Eurographics Symposium on Rendering (2024)
E Garces and E. Haines (Editors)

Path Sampling Methods for Differentiable Rendering

Tanli Su and Ioannis Gkioulekas

Carnegie Mellon University

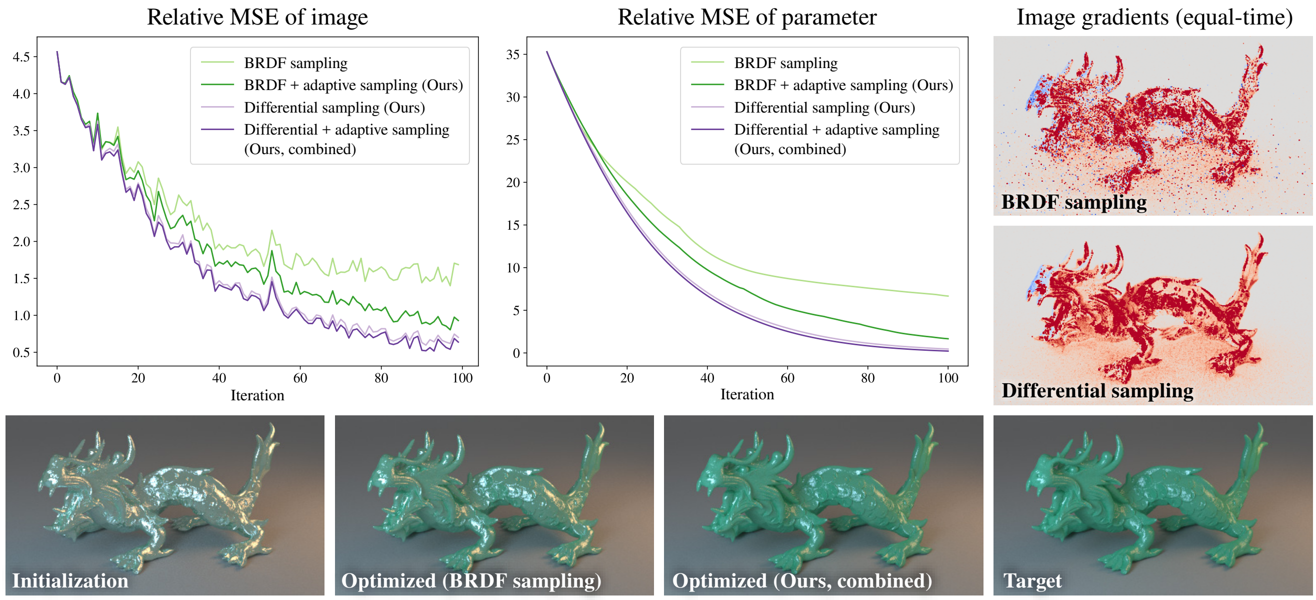

Figure 1: *We introduce differential sampling and adaptive sampling as new path sampling methods for differentiable rendering. Compared to BRDF sampling, our method produces less noisy gradients and better inverse rendering optimization.*

Abstract

We introduce a suite of path sampling methods for differentiable rendering of scene parameters that do not induce visibility-driven discontinuities, such as BRDF parameters. We begin by deriving a path integral formulation for differentiable rendering of such parameters, which we then use to derive methods that importance sample paths according to this formulation. Our methods are analogous to path tracing and path tracing with next event estimation for primal rendering, have linear complexity, and can be implemented efficiently using path replay backpropagation. Our methods readily benefit from differential BRDF sampling routines, and can be further enhanced using multiple importance sampling and a loss-aware pixel-space adaptive sampling procedure tailored to our path integral formulation. We show experimentally that our methods reduce variance in rendered gradients by potentially orders of magnitude, and thus help accelerate inverse rendering optimization of BRDF parameters.

CCS Concepts

• ***Computing methodologies*** → ***Ray tracing;***

1. Introduction

Differentiable rendering has emerged in the past decade as an important methodology for solving inverse and design problems in light transport. Starting with differentiable rendering algorithms specialized to subsurface scattering [GZB*13] and progressing to more general formulations [LADL18], differentiable rendering has advanced to support differentiation of surface and volumetric transport with respect to arbitrary scene parameters. We distinguish between parameters that induce visibility-driven discontinuities—e.g., object shape and pose—versus those that do not—e.g., bidirectional reflectance distribution function (BRDF), scattering material, local shading. We focus on improving rendering performance when differentiating with respect to the latter type of parameters.

Thanks to the similarity between formulations for differentiable and primal rendering, many existing Monte Carlo methods for differentiable rendering use path sampling algorithms identical to those for primal rendering (e.g., path tracing) [VSJ21, CLZ*20, ZMY*20]. However, this practice is suboptimal because such algorithms are designed to approximate path integrands in primal rendering, which in turn can be significantly different from path integrands in differentiable rendering. To date, algorithms that adapt path sampling techniques to differentiable rendering do so in a manner that requires tracing additional "side paths" [ZSGJ21, ZDDZ21, BXB*24]. Consequently, such algorithms exhibit quadratic complexity and come with a computational overhead that greatly outweighs the benefits of the improved importance sampling they provide [VSJ21].

We introduce path sampling methods for differentiable rendering that simultaneously provide improved importance sampling and maintain linear complexity. To this end, we first derive a new differential path integral formulation (Section 3) for differentiable rendering, which we relate to prior such formulations [CLZ*20, GLZ16, VSJ21, NDSRJ20]. We then derive path sampling methods (Section 4) that are analogous to path tracing and path tracing with next event estimation in primal rendering, but importance sample paths based on our formulation. We show that our methods have linear complexity, and can be readily combined with path replay backpropagation [VSJ21] for efficient implementation. We further enhance our path sampling methods using a multiple importance sampling (MIS) procedure tailored to our differential path integral formulation, and a loss-aware pixel-space adaptive sampling procedure suitable for inverse rendering applications.

We use synthetic experiments (Section 5) to show that our improved methods result in significantly reduced variance when estimating derivatives with respect to BRDF parameters, and accelerated convergence in related inverse rendering tasks. The code for our path sampling methods is available on the project website[†].

2. Related work

We focus on differentiable rendering with respect to scene parameters that do not induce visibility-driven discontinuities, and in particular BRDF parameters. Thus, differentiable rendering techniques that deal with such discontinuities [LADL18, ZMY*20, XBLZ23, BLD20, BGL*22, VSJ22, LHJ19, ZWZ*19] are outside our scope.

Original general-purpose differentiable rendering implementations targeted optimization of such parameters, albeit through expensive forward-mode automatic differentiation [NDVZJ19]. Nimier-David et al. [NDSRJ20] use *radiative backpropagation* (RB) to compute derivatives in separate primal and adjoint steps, resulting in much lower memory cost than naive automatic differentiation. However, they require a recursive radiance estimate at each scattering event along a light path, resulting in branching and thus quadratic time complexity. Vicini et al. [VSJ21] achieve linear complexity by introducing *path replay backpropagation* (PRB) to the adjoint step. PRB is analogous to previous *score-based estimators* for differentiable rendering [ZWDR16, GLZ16, CLZ*20, KSZ*15].

Most of these previous techniques reuse path sampling methods for primal rendering (e.g., path tracing, with or without next event estimation and MIS [HHM22]). Yet sampling methods tailored for differentiable rendering can result in significant performance improvements [ZSGJ21]. Nimier-David et al. [NDMKJ22] propose sampling methods for estimating derivatives of volumetric scattering parameters. Their method requires branching only once per light path and can be used with PRB to maintain linear complexity. In surface rendering, Belhe et al. [BXB*24] and Zhang et al. [ZDDZ21] propose differential and antithetic (resp.) BRDF sampling methods for estimating BRDF parameter derivatives. However, their methods require branching at every scattering event and thus have the same problem as RB—they have quadratic complexity for global illumination. We introduce a theoretical formulation and sampling method that allows using the sampling methods of Belhe et al. [BXB*24] with global illumination and PRB, maintaining linear complexity. Our method for achieving linear complexity in surface rendering bears some similarity to that of Nimier-David et al. [NDMKJ22] in volume rendering—both use a random differential scattering event—yet our method completely avoids branching. Additionally, our formulation allows MIS across entire paths, whereas theirs only locally at each scattering event. Lastly, our method is orthogonal to—and can be readily combined with—other techniques for accelerating differentiable and inverse rendering, e.g., recursive control variates [NRN*23] and caching [HZ22].

