Σ) (d) After EM (first-order Σ)

Figure 6: Various approaches to updating the covariance matrix Σ of a tangent-space Gaussian distribution with mean μ to a mini-batch of samples. We visualize the Gaussian distribution as well as the mini-batch in world space (top row) and in the μ-centered tangent space (bottom row). **(a)** The state of the Gaussian distribution prior to the EM step. **(b)** When Σ is naïvely updated using the regular EM equations (12) and (13), it admits an incorrect shape due to the distortion of mapping from the tangent space centered around the old mean μ to the one centered around the new mean $\hat{\mu}$. **(c)** Updating Σ in the tangent space centered around the new mean $\hat{\mu}$ as per Equation (15) results in an accurate fit but is computationally costly, because the mini-batch must be transformed into the $\hat{\mu}$-centered tangent space first. **(d)** Updating Σ using the regular EM equations (same as **(b)**) and subsequently transforming it into the $\hat{\mu}$-centered tangent space using our first-order approximation from Equation (17) results in an accurate fit with much lower cost.

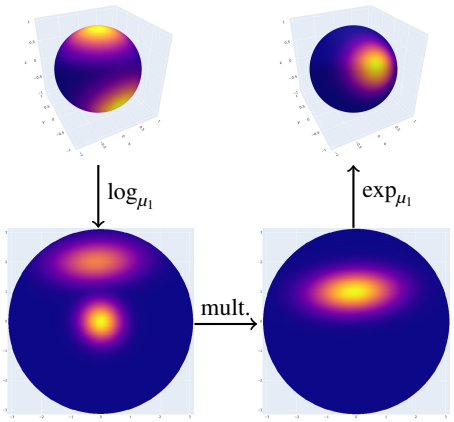


Figure 7: The product of two tangent-space Gaussians that are parameterized by (μ_1, Σ_1) and (μ_2, Σ_2) is computed in the μ_1-centered tangent space according to Equation (17). Then, samples for path guiding can be drawn and mapped to world-space by the $\exp_{\mu_1}$-map.

We aim for a middle ground by considering the $\mathbb{R}^2 \mapsto \mathbb{R}^2$ mapping from one tangent space to the other:

$$v_1 = \log_{\mu_1} \exp_{\mu_2}(v_2) \tag{16}$$

If this mapping was *linear*, it could be represented by a 2×2 matrix J and the correct transformation of the covariance matrix could be computed efficiently as $J\Sigma_2 J^\mathsf{T}$. However, the mapping is not actually linear, so we propose setting J to a *local linear approximation*—the Jacobian matrix—of $\log_{\mu_1} \exp_{\mu_2}$. We demonstrate the accuracy of this approach in Figure 6d, where we compute J as $J := J_{\log_{\mu_1}}(\exp_{\mu_2}(0)) J_{\exp_{\mu_2}}(0)$ with $J_{\log}$ and $J_{\exp}$ being defined in Appendix A.

Thus, the product distribution of a pair of components defined by (μ_1, Σ_1) and (μ_2, Σ_2) can be approximated in the μ_1-centered tangent space as the product of the following two Gaussian distributions:

$$\boxed{\mathcal{N}(0, \Sigma_1) \otimes \mathcal{N}\left(\log_{\mu_1}(\mu_2), J\Sigma_2 J^\mathsf{T}\right),} \tag{17}$$

Figure 7 illustrates this procedure. Here, it is worth pointing out that the approximation error of our proposed covariance transformation $J\Sigma_2 J^\mathsf{T}$ only applies to the *second* mixture component. Thus, in practice, however small the approximation error may be, one could choose the second mixture component to be the one where the error is most tolerable. Alternatively, one might consider the computational cost of computing $J\Sigma_2 J^\mathsf{T}$. In our case, due to our usage of SIMD vectorization, it is beneficial to choose the second mixture to be the one with fewer components, enabling us to vectorize over the components of the first mixture. We obtained the best overall efficiency by setting the first mixture to the incident radiance and the second mixture to the BSDF.

Efficient tangent-space covariance update. The first-order approximation that enables accurate product sampling can also be used improve performance: in Section 3.2, we state that the updated

covariance matrix $\hat{\Sigma}$ must be computed in the tangent space centered around the newly computed mean $\hat{\mu}$ as opposed to the previous mean μ; see Equation (15). This computation is expensive, because every data point in the mini batch as well as its corresponding probability density must be mapped into the $\hat{\mu}$-centered tangent space. We side-step this expensive computation by computing $\hat{\Sigma}$ in the original μ-centered tangent space and then approximately transforming it into the $\hat{\mu}$-centered tangent space using the same first-order approximation that we use in Equation (17) for product sampling. In Figure 6, we visually demonstrate the accuracy of this approach.

4. Implementation in a Renderer

As mentioned in Section 3, our algorithm requires us to train both a BSDF model $p_{f_s}(\omega_i, \omega_o, \phi)$ as well a model of the incident radiance $p_{L_i}(\omega_i, \mathbf{x})$. In this section, we detail these procedures.

4.1. BSDF Learning

Our approach to BSDF learning allows us, in theory, to fit a single nD mixture model to each type of BSDF that the renderer supports, where n is the number of BSDF parameters. For each sampling decision we can then specialize the previously learned general BSDF models by conditioning them on-the-fly on their parameters $\phi(\mathbf{x})$ at the shading location $\mathbf{x}$. This way, we avoid a scene-specific precomputation of BSDFs, as well as opaquely handle both spatially-uniform and spatially-varying BSDFs.

In practice, to keep the number of dimensions manageable, we limit ourselves to isotropic BSDFs. Our model has not been tested on anisotropic BSDFs. It might be necessary to increase the number of Gaussian components to represent such models, which would be computationally prohibitive for our current approach. One could use a kD-tree over the BSDF parameter space, similar to the one we use over the spatial dimensions of incident radiance, or an alternative pruning framework, such as that of Herholz et al. [HES*18]; see the discussion in Section 6 for more details.

BSDF input sampling. A BSDF mixture model is fitted by repeatedly sampling batches of BSDF inputs and then applying mini-batch EM to the mixture model. Random BSDF parameters ϕ as well as the spherical coordinates of the outgoing direction ω_o are sampled uniformly. Subsequently, ω_i is sampled using the BSDF's built-in importance sampling routine and mini-batch EM is applied.

Pruning before product sampling. Since product sampling requires computing the product distribution between *each pair* of mixture components within the BSDF and the incident-radiance model, the computational cost of product sampling can grow quadratically in the number of overall mixture components. To suppress this quadratic growth, we employ a simple but effective strategy: after conditioning the BSDF model on its parameters, we perform product sampling using *only* the two resulting BSDF mixture components with *largest* magnitude.

4.2. Online Learning of the Radiance During Path Tracing

Path tracing of an N-spp image is performed in a fixed number of iterations that compute 4 spp each. The i-th iteration is importance sampled (i.e. path guided) using the product of the current state $p^i_{L_i}$ of the incident-radiance model and the pre-computed BSDF distribution p_{f_s}, each conditioned on-the-fly on the local shading parameters (see Sections 3.3 and 3.4). The purpose of dividing path tracing into iterations is to facilitate periodic refinement of the incident-radiance model: after each iteration, the kD-tree is adapted to the path distribution and a subsequent mini-batch EM step is performed to produce a better distribution $p^{i+1}_{L_i}$ for importance sampling in the next iteration.

Spatial component of the radiance distribution. Our 5D SD-MMs should learn $p_{L_i}(\omega_i, \mathbf{x}) = p_{L_i}(\omega_i \,|\, \mathbf{x}) p_{L_i}(\mathbf{x})$. However, during rendering, we only have access to $p_{L_i}(\omega_i \,|\, \mathbf{x})$ in closed form. The spatial distribution of samples, $p_{L_i}(\mathbf{x})$, is defined by the ray-tracing procedure and is not known in a general setting. While it would be possible to approximate the spatial distribution using a k-nearest-neighbor estimate, and then use that estimate to approximate the correct Monte Carlo weights in Eq. 12 [Dod20], we have found the computational overhead of this approach to overshadow any possible gains due the quality of the learned distribution. Instead, in our work, we assume a uniform distribution within each kD-tree leaf node. In practice, this assumption becomes increasingly correct as the size of the kD-tree leaf nodes decreases during training.