3. Differential path space integral

We differentiate the path integral formulation of light transport. In this formulation, the path space $\mathcal{P}$ comprises all light paths $\bar{\mathbf{x}} = \mathbf{x}_0\mathbf{x}_1\ldots\mathbf{x}_N$ of length $N \geq 1$ that connect the camera and light sources. The pixel value I in the rendered image equals an integral over $\mathcal{P}$:

$$I = \int_{\mathcal{P}} f(\bar{\mathbf{x}})\mathrm{d}\bar{\mathbf{x}} = \int_{\mathcal{P}} \left[\prod_{i=0}^{N} f_i(\bar{\mathbf{x}})\right]\mathrm{d}\bar{\mathbf{x}}. \tag{1}$$

The path *measurement contribution* $f(\bar{\mathbf{x}})$ is the product of measurement contributions $f_i(\bar{\mathbf{x}})$ from each path vertex, where

$$f_i(\bar{\mathbf{x}}) := \begin{cases} W_{\mathrm{e}}(\mathbf{x}_0, \omega_{\mathbf{x}_1,\mathbf{x}_0})\, G(\mathbf{x}_0,\mathbf{x}_1), & \text{if } i = 0, \\ G(\mathbf{x}_i,\mathbf{x}_{i+1})\, \rho_s(\mathbf{x}_{i-1},\mathbf{x}_i,\mathbf{x}_{i+1}), & \text{if } 0 < i < N, \\ L_{\mathrm{e}}(\mathbf{x}_N, \omega_{\mathbf{x}_N,\mathbf{x}_{N-1}}), & \text{if } i = N, \end{cases} \tag{2}$$

and: $\omega_{\mathbf{x},\mathbf{y}}$ is the unit direction from $\mathbf{x}$ to $\mathbf{y}$; W_{e} is the sensor importance; G is the geometry term; ρ_s is the BRDF; and L_{e} is the emitted radiance from the light source.

The derivative of the path integral equation (1) with respect to a

[†] https://imaging.cs.cmu.edu/path_sampling_differentiable_rendering

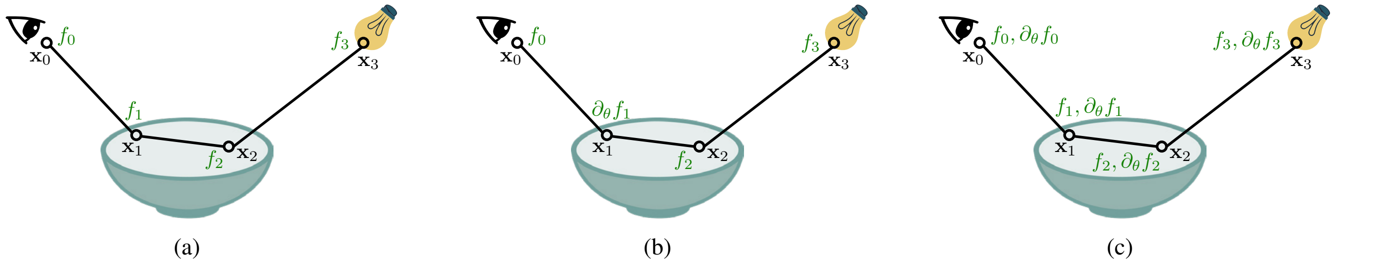

Figure 2: *Given a light path, we can compute the contribution or differential contribution from each vertex. (a) Computing the contribution at each vertex corresponds to computing the integrand of the primal rendering path integral (1). (b) Computing the differential contribution at one vertex of the path corresponds to computing the integrand of the differential path space integral (7). (c) Computing both the contribution and the differential contribution at each vertex makes it possible to compute the integrand of the path space integral (4).*

scene parameter θ that does not induce discontinuities is

$$\partial_\theta I = \int_{\mathcal{P}} \partial_\theta \left[\prod_{i=0}^{N} f_i(\bar{\mathbf{x}}) \right] \mathrm{d}\bar{\mathbf{x}} \tag{3}$$

$$\Rightarrow \quad \underbrace{\partial_\theta I = \int_{\mathcal{P}} \left[\sum_{i=0}^{N} g(\bar{\mathbf{x}}, i; \theta) \right] \mathrm{d}\bar{\mathbf{x}}}_{\text{path space integral}}, \tag{4}$$

where we define the *differential measurement contribution*:

$$g(\bar{\mathbf{x}}, i; \theta) := \left[\prod_{j=0}^{N} f_j(\bar{\mathbf{x}}) \right] \frac{\partial_\theta f_i(\bar{\mathbf{x}})}{f_i(\bar{\mathbf{x}})} = f(\bar{\mathbf{x}}) \frac{\partial_\theta f_i(\bar{\mathbf{x}})}{f_i(\bar{\mathbf{x}})}. \tag{5}$$

Each term $g(\bar{\mathbf{x}}, i; \theta)$ in the summation (4) is the same as the path measurement contribution $f(\bar{\mathbf{x}})$, except at the vertex $\mathbf{x}_i$ it replaces $f_i(\bar{\mathbf{x}})$ with its derivative $\partial_\theta f_i(\bar{\mathbf{x}})$. We call $\mathbf{x}_i$ the *differential vertex*.

Additionally, we denote by $\partial\mathcal{P}_n^N$ the set of all light paths $\bar{\mathbf{x}} = \mathbf{x}_0\mathbf{x}_1 \ldots \mathbf{x}_N$ with differential vertex $\mathbf{x}_n$ (where $n \in \{0, 1, \ldots, N\}$). We define the *differential path space* as

$$\partial\mathcal{P} := \bigcup_{N=1}^{\infty} \bigcup_{n=0}^{N} \partial\mathcal{P}_n^N. \tag{6}$$

Then, we can rewrite the path space integral (4) as:

$$\underbrace{\partial_\theta I = \int_{\partial\mathcal{P}} g(\bar{\mathbf{x}}, n; \theta) \mathrm{d}\bar{\mathbf{x}}}_{\text{differential path space integral}}. \tag{7}$$

We visualize the formulations of Equation (4) and Equation (7) in Figure 2 and will alternate between them as we explain previous work and our method. Our sampling method (Section 4.1) samples paths from the differential path space of Equation (7), but estimates the integrand of the path space integral in Equation (4).

Inverse rendering optimization. Given a loss function $\mathcal{L}(\mathbf{I}(\theta), \tilde{\mathbf{I}})$ that compares rendered images $\mathbf{I}(\theta)$ with reference images $\tilde{\mathbf{I}}$, we want to solve the optimization problem:

$$\min_\theta \mathcal{L}(\mathbf{I}(\theta), \tilde{\mathbf{I}}) \tag{8}$$

using gradient-based optimization. To do so, we need to estimate the gradient of the loss function with respect to the parameter θ. We can express this gradient as an integral over the differential path space:

$$\begin{aligned} \partial_\theta \mathcal{L} &= \partial_{\mathbf{I}} \mathcal{L} \cdot \partial_\theta \mathbf{I} & (9) \\ &= \partial_{\mathbf{I}} \mathcal{L} \cdot \int_{\partial\mathcal{P}} g(\bar{\mathbf{x}}, n; \theta) \mathrm{d}\bar{\mathbf{x}} & (10) \\ &= \int_{\partial\mathcal{P}} \partial_{\mathbf{I}} \mathcal{L} \cdot g(\bar{\mathbf{x}}, n; \theta) \mathrm{d}\bar{\mathbf{x}}. & (11) \end{aligned}$$

The image gradient $\partial_\theta \mathbf{I}$ in Equation (9) is a forward-mode derivative that captures how an infinitesimal perturbation of a scene parameter changes the rendered image. Although forward-mode derivatives are useful for visualization, they are impractical to compute during inverse rendering [NDSRJ20]. Instead, we directly compute the reverse-mode derivative $\partial_\theta \mathcal{L}$, which captures how an infinitesimal change in the scene parameter changes the image loss.

Differentiable rendering techniques typically require a separate rendering pass to estimate the gradient $\partial_{\mathbf{I}} \mathcal{L}$. (also called *adjoint radiance* [NDSRJ20]). This estimate must be uncorrelated with the image gradient $\partial_\theta \mathbf{I}$ in order for the final loss gradient $\partial_\theta \mathcal{L}$ to be unbiased [ALKN19, GZB*13]. Including $\partial_{\mathbf{I}} \mathcal{L}$ in the integral (11) [CSN*23] creates importance sampling opportunities unique to differentiable rendering [CSN*23]; these opportunities include sampling pixels based on their adjoint radiance (Section 4.2).

3.1. Relationship to previous formulations

Our formulation of the differential path integral closely relates to, and helps explain and contrast, previous works, as we explain below.