Refinement of the spatial kD-tree. During each iteration, we collect the encountered path vertices such that at the end of the iteration, each kD-tree leaf contains a list of all vertices that were located within it. If the *number* of path vertices M in any leaf is larger than a threshold c, that leaf is subdivided along the axis with the largest variation of vertex positions, where the subdivision plane is located at the mean position along that axis [HZE*19]. After subdivision, the vertices are redistributed into the two new leaves, which are then further subdivided recursively until they have fewer than c vertices. The mixture model from the parent node is *copied* into the newly created leaves—the subsequent EM procedure will adapt each copy to the different light distributions in each leaf. We empirically found $c = 16000$ to yield good results, which is of similar magnitude as the subdivision constant used in PPG [MGN17].

Mini-batch EM fitting. Before the next iteration starts, we select all leaves that accumulated $M \geq 16$ vertices—for numeric stability—and independently apply a *single* mini-batch EM step to each of them, where the mini-batch consists of all M vertices contained in the respective leaf node. The vertices are subsequently cleared. The EM step follows the procedure outlined in Section 3.2, where each vertex's incident-radiance estimate corresponds to its weights w, and its spatial and directional coordinates correspond to Euclidean and tangent-space dimensions, respectively.

Note, that the EM step can be computed in parallel across all leaf nodes, because the mixtures are independent from each other.

Early stopping of training. Our incident-radiance mixture model converges to a good guiding distribution relatively early in the rendering process. Thus, we stop the mini-batch EM training after the first $1/4$th of the rendering budget is exhausted to avoid its cost. During the last $3/4$ths of the rendering budget, our mixture model is no longer trained and merely used for guiding.

4.3. Robust EM Optimization of the Incident-Radiance PDF

While the EM algorithm is theoretically elegant and converges quickly, it is prone to get stuck in undesirable local minima, making it unstable in our online-learning setting. The instability becomes worse when operating on noisy Monte Carlo weights, which are common in rendering when estimating incident radiance (e.g. "fireflies"). We therefore found it paramount to employ the following array of regularization techniques to robustify the EM training of our incident radiance approximation.

Initialization. We aim at initializing the Gaussian mixture such that it evenly covers the 5D domain (no "clumping") to prevent EM from converging to a local optimum that misses a mode of L_i. To this end, and to avoid lock contention, prior to rendering, the kD-tree is pre-subdivided 3 times at the center of each axis, resulting in a regular $8\times8\times8$ tesselation of the scene's bounding box. After each render iteration, the we split the kD-tree and distribute the collected samples to the newly created leaf nodes. If a leaf node is uninitialized and contains at least 16 path vertices, we initialize a mixture model containing 16 Gaussians inside of that leaf node. If a leaf node contains fewer than 16 samples, the initialization of the corresponding mixture model is postponed until 16 samples were received to ensure stability in noisy scenes.

After 16 vertices were received, initialization consists of the following three steps. First, a slightly modified `k-means++` [AV07] scheme is applied to the vertices to select 2 spatial coordinates $\mathbf{x}_1, \mathbf{x}_2$ that lie on a geometric surface, have a minimum distance from each other (if possible), and that roughly follow the spatial distribution of radiance; see Appendix C for details. Then, at each one of the 2 chosen positions, 8 Gaussians are initialized whose directional mean components $\omega_1, \ldots, \omega_8$ cover the sphere roughly evenly. Formally, the set of initial mean vectors is $\{\mathbf{x}_1, \mathbf{x}_2\} \times \{\omega_1, \ldots, \omega_8\}$.

Third, the covariance matrix at each mean vector $(\mathbf{x}, \omega)$ is initialized to satisfy the following desired properties. It should be

- isotropic in the tangent plane of the surface at $\mathbf{x}$ and in the ω-centered directional tangent space, and
- flat along the surface normal at $\mathbf{x}$.

To this end, in the tangent plane, the covariance matrix's radius is set to be proportional to the size of the leaf node containing it, and inversely proportional to the number of distinct spatial coordinates in the mixture (in our case 2). In contrast, we make the covariance matrix as thin as possible along the direction of the surface normal, to avoid mixing samples from close-by surfaces facing each other. In the remaining ω-centered tangent space dimensions, the covariance matrix's radius is inversely proportional to the number of distinct directional coordinates in the mixture model, i.e. 8. See Appendix B for the mathematical details of this procedure.

Averaging of sufficient statistics across minibatches. In Section 3.2, we described the computation of a per-mixture-component triplet of sufficient statistics $(S^{(0)}, S^{(1)}, S^{(2)})$ as sample averages over mini batches. In practice, such sample averages are often too noisy to be directly useful for EM optimization. Thus, similar to previous work [CM09; Cap11; VKŠ*14], we additionally average the sufficient statistics *across* mini batches using the Robbins-Monroe algorithm [RM51], which ensures that EM converges under mild assumptions[¶]. Let $\hat{S}^{(i)}_{k,j}$ be the averaged sufficient statistics of the k^{th} mixture component after processing the j-th mini batch; they are computed as

$$\hat{S}^{(i)}_{k,j} = (1-\eta_j)\cdot\hat{S}^{(i)}_{k,j-1} + \eta_j \cdot S^{(i)}_k\,, \tag{18}$$

where η_j controls the strength of averaging. While any sequence η_j that satisfies $\sum_j^\infty \eta_j = \infty \wedge \sum_j^\infty \eta_j^2 < \infty$ makes EM converge in theory, we adopt the sequence $\eta_j = (\beta j + 1)^{-\alpha}$ inspired by common practice [CM09; Cap11; VKŠ*14; NFM20]. We set $\alpha = 0.5$ as done in previous work, $\beta = 0.1$ to achieve a long tail (averaging over many iterations to reduce noise).

Special care must be taken when averaging the second-moment sufficient statistics in tangent spaces. The second-moment statistic of the current mini batch $S^{(2)}$ is defined in the tangent space centered about the *current* mean vector μ_j, whereas the previously computed statistic $\hat{S}^{(2)}_{j-1}$ is defined in the tangent space centered about the *previous* mean-vector μ_{j-1}. Thus, prior to averaging, $\hat{S}^{(i)}_{j-1}$ must first be transformed into the μ_j-centered tangent space. To this end, we use the same first-order approximation that we use in Equation (17), resulting in the following modified update rule for the tangent-space components of the second-moment sufficient statistics:

$$\hat{S}^{(2)}_j = \eta_j \cdot J\hat{S}^{(2)}_{j-1}J^\intercal + (1-\eta_j)\cdot S^{(2)}\,. \tag{19}$$

In the above equation, J is the Jacobian matrix of $\log_{\mu_{j-1}}\exp_{\mu_j}$ at 0.

Lastly, same as Vorba et al. [VKŠ*14], we also apply the Robbins-Monroe averaging scheme (with the same sequence η_j) to the approximate normalization factor that is used in the M-step.

Maximum a-posteriori EM. To learn incident radiance, we use maximum a-posteriori (MAP) EM, which is an extension of the maximum-likelihood EM algorithm that we outlined in Section 3.2. Unlike maximum-likelihood EM, MAP EM takes "prior" information about the mixture parameters (π_k, μ_k, Σ_k) into account. Such prior information regularizes the optimization by biasing EM towards solutions that resemble a chosen prior distribution. Following Gauvain and Lee [GL94], we choose a prior Dirichlet distribution over the weights π_k and a prior inverse Wishart distribution over the covariances Σ_k (no prior is used over μ_k). This formulation results in a closed-form MAP EM optimization that matches maximum-likelihood EM, except for the following modifications to the tangent-space update formulae of the mixture weights $\hat{\pi}_k$ and the covariance matrices $\hat{\Sigma}_k$:

$$\hat{\pi}_k = \frac{\beta_j q + S^{(0)}_k}{K\beta_j q + \sum_i^K S^{(0)}_i}\,,\quad \hat{\Sigma}_k = \frac{\beta_j B + S^{(2)}_k}{\beta_j a + S^{(0)}_k}\,. \tag{20}$$

The scalars q, a and the matrix B parameterize the prior distributions and β_j progressively downscales the strength of the priors as the training step j increases. This downscaling is a consequence of adding more samples to a Bayesian estimator, and in practice

¶ The assumptions include a uniform bound on $S^{(i)}$, which is not necessarily met in rendering. Nonetheless, we observe stable behavior in practice.

removes the bias introduced by the prior distributions when sufficiently many data points have been observed for a robust maximum likelihood estimate.

We choose the prior parameters

$$q = 1/K, \quad a = 5/K, \quad B = a\,\mathrm{diag}\,(0.1, 0.1, 1, 1, 1) \times 10^{-4}, \quad (21)$$

where the first two entries on the diagonal of B are responsible for regularizing the directional tangent-space dimensions ν and the remaining entries are responsible for the spatial dimensions $\mathbf{x}$—hence their different scale.

The relatively large Dirichlet prior q ensures that each mixture component is weighted roughly equally in the early stages of the optimization, whereas the relatively large Wishart prior parameter a ensures that the covariance matrix does not fluctuate too strongly due to initially chosen mixture weights. Since $S_i^{(0)}$ is roughly inversely proportional to the number of mixture components K, we include a division by K in our definitions of q and a. The effect is that the relative prior strength per mixture component is constant.

Our choice of a small Wishart prior parameter B serves the purpose of preventing the covariance matrices from collapsing when few samples are available, e.g. onto a firefly. Other than that (e.g. as soon as $S_k^{(0)}$ is sufficiently large), the small value of B does not noticeably restrict the shape of the learned covariance matrices. Thus, accurately tailoring the spatial components of B to the scene scale is usually not necessary. Nonetheless, we normalize $\mathbf{x}$ to span the range $[0,1]^3$ within the scene's bounding box, and make the initialization of the covariance matrices proportional to the size of the kD-tree leaf nodes containing them. In extreme cases, such as an expansive outdoor environment, it could possibly be necessary to also additionally scale the spatial components of B by the size of the corresponding kD-tree leaf node. We experimented with such an approach, but had difficulties in making it work well across all our scenes due to numerical instabilities.

5. Results

We implemented our method within the Mitsuba renderer [Jak10], making heavy use of SIMD vectorization via Enoki [Jak19] for computations pertaining to our mixture models. Our reference implementation will be released publicly upon publication of this work.

All images in these section were generated at a resolution of 640×360 pixels, using 1024 samples per pixel, on an Intel Xeon W-2135 CPU with 12 cores and 64 GB RAM. The reference images were rendered with standard path tracing using a very large sample count that was chosen for each scene such that the reference image has no visible left-over noise. All comparisons have next-event estimation and Russian roulette disabled and the number of path vertices is bounded to 10.

We compare our method with the improved version [Mül19] of "Practical Path Guiding" (PPG) [MGN17] as well as with parallax-aware von Mises-Fisher mixtures [RHL20] trained on radiance (VMM radiance).

When comparing with PPG, we disable PPG's learning of the BSDF selection probability to ensure a fair comparison to our work, which does not currently have this feature, but could be extended with it. Instead, we use the fixed BSDF sampling fraction of 50% for PPG and for our radiance-based guiding scheme (SDMM radiance), and 30% for our product-based guiding scheme (SDMM product).

Like PPG, our method automatically chooses when to terminate training based on the overall rendering budget and is designed to include all iterations—training and rendering—in the final image. The implementation of Ruppert et al. [RHL20], however, has distinct training and rendering components, which puts it at a disadvantage in online-training comparisons. Therefore, in addition to comparing with Ruppert et al. [RHL20] at equal sample counts (Table 1 and Figure 9) in an online-training setting, we provide an equal training- and rendering time comparison in Table 2 and Figure 10.

Online training comparisons. In Table 1 and Figure 9 we compare PPG, radiance-based VMMs, and our method with (SDMM product) and without (SDMM radiance) product guiding, on 12 virtual scenes that have varied illumination characteristics. For each method and scene, we report the mean absolute percentage error (MAPE), the render time, and the speedup vs. PPG at reaching the lowest common MAPE value. MAPE is defined as $\frac{1}{N}\sum_{i=1}^{N} |v_i - \hat{v}_i| / (\hat{v}_i + \varepsilon)$, where $\hat{v}_i$ is the value of the i-th pixel in the reference image, v_i is the value of the i-th rendered pixel, and $\varepsilon = 0.01$ prevents near-black pixels from dominating the metric. A $2\times$ smaller MAPE loosely corresponds to $4\times$ faster rendering.

A common trend of our approach is that it achieves the lower per-sample error than PPG in all scenes and achieves slightly lower error than radiance-based VMMs in 8 our of 12 scenes. Product-based guiding further reduces the error, but at great computational cost.

As expected, our method performs well when there exists a large degree of spatio-directional correlation in the incident radiance, e.g. as induced by small, local luminaires in the BOOKSHELF, the CORNELL BOX, the WATER CAUSTIC, and the YET ANOTHER BOX scenes; see the insets in Figure 9.

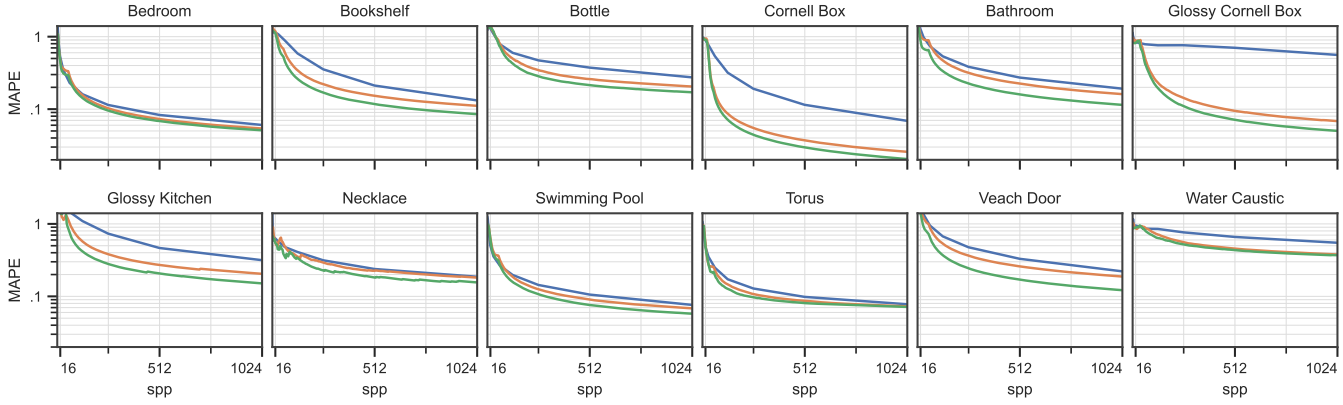
However, the computational overhead of our technique is significantly larger than that of PPG and VMMs. Taking the render time into account and comparing *time to equal error*, our radiance- and product-based guiding approaches only outperform PPG in 6 out of 12 scenes, with our radiance- and product-based guiding performing competitively with each other. In Figure 8, we reinforce this observation by plotting MAPE against render time and against the number of samples per pixel. We report additional metrics and false-color error visualizations in our supplementary HTML-based results viewer.

Offline training comparisons. To provide a fair equal-time ecomparison with Ruppert et al. [RHL20], we match their experimental setup where each method trains and renders for a fixed, equal time; see Table 2 and Figure 10. We have made no attempt to fine-tune our method to this paradigm, and no attempt to modify their method to include training iterations into the final image. Instead, we use both methods as-is, and compare the images generated after training has been terminated in both methods. To ensure fairness, we disabled the inverse-variance based combination of iterations for our method and instead weigh all samples equally.

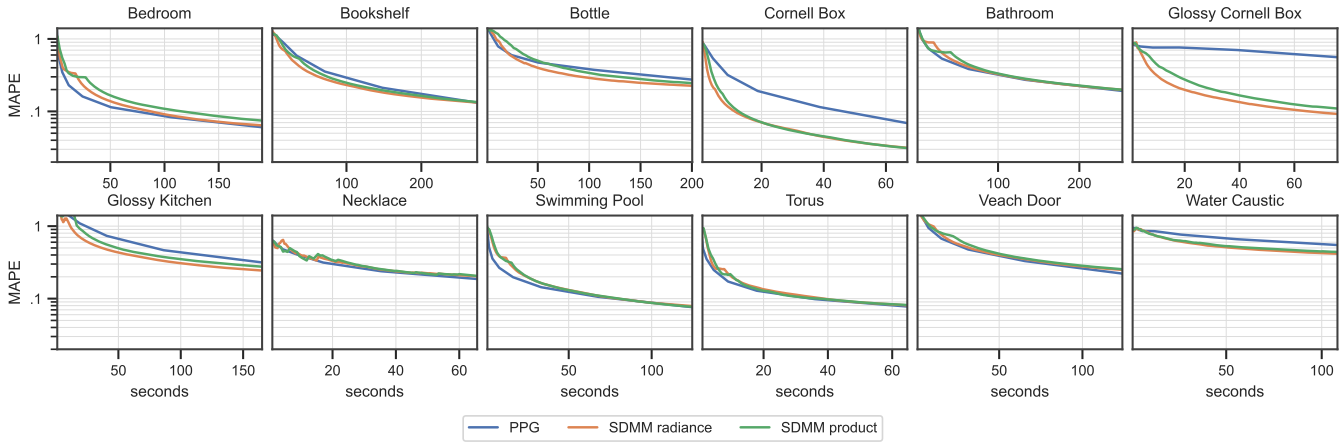
We note that Ruppert et al. [RHL20] outperform our algorithm on the majority of the scenes. Their algorithm is significantly faster

Table 1: Comparison of our method (SDMM radiance & product) with practical path guiding (PPG) [Mül19] and parallax-aware von Mises-Fisher mixtures (VMM radiance) [RHL20] using the respective authors' implementations. We rendered each scene with 1024 spp, where the training and rendering stages of each method (except for PPG) were allotted 252 and 772 spp. We report mean absolute percentage error (MAPE), render time, and the speedup vs. PPG at reaching the lowest common MAPE value. For Ruppert et al., we omit the speedup due to their separation of training and rendering. Instead, we report equal-time results in Figure 10. The best entries are highlighted in **bold** letters. Our radiance-based method (SDMM radiance) achieves lower error than PPG and on average similar error as radiance-based VMMs, but at the cost of increased render time. Incorporating the product (SDMM product) reduces MAPE further while again increasing the render time.