Score function estimator. Previous work [KSZ*15, GLZ16, ZWDR16, CLZ*20, STBLG20] describes a variant of our path integral formulation without the differential path space. For example, Equation (5) from Che et al. [CLZ*20] describes the same integral as ours in terms of a *path score function* S:

$$\partial_\theta I = \int_{\mathcal{P}} f(\bar{\mathbf{x}}) S(\bar{\mathbf{x}}) \mathrm{d}\bar{\mathbf{x}}, \tag{12}$$

$$S(\bar{\mathbf{x}}) := \sum_{i=0}^{N} \frac{\partial_\theta f_i(\bar{\mathbf{x}})}{f_i(\bar{\mathbf{x}})}. \tag{13}$$

This equation is equivalent to our Equations (4) and (5) when expanded. The differential path integral is usually estimated using

Monte Carlo integration, by sampling M paths from a distribution p, and summing the integrands $f(\overline{\mathbf{x}}^m)S(\overline{\mathbf{x}}^m)$ for each path $\overline{\mathbf{x}}^m$:

$$\langle \partial_\theta I \rangle \coloneqq \frac{1}{M}\sum_{m=1}^{M}\frac{f(\overline{\mathbf{x}}^m)S(\overline{\mathbf{x}}^m)}{p(\overline{\mathbf{x}}^m)}. \tag{14}$$

Computing the integrand $f(\overline{\mathbf{x}}^m)S(\overline{\mathbf{x}}^m)$ for a single path from the path space $\mathcal{P}$ produces the same result as: taking all N paths from the differential path space $\partial\mathcal{P}$ that have identical vertices but varying locations of the differential vertex, and summing their differential measurement contributions to obtain $\sum_{i=0}^{N} g(\overline{\mathbf{x}}, i; \theta)$. For the estimator of Equation (14) to have low variance, the distribution $p(\overline{\mathbf{x}})$ should be a good approximation of the integrand. The standard approach for surface rendering is to sample each path vertex proportional to the local (cosine-weighted) BRDF; this is a good strategy for primal rendering, but not necessarily for differentiable rendering, because it only targets the $f(\overline{\mathbf{x}}^m)$ term and not the $S(\overline{\mathbf{x}}^m)$ term of the integrand. Our sampling method in Section 4 corrects this issue.

Radiative backpropagation. We show in Appendix A that our path integral formulation is equivalent to the recursive formulation based on the rendering equation that Nimier-David et al. [NDSRJ20] and Vicini et al. [VSJ21] use to formulate RB and PRB, respectively. For path vertices $\mathbf{x}_i$ with $0 < i < N$, the rendering equation is

$$L_{\mathrm{o}}(\mathbf{x}_i, \omega_{\mathbf{x}_i,\mathbf{x}_{i-1}}) = \int_{\mathcal{S}^2} f_i(\overline{\mathbf{x}}) L_{\mathrm{i}}(\mathbf{x}_i, \omega_{\mathbf{x}_{i+1},\mathbf{x}_i}) \mathrm{d}\omega_{\mathbf{x}_{i+1},\mathbf{x}_i} \tag{15}$$

and its derivative is

$$\partial_\theta L_{\mathrm{o}}(\mathbf{x}_i, \omega_{\mathbf{x}_i,\mathbf{x}_{i-1}}) = \int_{\mathcal{S}^2} \Big[\partial_\theta f_i(\overline{\mathbf{x}}) L_{\mathrm{i}}(\mathbf{x}_i, \omega_{\mathbf{x}_{i+1},\mathbf{x}_i}) + f_i(\overline{\mathbf{x}}) \partial_\theta L_{\mathrm{i}}(\mathbf{x}_i, \omega_{\mathbf{x}_{i+1},\mathbf{x}_i})\Big] \mathrm{d}\omega_{\mathbf{x}_{i+1},\mathbf{x}_i}, \tag{16}$$

where L_{o} and L_{i} are outgoing and incident radiance, and we assume that light is only emitted at the end of a path.

RB estimates the loss gradient $\partial_\theta\mathcal{L}$ by first computing the adjoint radiance $\partial_{\mathbf{I}}\mathcal{L}$ and then backpropagating it through Equation (16) with recursive Monte Carlo sampling. At each path vertex, RB computes the differential measurement contribution $\partial_\theta f_i$ and the measurement contribution f_i, then combines them with recursive estimates for the incident radiance L_{i} and its gradient $\partial_\theta L_{\mathrm{i}}$ (resp.). Each recursive invocation is equivalent to "unrolling" one vertex in our path integral formulation. The disadvantage of RB is that it requires two separate recursive estimates at each intermediate path vertex to estimate both L_{i} and $\partial_\theta L_{\mathrm{i}}$, resulting in time complexity that is quadratic in the number of path vertices.

Path replay backpropagation. Though Vicini et al. [VSJ21] formulate PRB in terms of the recursive rendering equation, it is easier to explain in terms of a path integral that includes the adjoint radiance:

$$\partial_\theta\mathcal{L} = \int_{\mathcal{P}} \partial_{\mathbf{I}}\mathcal{L} \cdot \left[\sum_{i=0}^{N} g(\overline{\mathbf{x}}, i; \theta)\right] \mathrm{d}\overline{\mathbf{x}} \tag{17}$$

$$= \int_{\mathcal{P}} \sum_{i=0}^{N} \left[\partial_{\mathbf{I}}\mathcal{L} \cdot f(\overline{\mathbf{x}}) \frac{\partial_\theta f_i(\overline{\mathbf{x}})}{f_i(\overline{\mathbf{x}})}\right] \mathrm{d}\overline{\mathbf{x}}. \tag{18}$$

This integral is equivalent to Equation (11) but integrates over the path space instead of the differential path space. PRB estimates this integral by performing three rendering passes. The first pass estimates the adjoint radiance $\partial_{\mathbf{I}}\mathcal{L}$. The second pass, which is uncorrelated with the first, renders the image $\mathbf{I}$ again to precompute path measurement contributions $f(\overline{\mathbf{x}})$ for the third pass. The third pass replays the same paths as the second pass and computes the product $\partial_{\mathbf{I}}\mathcal{L} \cdot f(\overline{\mathbf{x}}) \cdot {}^{\partial_\theta f_i(\overline{\mathbf{x}})}/_{f_i(\overline{\mathbf{x}})}$ from Equation (18) at each path vertex $\mathbf{x}_i$. At the end of each path, the sum of these products forms the estimate of the loss gradient. By introducing an additional rendering pass to precompute the path measurement contributions, PRB avoids the double recursion of RB and achieves linear time complexity.

4. Methods

We now present our new sampling methods for differentiable rendering, which include importance sampling of paths from the differential path space (Section 4.1) and importance sampling of pixels using the adjoint radiance (Section 4.2).

4.1. Sampling from the differential path space

Using our formulation in Equation (7), we estimate the differential path integral by sampling light paths from the differential path space $\partial\mathcal{P}$. This sampling method is tailored for derivative estimation because the resulting paths are sampled proportionally to their differential measurement contribution g, as opposed to the standard measurement contribution f in primal rendering.

Given a path $\overline{\mathbf{x}}$ with differential vertex $\mathbf{x}_n$ from the differential path space, we can compute not only its differential measurement contribution $g(\overline{\mathbf{x}}, n; \theta)$, but also the differential measurement contribution of all corresponding paths from the normal path space. We thus compute the sum of all these contributions, that is, the integrand $\sum_{i=0}^{N} g(\overline{\mathbf{x}}, i; \theta)$ of the path space integral (4), despite sampling paths from the differential path space of Equation (7).

To sample a light path from the differential path space $\partial\mathcal{P}$, we need to determine which vertex on the path is the differential vertex. We assume that vertices at the camera and light source do not depend on the scene parameter being optimized, so only intermediate vertices may be chosen as the differential vertex.

Local sampling of differential vertex. Ideally, we would like to sample the differential vertex based on the expected differential measurement contribution of a path, which depends on the scene's global illumination. However, there is no easy way to predetermine this value, so we instead make local sampling decisions. This is similar to Russian roulette techniques for stochastically terminating paths: even though there are more optimal methods for Russian roulette that consider the expected contribution of light paths, they require iterative rendering processes [VK16, RGH*22], thus in practice Russian roulette is implemented using only local sampling.