	[Mül19]			[RHL20]			Ours					
	PPG			VMM radiance			SDMM radiance			SDMM product		
BATHROOM	0.192	4.2m	**(1.0×)**	0.148	3.1m		0.161	6.5m	(0.9×)	**0.115**	12m	(0.9×)
BEDROOM	0.060	3.2m	(1.0×)	0.062	2.7m		0.054	4.5m	(0.9×)	**0.051**	7.1m	(0.7×)
BOOKSHELF	0.132	4.6m	(1.0×)	0.105	3.4m		0.111	6.6m	**(1.0×)**	**0.085**	11m	(1.0×)
BOTTLE	0.274	3.3m	(1.0×)	**0.152**	2.5m		0.205	4.3m	**(1.8×)**	0.171	9.5m	(1.3×)
CORNELL BOX	0.069	1.1m	(1.0×)	0.044	43s		0.026	1.6m	**(3.2×)**	**0.020**	2.2m	(3.2×)
GLOSSY KITCHEN	0.316	2.8m	(1.0×)	0.250	1.9m		0.205	4.4m	**(1.7×)**	**0.151**	11m	(1.3×)
NECKLACE	0.187	1.1m	**(1.0×)**	**0.127**	1.1m		0.181	1.5m	(0.8×)	0.157	2.7m	(0.8×)
SWIMMING POOL	0.076	2.1m	**(1.0×)**	0.079	2.0m		0.068	2.9m	(0.9×)	**0.058**	4.0m	(1.0×)
TORUS	0.078	1.1m	**(1.0×)**	0.085	1.1m		0.073	1.5m	(0.9×)	**0.072**	2.0m	(0.8×)
VEACH DOOR	0.221	2.1m	**(1.0×)**	0.286	1.5m		0.188	3.6m	(0.8×)	**0.122**	8.3m	(0.8×)
WATER CAUSTIC	0.549	1.8m	(1.0×)	0.639	1.9m		0.379	2.6m	**(2.8×)**	**0.370**	3.8m	(2.4×)
YET ANOTHER BOX	0.560	1.3m	(1.0×)	0.080	1.1m		0.068	2.4m	**(15.5×)**	**0.050**	5.2m	(9.2×)



(a) MAPE vs. samples per pixel



(b) MAPE vs. render time

Figure 8: We analyze the convergence behavior of our radiance- and product-based guiding algorithms by plotting MAPE (a) against the number of samples per pixel and (b) against the render time. (a) At equal sample counts, our algorithms consistently outperform PPG. (b) However, at equal render time, we are mostly on par with PPG due to our larger computational overhead. In scenes with strong spatio-directional correlation in the incident radiance, our algorithm significantly outperforms PPG (CORNELL BOX, BOOKSHELF, WATER CAUSTIC, YET ANOTHER BOX, GLOSSY KITCHEN).

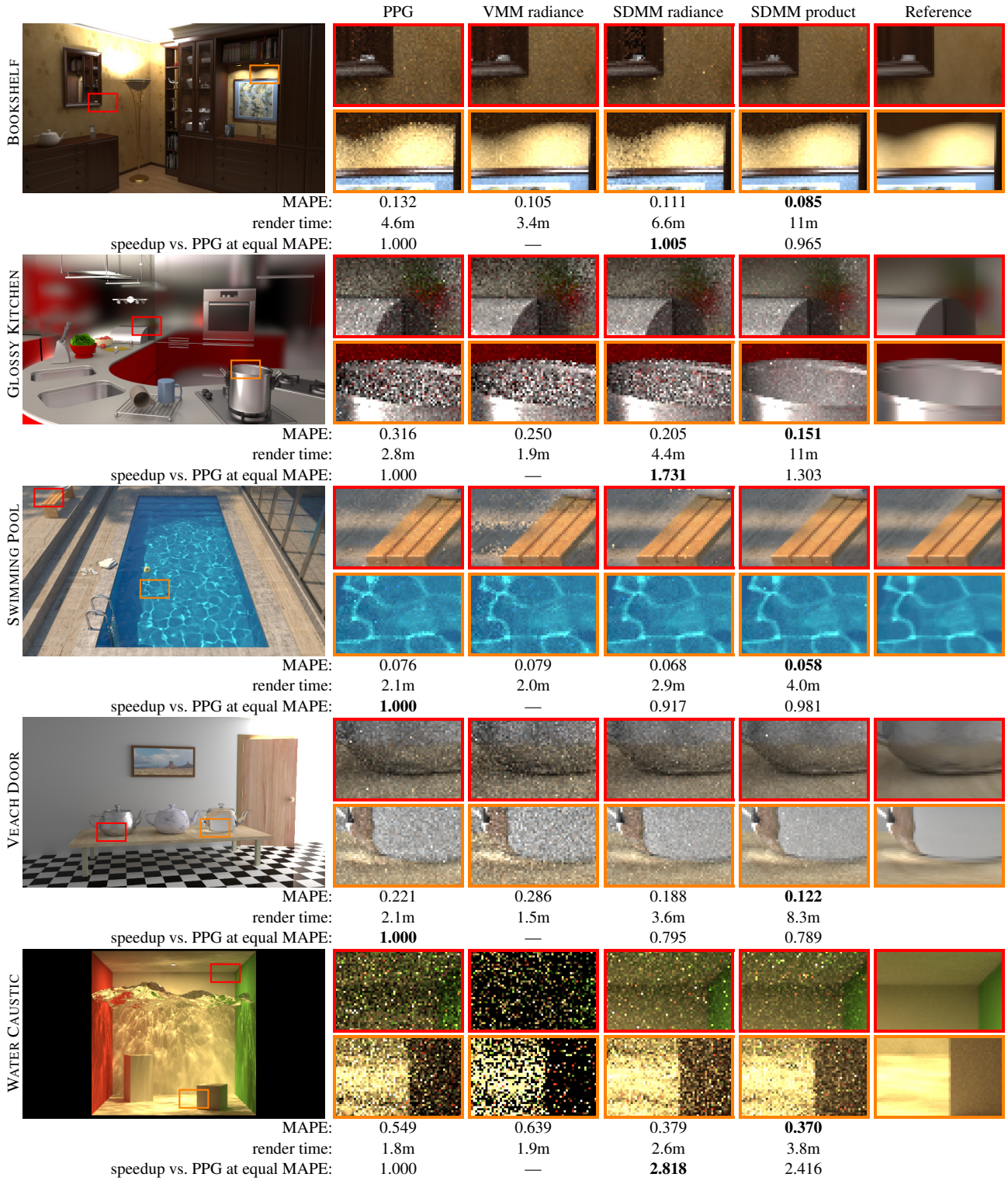


Figure 9: Visual comparison of the same experimental setup as in Table 1. For Ruppert et al., we omit the speedup due to their separation of training and rendering. Instead, we report equal-time results in Figure 10. Our radiance-based method (SDMM radiance) consistently achieves lower error than PPG and on average similar error as radiance-based VMMs. Incorporating the product (SDMM product) reduces the error further. However, the computational overhead of our methods is larger than that of the other methods, leading to VMMs performing best at equal time. As expected, our methods perform better when the incident radiance exhibits significant spatio-directional correlation (WATER CAUSTIC, BOOKSHELF, and GLOSSY KITCHEN) and performs worse in the opposite case, e.g. environment lighting in the SWIMMING POOL.

Table 2: Comparison of radiance-based VMMs [RHL20] with our SDMMs at equal training and rendering time. For both methods, we report mean absolute percentage error (MAPE) as well as training + rendering time. Due to their much lower computational cost, VMMs produce lower error than SDMMs on most scenes.

	Ruppert et al. [RHL20]		SDMM Radiance (Ours)	
Bathroom	**0.104**	1.8m+4.7m	0.179	1.8m+4.7m
Bedroom	**0.054**	1.4m+3.0m	0.059	1.4m+3.0m
Bookshelf	**0.080**	1.8m+4.8m	0.121	1.8m+4.8m
Bottle	**0.116**	1.1m+3.2m	0.213	1.1m+3.2m
Cornell Box	0.032	31s +1.1m	**0.029**	31s +1.1m
Glossy Kitchen	**0.176**	1.1m+3.3m	0.224	1.1m+3.3m
Necklace	**0.115**	27s +1.0m	0.183	27s +1.0m
Swimming Pool	**0.073**	55s +2.0m	0.073	55s +2.0m
Torus	0.078	34s + 59s	**0.074**	34s + 59s
Veach Door	**0.182**	1.0m+2.6m	0.206	1.0m+2.6m
Water Caustic	0.515	37s +2.0m	**0.395**	37s +2.0m
Yet Another Box	**0.056**	36s +1.8m	0.070	36s +1.8m

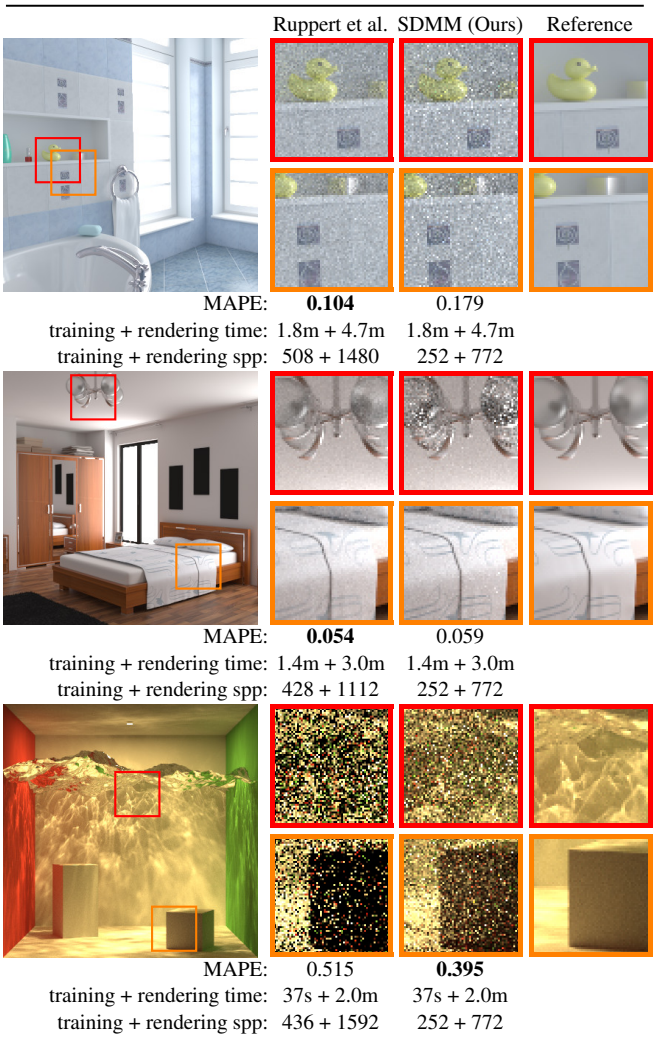


Figure 10: Visual comparison of the same experimental setup as in Table 2 for three selected scenes.

than ours, meaning that it is able to render far more samples in the same time compared as ours. Thus, even though the quality of learned mixture models is comparable—as evidenced by the equal sample count comparison Figure 9—their method is more practical in most of our test scenes.

Quality of the guiding distribution. In Figure 11, we visualize the learned spatio-directional mixture model in 3 scenes. For each scene, we show the 2D distributions obtained by conditioning the mixture model on 2 indicated spatial locations. The mixture model is not only accurate, but it also captures high-frequency spatio-directional correlation, which we illustrate in our supplementary video by smoothly varying the spatial coordinate that the model is conditioned on.

6. Discussion and Future Work

Mini-batch EM versus stepwise EM. Stepwise EM [CM09], later extended to robustly handle weighted samples by Vorba et al. [VKŠ*14], is an alternative online EM algorithm that could be used instead of mini-batch EM to train SDMM. The difference between the two algorithms is the frequency of the Robbins-Monroe update of the sufficient statistics Equation (18): in stepwise EM, the update is performed per sample, whereas in mini-batch EM, it is performed per mini-batch.

Because the aforementioned update is expensive in tangent spaces (due to Equation (19)), we prefer using mini-batch EM. Another argument in favor of mini-batch EM is that stepwise EM unduly weights earlier samples within the same batch higher due to them experiencing the Robbins-Monroe update earlier, despite them being sampled from the same distribution. This uneven averaging of the sufficient statistics of the batch unnecessarily increases the variance of the optimization.

Practicality of SDMM. Compared with PPG [Mül19; MGN17], we demonstrated superior equal-sample-count error of both our radiance- and product-based guiding approaches. However, the computational overhead of our Mitsuba implementation results in better overall efficiency on only a subset of the scenes. Compared with VMMs [RHL20], the difference is even larger: while we achieve similar error at equal sample counts, their lower computational cost leads to better efficiency than ours in most scenes. The practicality of spatio-directional mixture models is thus limited and further research into efficient implementations and approximations is needed.

To this end, we believe there are a promising number of optimizations that can still be made, such as automatic pruning of the product mixture [HEV*16]. Finally, the computational bottleneck may shift when more complex scenes are rendered, since the cost of ray tracing and shading will be larger.

Combination with Reprojection. While our data-driven approach of learning 5D mixtures may seem in contrast with the reprojection of 2D mixtures [RHL20], we believe that the two approaches are not mutually exclusive. In particular, there are situations in which the reprojection heuristic breaks down. Lensing effects are one such example, in which the appropriate hemispherical movement is actually the *reverse* of what reprojection would predict; we show an

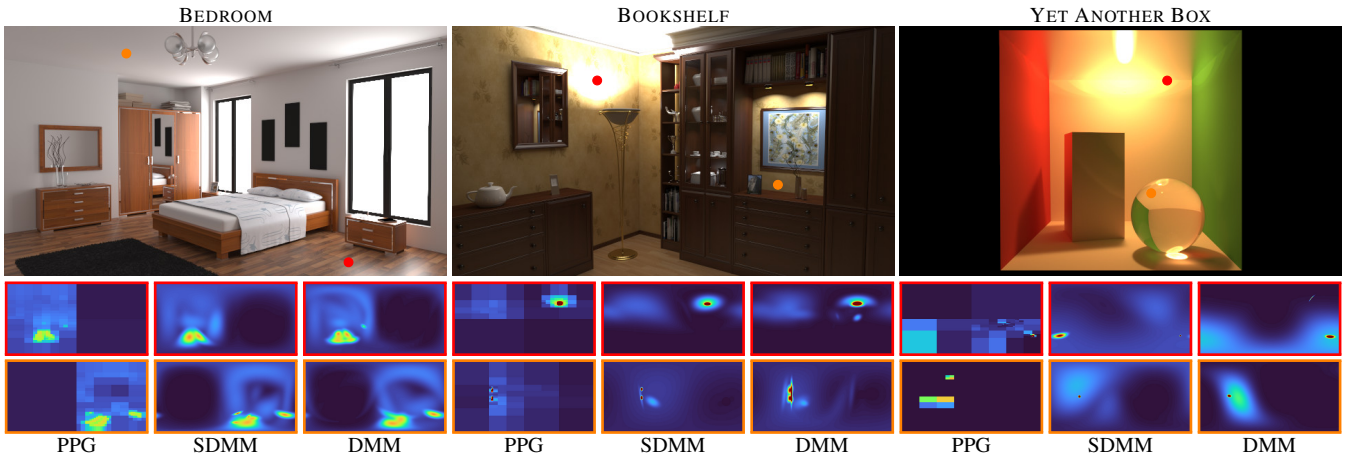


Figure 11: Learned directional PDFs, conditioned on two spatial locations per scene, visualized as false-color images in spherical coordinates. We compare our learned distributions (SDMM) with those learned by PPG and by a purely directional version of our mixture model (DMM). As expected, the SDMMs produce sharper distributions than the spatially marginalized DMMs without exhibiting discretization artifacts like PPG does—especially in the difficult YET ANOTHER BOX scene. Zooming in is recommended. The SDMMs also smoothly capture high-frequency spatio-directional correlation, which we illustrate in our supplementary video by smoothly varying the queried spatial coordinate.