In our setting, at each intermediate vertex $\mathbf{x}_i$, we first randomly decide with probability q whether the vertex will be the differential one. If it is, then we sample $\mathbf{x}_{i+1}$ from the differential probability density $p_{\partial f}(\mathbf{x}_i)$, weigh its contribution by ${}^1/_{p_{\partial f}(\mathbf{x}_i)q}$, and set $q = 0$ for all subsequent vertices on the path (so that there is only one differential vertex per path). Otherwise, we sample $\mathbf{x}_{i+1}$ using the standard probability density $p_f(\mathbf{x}_i)$ and weigh its contribution by ${}^1/_{p_f(\mathbf{x}_i)(1-q)}$. We write $p_f(\mathbf{x}_i)$ and $p_{\partial f}(\mathbf{x}_i)$ as shorthand for the sampling probabilities $p_f(\mathbf{x}_{i-1}\mathbf{x}_i\mathbf{x}_{i+1})$ and $p_{\partial f}(\mathbf{x}_{i-1}\mathbf{x}_i\mathbf{x}_{i+1})$, which we use to determine the location of vertex $\mathbf{x}_{i+1}$.

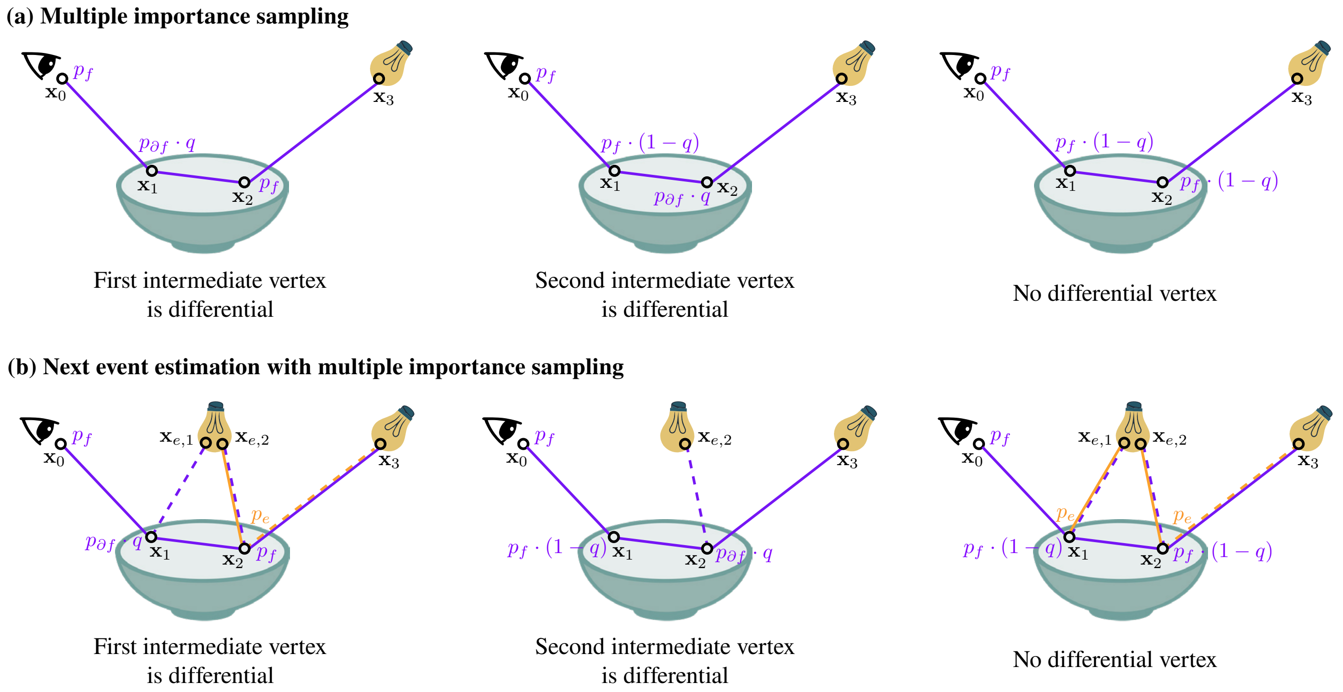

Figure 3: *To incorporate multiple importance sampling (MIS) and next event estimation (NEE), we vary the location of the differential vertex along a given path, as well as along NEE paths where the last vertex is sampled directly from the scene's emitters. Purple lines represent paths formed using BRDF or differential BRDF sampling, and orange lines represent paths formed using emitter sampling. Dashed lines represent sampling methods that are not actually sampled but are included in MIS for NEE. We obtain the probability of generating any of the paths shown by multiplying the probabilities labeled at each vertex of the path. Summing all of these path probabilities in (a) gives the mixture probability from Equation* (24)*. Summing the path probabilities for the full path $\mathbf{x}_0\mathbf{x}_1\mathbf{x}_2\mathbf{x}_3$ in (b) gives the mixture probability from Equation* (41)*, and summing them for either of the NEE paths ($\mathbf{x}_0\mathbf{x}_1\mathbf{x}_{e,1}$ or $\mathbf{x}_0\mathbf{x}_1\mathbf{x}_2\mathbf{x}_{e,2}$) in (b) gives the mixture probability from Equation* (42).

When sampling the differential vertex, we want to use a probability density $p_{\partial f}$ that is proportional to the derivative of the BRDF. Unlike the BRDF, however, the differential BRDF can have negative values, and simply sampling proportionally to its absolute value may result in high sign variance. Instead, we importance sample differential BRDFs by applying the single-signed decompositions from Belhe et al. [BXB*24]. These decompositions remove sign variance by separately sampling the positive and negative components of a function and then combining their contributions. For some of our experiments, we use the positivization technique by Zeltner et al. [ZSGJ21] to sample derivatives of microfacet BRDFs with respect to their roughness parameter. Whereas Zeltner et al. use antithetic sampling to generate samples from both the positive and negative components of the differential BRDF, we randomly choose one of the two components to sample the differential vertex from.

The probability q is a hyperparameter that influences how early we sample the differential vertex along each path. We experimented with different values of q (Section 5) and decided to use $q = 0.5$.

Multiple importance sampling. After we have sampled an entire path with a differential vertex $\mathbf{x}_n$, we can vary the location of the differential vertex to obtain other ways we could have sampled the same path (Figure 3a), thus creating an opportunity to incorporate MIS [VG95b]. Similar to how bidirectional path tracing [VG95a] uses MIS to combine sampling methods with varying camera and light subpaths, we use MIS to combine sampling methods that vary the location of the differential vertex along the path. The probability of sampling a path $\bar{\mathbf{x}} = \mathbf{x}_0\mathbf{x}_1 \ldots \mathbf{x}_N$ with differential vertex $\mathbf{x}_i$ is:

$$p(\bar{\mathbf{x}}, i) = \left[\prod_{j=0}^{N-1} p_f(\mathbf{x}_j)\right] \frac{p_{\partial f}(\mathbf{x}_i)}{p_f(\mathbf{x}_i)} (1-q)^{i-1} q. \tag{19}$$

It is also possible to sample the path without a differential vertex, an event that occurs with probability:

$$p(\bar{\mathbf{x}}) = \left[\prod_{j=0}^{N-1} p_f(\mathbf{x}_j)\right] (1-q)^{N-1}. \tag{20}$$

In total, there are N different ways in which the path could have been generated. Assuming we actually sampled the path with $\mathbf{x}_n$ as the differential vertex, the *balance-heuristic* MIS weight is

$$w(\bar{\mathbf{x}}, n) = \frac{p(\bar{\mathbf{x}}, n)}{\left[\sum_{i=1}^{N-1} p(\bar{\mathbf{x}}, i)\right] + p(\bar{\mathbf{x}})}. \tag{21}$$

We estimate the path integral of Equation (4) by sampling M paths $\{\bar{\mathbf{x}}^m\}_{m=1}^{M}$ from the differential path space in Equation (7) and com-

Algorithm 1 Path replay backpropagation with differential sampling and multiple importance sampling.

function SAMPLEPATH(`ray`)
  $L=0,\ \beta=1,\ w_1=0,\ w_2=1,$ `sampled_∂x` = FALSE
  for $i=0$ **to** $N-1$ **do**
    ▷ Accumulate radiance
    $L \mathrel{+}= \beta \cdot L_e(\dots)\ /\ (w_1+w_2)$
    ▷ Sample the BRDF or differential BRDF
    if !`sampled_∂x` **and** RAND() $< q$ **then**
      $\omega',\ f$ = SAMPLE_∂BRDF$(\dots)$
      `sampled_∂x` = TRUE
    else
      $\omega',\ f$ = SAMPLE_BRDF$(\dots)$
    ▷ Update throughput and values for MIS
    $\beta \mathrel{*}= f\ /\ p_f(\omega',\dots)$
    $w_1 \mathrel{+}= w_2 \cdot q \cdot p_{\partial f}(\omega',\dots)\ /\ p_f(\omega',\dots)$
    $w_2 \mathrel{*}= 1-q$
  return L

function SAMPLEPATHADJOINT(`ray`, L, δL)
  $\beta=1,\ w_1=0,\ w_2=1,$ `sampled_∂x` = FALSE
  for $i=0$ **to** $N-1$ **do**
    ▷ Reconstruct incident radiance
    $L \mathrel{-}= \beta \cdot L_e(\dots)\ /\ (w_1+w_2)$
    ▷ Sample the BRDF or differential BRDF
    if !`sampled_∂x` **and** RAND() $< q$ **then**
      $\omega',\ f$ = SAMPLE_∂BRDF$(\dots)$
      `sampled_∂x` = TRUE
    else
      $\omega',\ f$ = SAMPLE_BRDF$(\dots)$
    ▷ Update gradient, throughput, and values for MIS
    $\delta_\theta \mathrel{+}=$ BACKWARDGRAD$(f,\ \delta L \cdot L/f)$
    $\beta \mathrel{*}= f\ /\ p_f(\omega',\dots)$
    $w_1 \mathrel{+}= w_2 \cdot q \cdot p_{\partial f}(\omega',\dots)\ /\ p_f(\omega',\dots)$
    $w_2 \mathrel{*}= 1-q$
  return δ_θ