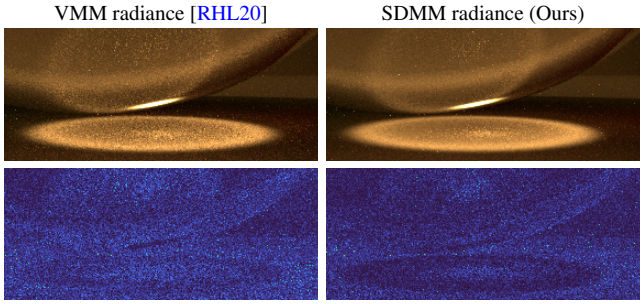


Figure 12: While the parallax heuristic of Ruppert et al. [RHL20] is cheap and accurate in most situations, certain effects, such as lensing, can not only result in non-linear parallax, but even invert its direction. Such effects are rarely relevant in practice, but we nonetheless show one such occurrence: the caustic in the YET ANOTHER BOX scene. The lower noise of our method at an equal number of samples is indicative of a locally more accurate fit to incident radiance.

example in Figure 12. While lensing effects occur relatively rarely in practice, this example demonstrates one the benefits of using a data-driven approach compared to hand-crafted hauristics. Thus, we believe that one could use 5D mixtures to learn an *offset* on top of the reprojections of Ruppert et al. [RHL20], fixing the rare special cases where the assumptions of the reprojection are incorrect.

Computational implications of tangent spaces. Although product sampling is enabled by parameterizing each Gaussian in its own tangent space, such a per-Gaussian parameterization necessitates the on-the-fly computation of numerous changes of variables: a change of variables is needed whenever multiple Gaussians interact, regardless of whether this interaction amounts to conditioning, product sampling, or EM optimization. Despite the optimizations that we already perform, these changes of variable are the computational bottleneck of our implementation. Thus, the efficiency of our method could be greatly improved by coming up with means to reduce the number of changes of variables, e.g. through a different mixture model, or to compute them more efficiently.

Alternative acceleration structures The choice of placing our Gaussians within a kD-tree was motivated by the intractable quadratic cost of using a single global mixture; see also Figure 5. Alternatively, the quadratic cost could also be mitigated by ignoring Gaussians whose density is below a threshold, e.g. by placing the Gaussians in a BVH, where the bounding volumes contain a large percentage of each Gaussians' mass. However, such an approach maintains two disadvantages over our kD-tree approach: first, using a global mixture may lead to Gaussians that cover significant portions of the scene, which result in quadratic cost regardless of acceleration structure. And second, thresholding the size of Gaussians without introducing large error is non-trivial, because the conditioning step may amplify small densities unpredictably. We implemented a prototype and experimentally confirmed these difficulties [Dod20].

Extensions. Our implementation does not perform a number of well-established extensions to path-guiding techniques to facilitate a simple comparison with previous work. When using our algorithm in practice, such orthogonal extensions should, however, be used. First, next-event estimation (NEE) should be performed and the learned guiding distribution should be optimized to *complement* NEE [MMR*19]. In the context of Gaussian mixtures, Ruppert et al. [RHL20] describe an appropriate modification to the Monte Carlo samples that the model is trained on. Second, the BSDF sampling fraction should be learned [Mül19] as opposed to set to a fixed constant. Third, adjoint-driven Russian roulette and splitting [VK16] should be performed.

Furthermore, there are also mixture-specific techniques that could conceivably enhance our algorithm. For example, careful splitting and merging of mixture components could simultaneously increase our model capacity as well as escape local minima in the EM optimization [RHL20]. We suspect that the splitting might only be necessary along the directional dimensions of the SDMMs, since the kD-tree data-structure already performs spatial splitting.

High-dimensional mixtures. We demonstrated the ability of Gaussian mixtures to directly approximate the 5D incident radiance as well as certain nD BSDF models. However, the BSDF models that we used were relatively simple, never exceeding $n = 5$. In the future, we would like to investigate the use of higher-dimensional mixtures to capture additional variation, such as found in the 10D Disney BSDF [Bur12], or additional dimensions in the incident radiance, such as the wavelength in spectral rendering. Such higher-dimensional mixtures need advanced data structures and a form of sparsity to combat the curse of dimensionality—a good starting point is likely the manifold framework of Herholz et al. [HES*18].

Non-linear spatio-directional dependencies. Our spatio-directional Gaussians explicitly model correlations (i.e. linear relationships) between the 5 dimensions of the incident radiance field. One example of such a linear relationship if parallax. However, the spatio-directional dimensions often also depend on each other *non-linearly*, which can not be modeled by Gaussian covariance matrices. A mixture of more sophisticated distributions may thus exhibit more accurate fits with fewer mixture components.

7. Conclusion

We demonstrated the feasibility of using spatio-directional mixture models for path guiding. In particular, we showed that a tangent-space parameterization of Gaussians enables product sampling among a 5D mixture that approximates incident radiance and nD mixtures that approximate the BSDFs. The use of thousands of mixture components—a necessity for accurately modeling the intricate radiance field—was made practical by a kD-tree data structure.

The tangent-space Gaussian mixture performed remarkably well in our experiments, making us hopeful that it can become an alternative to von Mises-Fisher mixtures, featuring anisotropy in addition to rotational symmetry.

As for data-driven 5D spatio-directional mixtures: they achieve competetive results to previous work in equal sample count comparisons, but are outperformed by other spatio-directional models that recover spatio-directional correlation from reprojection [RHL20]. Nonetheless, data-driven SDMMs help in the cases where reprojection is inaccurate, such as lensing effects by curved, specular geometries. This makes us hopeful that—if the overhead can be further reduced—SDMMs can become a practical tool on a practitioner's toolbelt, e.g. by combining them with reprojection.

Acknowledgments

We thank Marco Manzi for creating Figure 2 and Alex Keller for valuable feedback. We also thank the following people for providing scenes that appear in our figures: Benedikt Bitterli [Bit16], Johannes Hanika (NECKLACE), Miika Aittala, Samuli Laine, and Jaakko Lehtinen (VEACH DOOR), Olesya Jakob (TORUS), Ondřej Karlík (SWIMMING POOL), SlykDrako (BEDROOM), and Tiziano Portenier (BATHROOM, BOOKSHELF). Lastly, we thank Markus Gross for enabling and supporting this research.

References

[AV07] ARTHUR, DAVID and VASSILVITSKII, SERGEI. "K-Means++: The Advantages of Careful Seeding". *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms*. SODA '07. New Orleans, Louisiana: Society for Industrial and Applied Mathematics, 2007, 1027–1035. ISBN: 9780898716245 9.

[Bin74] BINGHAM, CHRISTOPHER. "An Antipodally Symmetric Distribution on the Sphere". *The Annals of Statistics* 2.6 (1974), 1201–1225. ISSN: 00905364. URL: http://www.jstor.org/stable/2958339 3.

[Bit16] BITTERLI, BENEDIKT. *Rendering resources*. https://benedikt-bitterli.me/resources/. 2016 15.

[Bur12] BURLEY, BRENT. "Physically-Based Shading at Disney". 2012 15.

[Cap11] CAPPÉ, OLIVIER. "Online Expectation-Maximisation". *Mixtures: Estimation and Applications*. Ed. by MENGERSEN, K., TITTERINGTON, M., and ROBERT, C. P. Wiley, Aug. 2011, 1–53. URL: https://hal.archives-ouvertes.fr/hal-00532968 5, 9.

[CDG*08] CAPPÉ, OLIVIER, DOUC, RANDAL, GUILLIN, ARNAUD, et al. "Adaptive Importance Sampling in General Mixture Classes". *Statistics and Computing* 18.4 (2008). Removed misleading comment in Section 2, 447–459. DOI: 10.1007/s11222-008-9059-x. URL: https://hal.archives-ouvertes.fr/hal-00180669 3.

[CM09] CAPPÉ, OLIVIER and MOULINES, ERIC. "On-line expectation-maximization algorithm for latent data models". *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 71.3 (June 2009), 593–613. ISSN: 1467-9868. DOI: 10.1111/j.1467-9868.2009.00698.x. URL: http://dx.doi.org/10.1111/j.1467-9868.2009.00698.x 5, 9, 13.

[DK18] DAHM, KEN and KELLER, ALEXANDER. "Learning Light Transport the Reinforced Way". *Monte Carlo and Quasi-Monte Carlo Methods*. Ed. by OWEN, ART B. and GLYNN, PETER W. Springer International Publishing, 2018, 181–195. ISBN: 978-3-319-91436-7 3.

[Dod20] DODIK, ANA. "Path Guiding using a Spatio-Directional Mixture Model". 2020 5, 8, 14.

[GBBE18] GUO, JERRY JINFENG, BAUSZAT, PABLO, BIKKER, JACCO, and EISEMANN, ELMAR. "Primary Sample Space Path Guiding". *Eurographics Symposium on Rendering - Experimental Ideas & Implementations*. Ed. by JAKOB, WENZEL and HACHISUKA, TOSHIYA. The Eurographics Association, 2018. ISBN: 978-3-03868-068-0 3.

[GL94] GAUVAIN, JEAN-LUC and LEE, CHIN-HUI. "Maximum a posteriori estimation for multivariate Gaussian mixture observations of Markov chains". *IEEE Trans. Speech Audio Process.* 2 (1994), 291–298 9.

[HD14] HEITZ, ERIC and D'EON, EUGENE. "Importance Sampling Microfacet-Based BSDFs using the Distribution of Visible Normals". *Computer Graphics Forum* 33.4 (July 2014), 103–112. DOI: 10.1111/cgf.12417. URL: https://hal.inria.fr/hal-00996995 3.

[HES*18] HERHOLZ, SEBASTIAN, ELEK, OSKAR, SCHINDEL, JENS, et al. "A Unified Manifold Framework for Efficient BRDF Sampling based on Parametric Mixture Models". *Eurographics Symposium on Rendering - Experimental Ideas & Implementations*. Ed. by JAKOB, WENZEL and HACHISUKA, TOSHIYA. The Eurographics Association, 2018. ISBN: 978-3-03868-068-0 2, 3, 8, 15.

[HEV*16] HERHOLZ, SEBASTIAN, ELEK, OSKAR, VORBA, JIŘÍ, et al. "Product Importance Sampling for Light Transport Path Guiding". *Computer Graphics Forum* (2016). ISSN: 1467-8659. DOI: 10.1111/cgf.12950 2, 3, 13.

[HP02] HEY, HEINRICH and PURGATHOFER, WERNER. "Importance Sampling with Hemispherical Particle Footprints". *Proceedings of the 18th Spring Conference on Computer Graphics*. SCCG '02. Budmerice, Slovakia: ACM, 2002, 107–114. ISBN: 1-58113-608-0. DOI: 10.1145/584458.584476. URL: http://doi.acm.org/10.1145/584458.584476 3.

[HZE*19] HERHOLZ, SEBASTIAN, ZHAO, YANGYANG, ELEK, OSKAR, et al. "Volume Path Guiding Based on Zero-Variance Random Walk Theory". *ACM Trans. Graph.* 38.3 (June 2019), 25:1–25:19. ISSN: 0730-0301. DOI: 10.1145/3230635. URL: http://doi.acm.org/10.1145/3230635 3, 6, 8.

[Jak10] JAKOB, WENZEL. *Mitsuba Renderer*. http://www.mitsuba-renderer.org. 2010 10.

[Jak19] JAKOB, WENZEL. *Enoki: structured vectorization and differentiation on modern processor architectures*. https://github.com/mitsuba-renderer/enoki. 2019 10.

[Jen95] JENSEN, HENRIK WANN. "Importance Driven Path Tracing using the Photon Map". *Rendering Techniques*. Vienna: Springer Vienna, 1995, 326–335. ISBN: 978-3-7091-9430-0. DOI: 10.1007/978-3-7091-9430-0_31. URL: http://dx.doi.org/10.1007/978-3-7091-9430-0_31 3.

[LW93] LAFORTUNE, ERIC P. and WILLEMS, YVES D. "Bi-Directional Path Tracing". *Compugraphics '93*. Alvor, Portugal, 1993, 145–153 3.

[LW95] LAFORTUNE, ERIC P. and WILLEMS, YVES D. "A 5D Tree to Reduce the Variance of Monte Carlo Ray Tracing". *Proc. EGWR*. June 1995, 11–20 3.

[MGN17] MÜLLER, THOMAS, GROSS, MARKUS, and NOVÁK, JAN. "Practical Path Guiding for Efficient Light-Transport Simulation". *Computer Graphics Forum* 36.4 (June 2017), 91–100. ISSN: 1467-8659. DOI: 10.1111/cgf.13227. URL: http://dx.doi.org/10.1111/cgf.13227 2, 3, 8, 10, 13.

[MMR*19] MÜLLER, THOMAS, MCWILLIAMS, BRIAN, ROUSSELLE, FABRICE, et al. "Neural Importance Sampling". *ACM Trans. Graph.* 38.5 (Oct. 2019), 145:1–145:19. ISSN: 0730-0301. DOI: 10.1145/3341156. URL: http://doi.acm.org/10.1145/3341156 3, 14.

[MRNK20] MÜLLER, THOMAS, ROUSSELLE, FABRICE, NOVÁK, JAN, and KELLER, ALEXANDER. "Neural Control Variates". *arXiv:2006.01524* (June 2020) 3.

[Mül19] MÜLLER, THOMAS. ""Practical Path Guiding" in Production". *ACM SIGGRAPH Courses: Path Guiding in Production, Chapter 10*. Los Angeles, California: ACM, 2019, 18:1–18:77. DOI: 10.1145/3305366.3328091 3, 10, 11, 13, 14.

[NFM20] NGUYEN, HIEN D., FORBES, FLORENCE, and MCLACHLAN, GEOFFREY J. "Mini-batch learning of exponential family finite mixture models". *Statistics and Computing* 30.4 (Jan. 2020), 731–748. ISSN: 1573-1375. DOI: 10.1007/s11222-019-09919-4. URL: http://dx.doi.org/10.1007/s11222-019-09919-4 5, 9.

[Pen06] PENNEC, XAVIER. "Intrinsic Statistics on Riemannian Manifolds: Basic Tools for Geometric Measurements". *Journal of Mathematical Imaging and Vision* 25.1 (2006), 127. DOI: 10.1007/s10851-006-6228-4. URL: https://doi.org/10.1007/s10851-006-6228-4 2–4.

[RHJD18] REIBOLD, FLORIAN, HANIKA, JOHANNES, JUNG, ALISA, and DACHSBACHER, CARSTEN. "Selective Guided Sampling with Complete Light Transport Paths". *ACM Trans. Graph.* 37.6 (Dec. 2018). ISSN: 0730-0301. DOI: 10.1145/3272127.3275030. URL: https://doi.org/10.1145/3272127.3275030 3.

[RHL20] RUPPERT, LUKAS, HERHOLZ, SEBASTIAN, and LENSCH, HENDRIK P. A. "Robust Fitting of Parallax-Aware Mixtures for Path Guiding". *ACM Trans. Graph.* 39.4 (July 2020). ISSN: 0730-0301. DOI: 10.1145/3386569.3392421. URL: https://doi.org/10.1145/3386569.3392421 2, 3, 10, 11, 13–15.

[RM51] ROBBINS, HERBERT and MONRO, SUTTON. "A Stochastic Approximation Method". *The Annals of Mathematical Statistics* 22.3 (1951), 400–407. ISSN: 00034851. URL: http://www.jstor.org/stable/2236626 9.

[SPK05] SALOJÄRVI, JARKKO, PUOLAMÄKI, KAI, and KASKI, SAMUEL. "Expectation Maximization Algorithms for Conditional Likelihoods". *Proceedings of the 22nd International Conference on Machine Learning*. ICML '05. Bonn, Germany: Association for Computing Machinery, 2005, 752–759. ISBN: 1595931805. DOI: 10.1145/1102351.1102446. URL: https://doi.org/10.1145/1102351.1102446 2, 5.

[STM14] SIMO-SERRA, EDGAR, TORRAS, CARME, and MORENO-NOGUER, FRANCESC. "Geodesic Finite Mixture Models". *Proceedings of the British Machine Vision Conference*. BMVA Press, 2014. DOI: http://dx.doi.org/10.5244/C.28.91 2–4.

[Vea97] VEACH, ERIC. "Robust Monte Carlo methods for light transport simulation". PhD thesis. Stanford, CA, USA, Dec. 1997. ISBN: 0-591-90780-1 3.