Algorithm 2 Inverse rendering with adaptive sampling.

while not converged **do**
  $\mathbf{I}$ = RENDER$(\dots)$ ▷ *1st pass*
  $\mathcal{L},\ \partial_{\mathbf{I}}\mathcal{L}$ = LOSS$(\mathbf{I},\ \tilde{\mathbf{I}})$
  `pixelWeights` = $|\partial_{\mathbf{I}}\mathcal{L}|\ /\ \sum|\partial_{\mathbf{I}}\mathcal{L}|$
  `rays` = SAMPLERAYS(`pixelWeights`)
  $\mathbf{I}'$ = SAMPLEPATH(`rays`) ▷ *2nd pass*
  $\partial_\theta\mathcal{L}$ = SAMPLEPATHADJOINT(`rays`, $\mathbf{I}'$, $\partial_{\mathbf{I}}\mathcal{L}$) ▷ *3rd pass*
  GRADIENTSTEP$(\partial_\theta\mathcal{L})$

bining their weighted contributions:

$$\langle \partial_\theta I \rangle^{\text{MIS}} := \frac{1}{M}\sum_{m=1}^{M} w(\overline{\mathbf{x}}^m, n^m)\frac{\sum_{i=0}^{N^m} g(\overline{\mathbf{x}}^m, i;\theta)}{p(\overline{\mathbf{x}}^m, n^m)} \tag{22}$$

$$= \frac{1}{M}\sum_{m=1}^{M}\frac{\sum_{i=0}^{N^m} g(\overline{\mathbf{x}}^m, i;\theta)}{p_{\text{mix}}(\overline{\mathbf{x}}^m)}, \tag{23}$$

where we have

$$p_{\text{mix}}(\overline{\mathbf{x}}) = \left[\prod_{i=0}^{N} p_f(\mathbf{x}_i)\right] \cdot \left[\underbrace{\left[\sum_{i=1}^{N-1}\frac{p_{\partial f}(\mathbf{x}_i)}{p_f(\mathbf{x}_i)}(1-q)^{i-1}q\right]}_{w_1} + \underbrace{(1-q)^{N-1}}_{w_2}\right]. \tag{24}$$

In Equation (22), the probability $p(\overline{\mathbf{x}}^m, n^m)$ with which we sample each path cancels out the numerator of the MIS weight (21), leaving the denominator of the estimator as a mixture probability $p_{\text{mix}}(\overline{\mathbf{x}}^m)$ that combines all possible ways we could have generated the path.

Next event estimation. Integrating our differential sampling method with next event estimation (NEE) is straightfoward, with the only complication being the derivation of the local MIS weight combining area and BRDF sampling. This complication is due to the fact that we have to consider also varying the location of the differential vertex along paths where the last vertex is sampled directly from the scene's emitters (Figure 3b). We derive the MIS weights for differential sampling with NEE in Appendix B.

Efficient implementation. As Algorithm 1 shows, we implement our sampling method using a simple modification to PRB, which outputs an estimate of differential radiance based on Equation (18). The second and third rendering passes in PRB (one primal, one adjoint) must sample the same light paths, so we use our differential sampling method for both passes. We include MIS by updating the values w_1 and w_2 (which form part of the mixture probability from Equation (24)) as we sample each path vertex, and we use them to weigh the radiance accumulated along the path. The product of densities p_f in the mixture probability (24) is included in the throughput term β. In practice, we also integrate our sampling method with next event estimation, which we omit in the pseudocode for simplicity.

4.2. Adaptive pixel sampling

In addition to importance sampling an intermediate differential vertex for each path, we can also importance sample the pixel location of its first vertex using the adjoint radiance $\partial_{\mathbf{I}}\mathcal{L}$: the adjoint radiance plays in Equation (11) a role analogous to the sensor importance in the standard path integral. After computing $\partial_{\mathbf{I}}\mathcal{L}$ from the first rendering pass in each optimization iteration, we compute a weight for each pixel that is proportional to the value of $|\partial_{\mathbf{I}}\mathcal{L}|$ at that pixel. We then use these weights to sample the sensor vertices of paths for the second and third rendering passes, which evaluate the loss gradient $\partial_\theta\mathcal{L}$. Algorithm 2 shows pseudocode for this procedure.

The adjoint radiance $\partial_{\mathbf{I}}\mathcal{L}$ has negative values, and sampling proportional to its absolute value may produce sign variance. We can remove this variance by applying positivization [BXB*24]—that is, sample the positive and negative components of $\partial_{\mathbf{I}}\mathcal{L}$ separately and then combine their estimates of the loss gradient $\partial_\theta\mathcal{L}$. In practice, we found that positivization had little effect on estimation variance. We thus avoid complicating implementation and use the simpler method of sampling proportionally to $|\partial_{\mathbf{I}}\mathcal{L}|$.

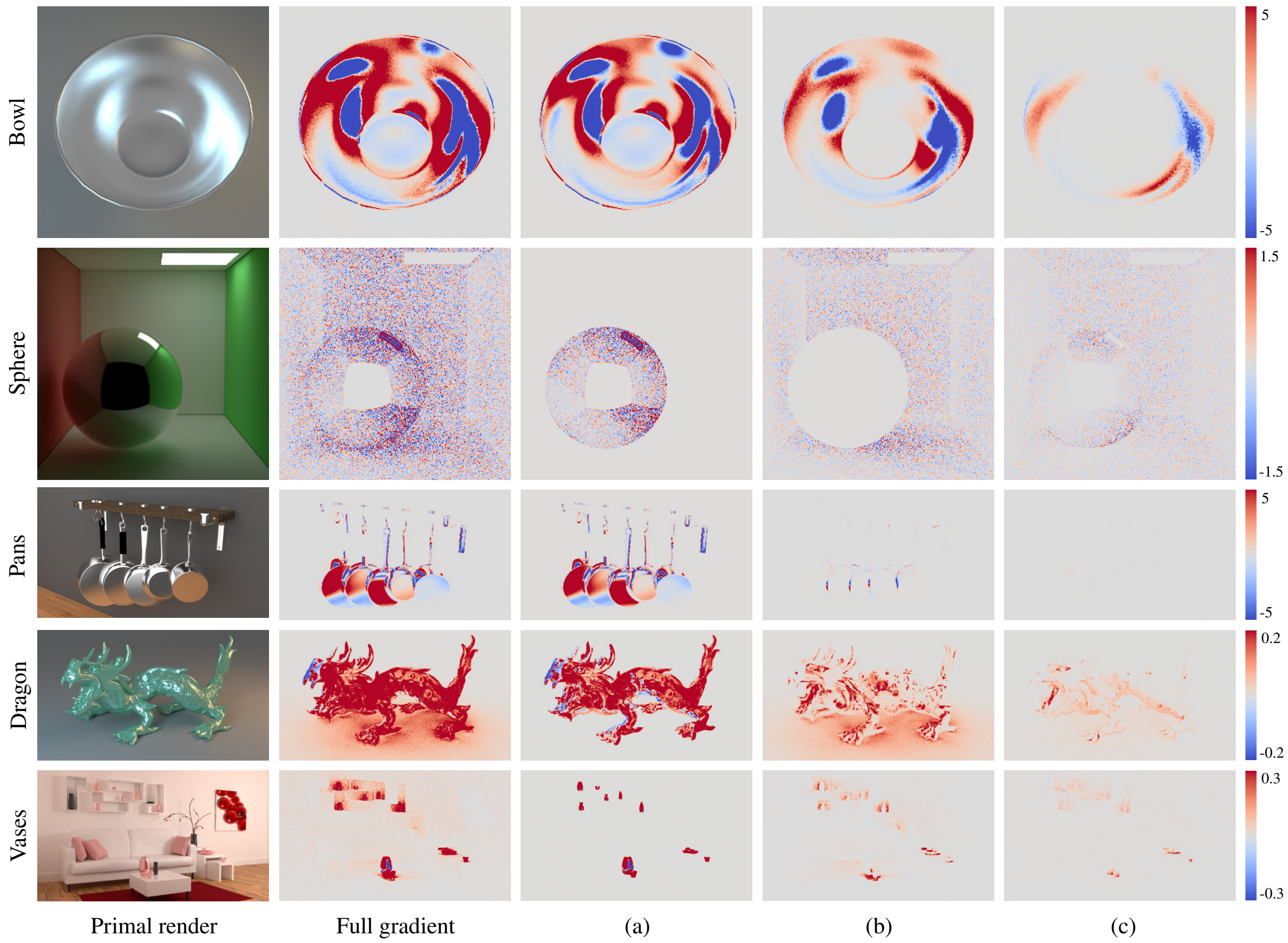

Figure 4: *Image gradients consist of contributions from multiple bounces of global illumination. We visualize the contributions of paths where the differential vertex occurs at the first (a), second (b), or third (c) intermediate vertex.*

5. Experiments

We implement our methods in Mitsuba 3 [JSR*22] and perform experiments to evaluate the impact of several hyperparameters of our proposed methods (Section 5.1). We evaluate our methods on BRDF optimization tasks by comparing their gradient variance (Section 5.2) and inverse rendering performance (Section 5.3) with standard path tracing that uses primal BRDF sampling (with and without NEE). All our comparisons are *equal-time comparisons*. We provide our code on the project website.

Figure 4 shows scenes we use for our experiments. The birds-eye-view bowl, sphere in a Cornell box, and pans use isotropic Beckmann microfacet BRDFs; we compute gradients with respect to the roughness of the microfacet distribution. The dragon and the living room vases use a mixture BRDF with diffuse and specular (isotropic GGX microfacet) lobes; we compute gradients with respect to the mixture weight. All scenes use area lights and contain significant interreflections. To highlight the importance of global illumination for differential light transport, the figure shows separately image gradient contributions from paths with differential vertex at the first, second, or third intermediate vertex.

5.1. Setting hyperparameters

Probability of sampling differentially. We experiment with the hyperparameter q (Section 4.1) that influences how early the differential vertex is sampled. Intuitively, we want the differential vertex to occur early (higher value of q) and thus have a large weight in the path contribution. However, we also want to consider later occurrences of the differential vertex to include differential contributions from indirect lighting. We find that $q = 0.5$ strikes a balance between the two and generally leads to the lowest variance (Figure 5).

Sample budget for adaptive sampling. In Table 1, we experiment with the proportion of samples used for evaluating the primal image (first rendering pass) versus the loss gradient (second and third rendering passes). We estimate the variance of loss gradients as follows: we set the parameter to some initial value different from the reference, render the primal image to compute per-pixel sam-

Proceedings published by Eurographics - The European Association for Computer Graphics.

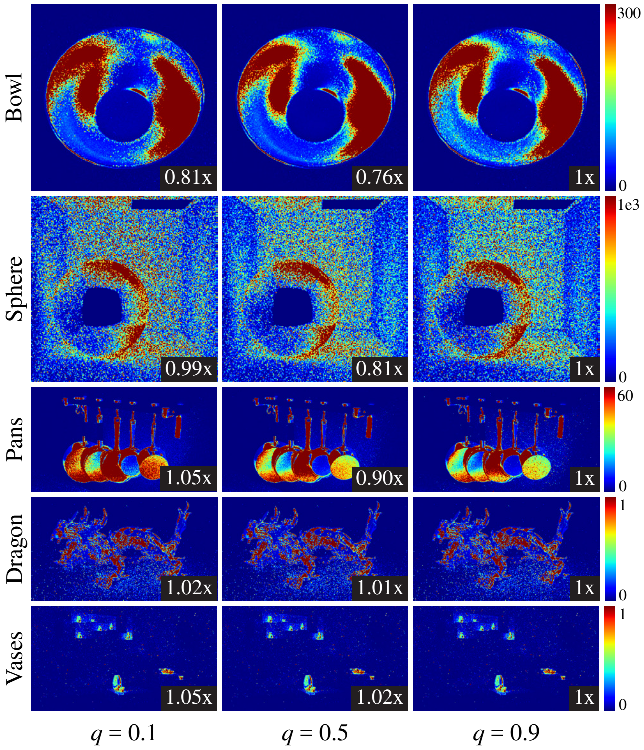

Figure 5: *We vary the value of the hyperparameter q and visualize the per-pixel variance of image gradients estimated using differential sampling with NEE and MIS. Numbers represent ratios of mean variance with the last column, lower is better.*

pling weights, use these weights to estimate the loss gradient 100 times, and then compute the sample variance of the 100 estimates. Because we use the primal image to compute the per-pixel weights for adaptive sampling, using more samples for the first rendering pass produces better sampling weights. We find that the variance of loss gradients is lowest when we allocate 75% of the sample budget for the primal phase and 25% of the sample budget for the adjoint phase; we use this ratio for the rest of our experiments.

5.2. Variance reduction

Variance of image gradients. We evaluate the impact of differential sampling, MIS, and NEE on the per-pixel variance of image gradients in Figure 6. We estimate this variance by rendering the image gradient 100 times and computing the sample variance for each pixel across the 100 estimates. With all features combined, our method yields significant variance reduction compared to BRDF sampling—with NEE, our method reduces variance to $0.69--0.84\times$ for the microfacet BRDF, and $0.0048--0.042\times$ for the mixture BRDF. We also estimate variance for microfacet BRDFs with varying roughness values in Figure 7. We find that our method leads to greater variance reduction for more specular (lower roughness) BRDFs.

Variance of loss gradients. To evaluate our adaptive sampling method, we estimate the variance of loss gradients and report them in Table 2. The experiment setup is the same as in Table 1 except

Table 1: *We report variance of loss gradients with varying sample budgets for the primal image versus the loss gradient, lower is better. We estimate gradients using adaptive sampling with differential sampling, MIS, and NEE. Sample ratios are primal : adjoint.*

Scene	1 : 3	2 : 2	3 : 1
Bowl	6.18	3.79	3.49
Sphere	25.6	13.1	8.62
Pans	1.01	0.967	0.412
Dragon	0.0171	0.0141	0.0109
Vases	2.29×10^{-5}	2.54×10^{-5}	1.31×10^{-5}

Table 2: *We report variance of loss gradients with and without adaptive sampling and differential sampling, lower is better. We estimate the gradients with MIS and NEE.*

Scene	BRDF	BRDF + adaptive	Differential + adaptive
Bowl	13.6	4.34	3.49
Sphere	172	51.7	8.62
Pans	4.32	0.487	0.412
Dragon	4.76	0.633	0.0109
Vases	0.00164	0.000193	1.31×10^{-5}

that we do not compute sampling weights for the baseline method (BRDF sampling), and we use a consistent sample budget ratio of 3:1. Figure 8 shows the per-pixel weights. We find that our combined differential and adaptive sampling method reduces variance by 1–2 orders of magnitude for the five scenes that tested.

5.3. Inverse rendering

Our method's improvement in gradient estimation leads to improved inverse rendering performance. In Figure 1 and Figure 9b, we optimize the mixture weight of a BRDF with diffuse and specular lobes. In Figure 9a, we optimize the roughness of an isotropic Beckmann microfacet BRDF. For all inverse rendering results, we render the primal image using 3–6 samples per pixel, and we estimate the loss gradient using 1–2 samples per pixel. We use the Adam optimizer [KB14] with the image's relative mean square error as the loss function. Our method improves convergence in all cases, although the improvement varies for different BRDFs. For example, the "Dragon" scene shows greater improvement with the mixture BRDF (Figure 1) than with the microfacet BRDF (Figure 9a). We can understand this difference, which is consistent with previous work [BXB*24], as follows: improvements from our method are more significant for BRDFs whose differential and primal probability density functions are very different. If the two are similar, then our method gives similar results as path tracing with primal BRDF sampling. In our examples, the differential versus primal difference is larger for the mixture BRDF than for the microfacet BRDF.