[VG95] VEACH, ERIC and GUIBAS, LEONIDAS J. "Optimally Combining Sampling Techniques for Monte Carlo Rendering". *Proc. SIGGRAPH*. 1995, 419–428. ISBN: 0-89791-701-4. DOI: 10.1145/218380.218498. URL: http://doi.acm.org/10.1145/218380.218498 3.

[VHH*19] VORBA, JIŘÍ, HANIKA, JOHANNES, HERHOLZ, SEBASTIAN, et al. "Path Guiding in Production". *ACM SIGGRAPH Courses*. Los Angeles, California: ACM, 2019, 18:1–18:77. DOI: 10.1145/3305366.3328091 3.

[VK16] VORBA, JIŘÍ and KŘIVÁNEK, JAROSLAV. "Adjoint-Driven Russian Roulette and Splitting in Light Transport Simulation". *ACM Trans. Graph.* 35.4 (July 2016) 14.

[VKŠ*14] VORBA, JIŘÍ, KARLÍK, ONDŘEJ, ŠIK, MARTIN, et al. "On-line Learning of Parametric Mixture Models for Light Transport Simulation". *ACM Trans. Graph.* 33.4 (Aug. 2014) 2, 3, 5, 9, 13.

[WMLT07] WALTER, BRUCE, MARSCHNER, STEPHEN R., LI, HONGSONG, and TORRANCE, KENNETH E. "Microfacet Models for Refraction through Rough Surfaces". *Proceedings of the 18th Eurographics Conference on Rendering Techniques*. EGSR'07. Grenoble, France: Eurographics Association, 2007, 195–206. ISBN: 9783905673524 3.

[XSD*13] XU, KUN, SUN, WEI-LUN, DONG, ZHAO, et al. "Anisotropic Spherical Gaussians". *ACM Transactions on Graphics* 32.6 (2013), 209:1–209:11 3.

[ZZ19] ZHENG, QUAN and ZWICKER, MATTHIAS. "Learning to Importance Sample in Primary Sample Space". *Computer Graphics Forum* 38.2 (2019), 169–179. DOI: 10.1111/cgf.13628 3.

Appendix A: Jacobian Matrices

The Jacobian matrices of the μ-centered $\log_\mu$ and $\exp_\mu$ maps can be derived by differentiating Equations (7) and (8), resulting in

$$J_{\log_\mu}(\omega) = \begin{bmatrix} \mathcal{J}_{\log_\mu} & 0 & \omega_x^{\circlearrowleft} \frac{\omega_z^{\circlearrowleft} - \mathcal{J}_{\log_\mu}}{(1-\omega_z^{\circlearrowleft}\omega_z^{\circlearrowleft})\mathcal{J}_{\log_\mu}} \\ 0 & \mathcal{J}_{\log_\mu} & \omega_y^{\circlearrowleft} \frac{\omega_z^{\circlearrowleft} - \mathcal{J}_{\log_\mu}}{(1-\omega_z^{\circlearrowleft}\omega_z^{\circlearrowleft})\mathcal{J}_{\log_\mu}} \end{bmatrix} R_\mu \quad (22)$$

$$J_{\exp_\mu}(v) = R_\mu^{-1} \begin{bmatrix} \operatorname{sinc}\|v\| + v_x^2 \mathcal{J}_{\exp_\mu} & \operatorname{sinc}\|v\| + v_x v_y \mathcal{J}_{\exp_\mu} \\ \operatorname{sinc}\|v\| + v_y v_x \mathcal{J}_{\exp_\mu} & \operatorname{sinc}\|v\| + v_y \mathcal{J}_{\exp_\mu} \\ -v_x \operatorname{sinc}\|v\| & -v_y \operatorname{sinc}\|v\| \end{bmatrix}, \quad (23)$$

$$\mathcal{J}_{\log_\mu} = \frac{1}{\operatorname{sinc}(\cos^{-1}(\omega_z^{\circlearrowleft}))}, \quad \mathcal{J}_{\exp_\mu} = \frac{\cos\|v\| - \operatorname{sinc}\|v\|}{\|v\|^2}, \quad (24)$$

where $\omega^{\circlearrowleft} = R_\mu \omega$ and R_μ is the rotation matrix that rotates μ to $(0,0,1)$. We use the *unnormalized* sinc function.

Appendix B: Covariance Initialization

Given an initial 5D mean-vector $(\mathbf{x}, \omega)$ of a Gaussian mixture component, we wish to initialize its covariance matrix such that it is

- isotropic in the tangent plane of the surface at $\mathbf{x}$ and in the tangent space around ω, and
- flat along the surface normal at $\mathbf{x}$.

To this end, we let 90% of the mass of each Gaussian to be contained within a certain radius, r_{st} in the surface's tangent plane, and r_n along the normal. More concretely, we compute the initial spatial 3×3 part of the covariance matrix as

$$\Sigma^{\mathbf{x}} = \frac{r_{st}^2 \mathbf{s}\mathbf{s}^T + r_{st}^2 \mathbf{t}\mathbf{t}^T + r_n^2 \mathbf{n}\mathbf{n}^T}{\chi_3^{2^{-1}}(0.9)}, \tag{25}$$

where $\mathbf{s}, \mathbf{t}, \mathbf{n}$ form the basis of the local coordinate frame at the selected point and $\chi_3^{2^{-1}}$ is the inverse cumulative distribution function of the chi-square distribution. We empirically found good results with the values $r_n = 3 \cdot 10^{-2}$, and $r_{st} = 2 \cdot \frac{d}{2}$, where d is the length of the longest side of the leaf node containing the Gaussian and the division by 2 is due to the number of distinct spatial mean vectors in the mixture.

The initial directional 2×2 part Σ^{ω} of the covariance matrix, living in the ω-centered tangent space, is set to be isotropic and inversely proportional to the number of directional Gaussian components at a spacial location (in our case 8), with the diagonal entries being equal to $\sigma^2 = \frac{2\pi}{8}$, resulting in the full covariance matrix

$$\Sigma = \begin{bmatrix} \Sigma^{\omega} & \mathbf{0} \\ \mathbf{0} & \Sigma^{\mathbf{x}} \end{bmatrix}. \tag{26}$$

Appendix C: Modified `k-means++` Algorithm

The modification to the original `k-means++` algorithm consists of defining a custom distance metric $D(\mathbf{x}_i, \mathbf{x}_j)$ between spatial positions $\mathbf{x}_i$ and $\mathbf{x}_j$ that takes their surface normals n_i and n_j into account, as well as defining a custom resampling probability $p^{++}(\mathbf{x}_i)$ that takes the corresponding Monte Carlo weight w_i of the sample at $\mathbf{x}_i$ into account.

Our custom distance metric extends the spatial euclidean distance by the geodesic distance between normals:

$$\mathrm{dist}(\mathbf{x}_i, \mathbf{x}_j)^2 = \left(\cos^{-1}(\mathrm{n}_i \cdot \mathrm{n}_j)/\pi\right)^2 + \|\mathbf{x}_i - \mathbf{x}_j\|^2, \tag{27}$$

where the division by π normalizes geodesic distances to the range $[0,1]$. Additionally, we enforce a poisson-disk distribution of points by setting the distance to zero within pre-defined spatial and spherical radii $T_{\mathbf{x}}$ and T_{n}:

$$D(\mathbf{x}_i, \mathbf{x}_j) = \begin{cases} \mathrm{dist}(\mathbf{x}_i, \mathbf{x}_j) & \text{if } \|\mathbf{x}_i - \mathbf{x}_j\| > T_{\mathrm{n}} \vee \cos^{-1}(\mathrm{n}_i \cdot \mathrm{n}_j) > T_{\mathbf{x}}, \\ 0 & \text{otherwise}. \end{cases} \tag{28}$$

We empirically choose $T_{\mathrm{n}} = 0.2$ and $T_{\mathbf{x}} = 1.6 \times 10^{-3}$.

Building on top of the custom distance metric, we extend the resampling probability $p^{++}(\mathbf{x}_i)$ with the Monte Carlo weight w_i by multiplying the distance to the closest neighbor $\mathbf{x}_j$ by the clamped Monte Carlo weight in the interval $[W_0, W_1]$:

$$p^{++}(\mathbf{x}_i) = \mathrm{clamp}(w_i, W_0, W_1) \min_{j \neq i} D(\mathbf{x}_i, \mathbf{x}_j). \tag{29}$$

The clamping to the interval $[W_0, W_1]$ serves the double-purpose of allowing zero-valued (black) samples to contribute when fewer than 3 radiance-carrying samples are nearby, as well as preventing fireflies from dominating other samples. We empirically found $W_0 = 10^{-1}$ and $W_1 = 3$ to work well.