6. Conclusion

We introduced methods that importance sample paths for differentiable rendering of scene parameters that do not induce visibility-driven discontinuities. Our methods incorporate recent differential

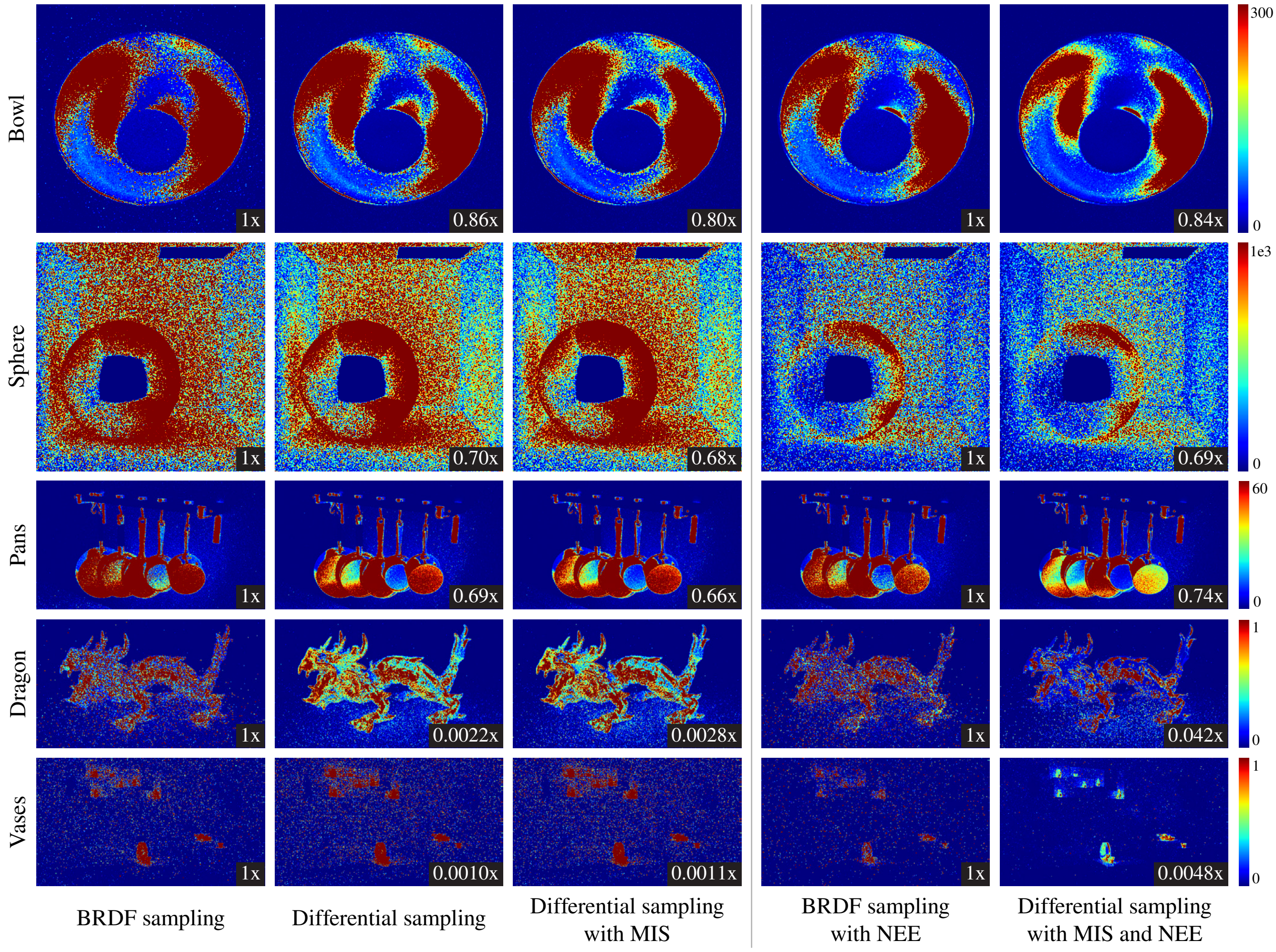

Figure 6: *We visualize the per-pixel variance of image gradients with and without differential sampling, multiple importance sampling (MIS), and next event estimation (NEE). Numbers represent ratios of mean variance between our method and BRDF sampling, lower is better.*

BRDF sampling routines [ZSGJ21, BXB*24] with a new differential path integral formulation, to achieve linear complexity and enable efficient implementation through PRB. Additionally, our methods incorporate new MIS and adaptive sampling procedures tailored for differentiable and inverse rendering applications.

Our paper takes only first steps towards developing path sampling methods for differentiable rendering. Thus, our methods are still limited in a few important ways, all of which suggest directions for future exploration. First, we have demonstrated our methods only for differentiation and optimization with respect to BRDF parameters, though our theory should apply for other parameters that do not induce visibility-driven discontinuities, e.g., local shading and normals, texture-space parameters [NDDJK21], and scattering materials [GLZ16, CLZ*20]. The latter case would require handling derivatives of transmittance terms [NDMKJ22] and null-scattering path integral formulations [MGJ19].

Second, our methods sample paths with high derivative contributions for a specific scene parameter, but such paths are typically suboptimal when computing derivatives for other scene parameters. Making our methods practical for inverse rendering problems that optimize more than one scene parameter could be done using ideas from spectral sampling, e.g., using a "hero" parameter [WND*14].

Third, our methods are specific to parameters that do not introduce visibility-driven discontinuities. Developing path sampling techniques for parameters that create such discontinuities, and thus necessitate computing boundary integrals [ZMY*20], remains a significant challenge and fascinating future research direction.

Acknowledgements. We thank Yash Belhe for giving us early access to code for the differential BRDF sampling methods in Belhe et al. [BXB*24]. The "Pans" and "Sphere" scenes are by Zhang et al. [ZDDZ21], the "Dragon" scene by Nicolet et al. [NRN*23], and the "Vases" scene by user Wig42 on Blend Swap. This work was supported by NSF awards 1900849, 2008123, NSF Graduate Research Fellowship DGE2140739, and a Sloan Research Fellowship.

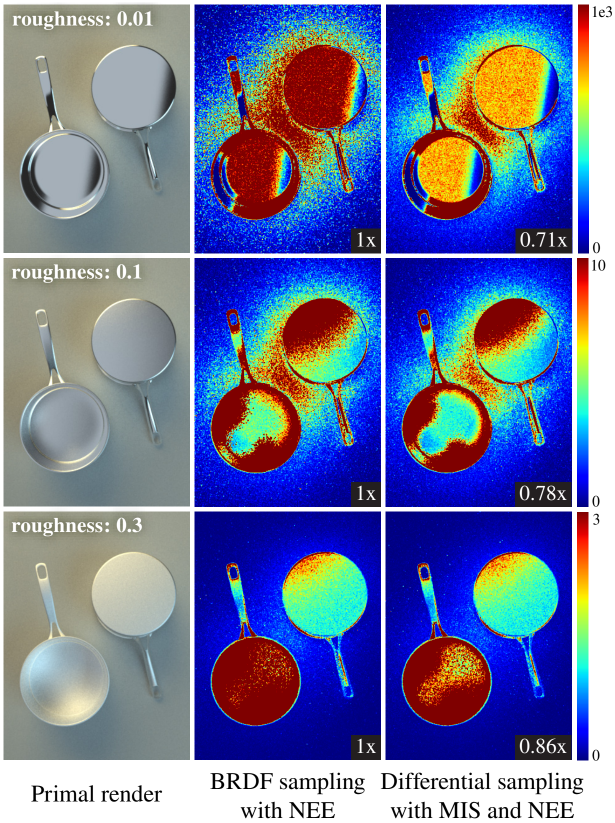

Figure 7: *We visualize the per-pixel variance of image gradients for a scene with varying roughnesses. Numbers are ratios of mean variance between our method and BRDF sampling, lower is better.*

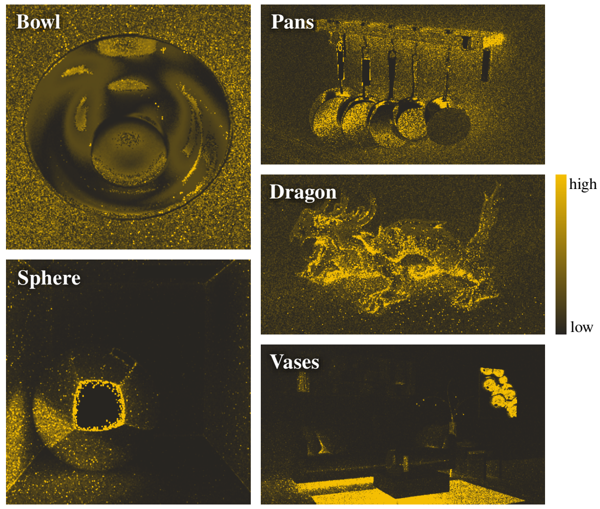

Figure 8: *We visualize the per-pixel weights used for adaptive sampling at one iteration during inverse rendering.*

References

[ALKN19] AZINOVIC D., LI T.-M., KAPLANYAN A., NIESSNER M.: Inverse path tracing for joint material and lighting estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2019), pp. 2447–2456. 3

[BGL*22] BANGARU S. P., GHARBI M., LUAN F., LI T.-M., SUNKAVALLI K., HASAN M., BI S., XU Z., BERNSTEIN G., DURAND F.: Differentiable rendering of neural SDFs through reparameterization. In *SIGGRAPH Asia 2022 Conference Papers* (2022), pp. 1–9. 2

[BLD20] BANGARU S. P., LI T.-M., DURAND F.: Unbiased warped-area sampling for differentiable rendering. *ACM Transactions on Graphics (TOG) 39*, 6 (2020), 1–18. 2

[BXB*24] BELHE Y., XU B., BANGARU S. P., RAMAMOORTHI R., LI T.-M.: Importance sampling BRDF derivatives. *ACM Transactions on Graphics* (2024). 2, 5, 6, 8, 9

[CLZ*20] CHE C., LUAN F., ZHAO S., BALA K., GKIOULEKAS I.: Towards learning-based inverse subsurface scattering. In *2020 IEEE International Conference on Computational Photography (ICCP)* (2020), IEEE, pp. 1–12. 2, 3, 9

[CSN*23] CHANG W., SIVARAM V., NOWROUZEZAHRAI D., HACHISUKA T., RAMAMOORTHI R., LI T.-M.: Parameter-space ReSTIR for differentiable and inverse rendering. In *ACM SIGGRAPH 2023 Conference Proceedings* (2023), pp. 1–10. 3

[GLZ16] GKIOULEKAS I., LEVIN A., ZICKLER T.: An evaluation of computational imaging techniques for heterogeneous inverse scattering. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part III 14* (2016), Springer, pp. 685–701. 2, 3, 9

[GZB*13] GKIOULEKAS I., ZHAO S., BALA K., ZICKLER T., LEVIN A.: Inverse volume rendering with material dictionaries. *ACM Transactions on Graphics (TOG) 32*, 6 (2013), 1–13. 1, 3

[HHM22] HASSELGREN J., HOFMANN N., MUNKBERG J.: Shape, light, and material decomposition from images using Monte Carlo rendering and denoising. *Advances in Neural Information Processing Systems 35* (2022), 22856–22869. 2

[HZ22] HADADAN S., ZWICKER M.: Differentiable neural radiosity. *arXiv preprint arXiv:2201.13190* (2022). 2

[JSR*22] JAKOB W., SPEIERER S., ROUSSEL N., NIMIER-DAVID M., VICINI D., ZELTNER T., NICOLET B., CRESPO M., LEROY V., ZHANG Z.: Mitsuba 3 renderer, 2022. https://mitsuba-renderer.org. 7

[KB14] KINGMA D. P., BA J.: Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014). 8

[KSZ*15] KHUNGURN P., SCHROEDER D., ZHAO S., BALA K., MARSCHNER S.: Matching real fabrics with micro-appearance models. *ACM Trans. Graph. 35*, 1 (2015), 1–1. 2, 3

[LADL18] LI T.-M., AITTALA M., DURAND F., LEHTINEN J.: Differentiable Monte Carlo ray tracing through edge sampling. *ACM Transactions on Graphics (TOG) 37*, 6 (2018), 1–11. 1, 2

[LHJ19] LOUBET G., HOLZSCHUCH N., JAKOB W.: Reparameterizing discontinuous integrands for differentiable rendering. *ACM Transactions on Graphics (TOG) 38*, 6 (2019), 1–14. 2

[MGJ19] MILLER B., GEORGIEV I., JAROSZ W.: A null-scattering path integral formulation of light transport. *ACM Transactions on Graphics (TOG) 38*, 4 (2019), 1–13. 9

[NDDJK21] NIMIER-DAVID M., DONG Z., JAKOB W., KAPLANYAN A.: Material and lighting reconstruction for complex indoor scenes with texture-space differentiable rendering. 9

[NDMKJ22] NIMIER-DAVID M., MÜLLER T., KELLER A., JAKOB W.: Unbiased inverse volume rendering with differential trackers. *ACM Transactions on Graphics (TOG) 41*, 4 (2022), 1–20. 2, 9

[NDSRJ20] NIMIER-DAVID M., SPEIERER S., RUIZ B., JAKOB W.: Radiative backpropagation: An adjoint method for lightning-fast differentiable rendering. *ACM Transactions on Graphics (TOG) 39*, 4 (2020), 146–1. 2, 3, 4

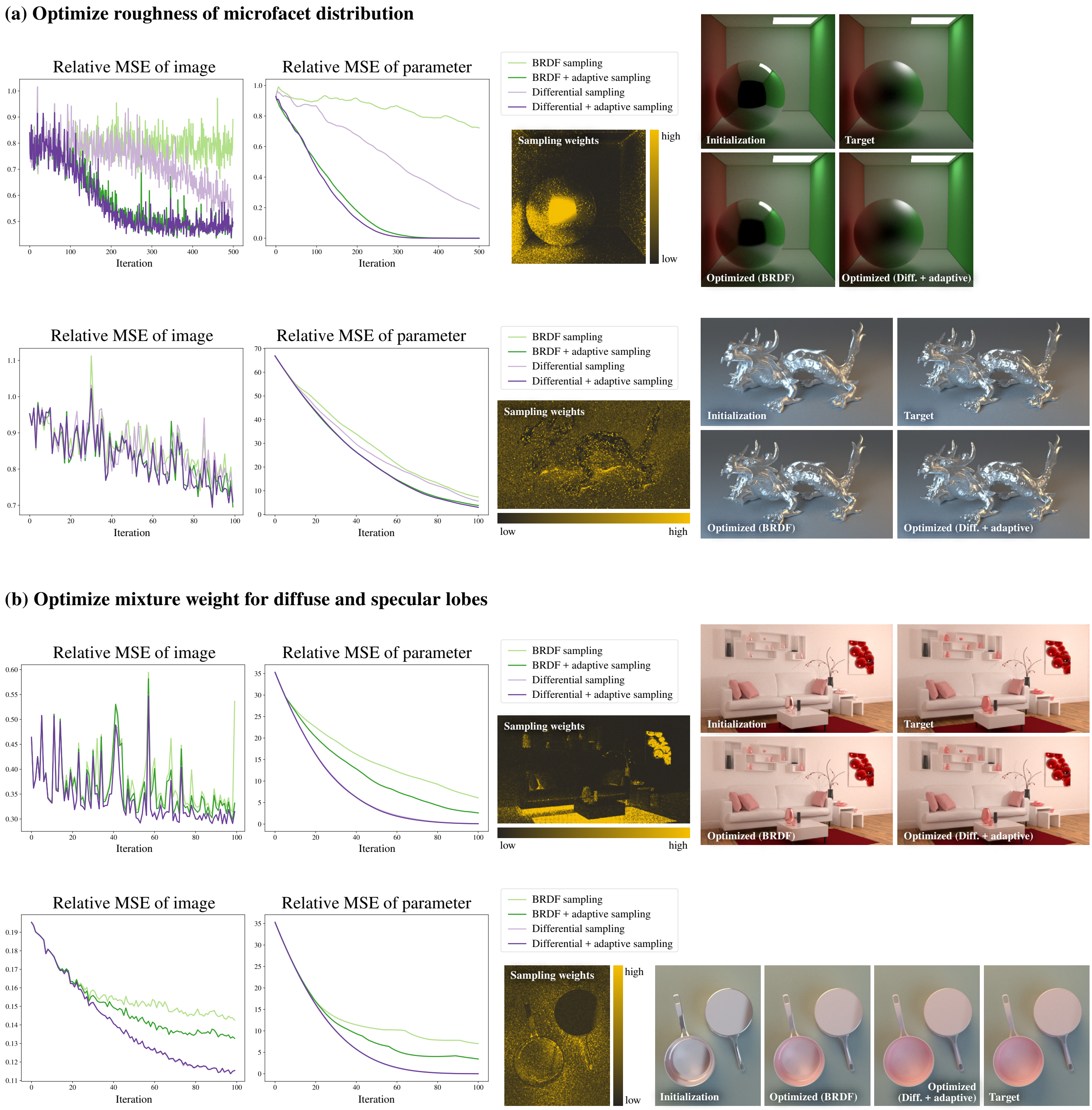

Figure 9: *We perform inverse rendering to optimize a microfacet BRDF (a) and a mixture BRDF (b). Our path sampling methods result in better convergence than BRDF sampling. We use only the relative image loss for optimization, but show also the parameter loss to assess convergence. The per-pixel sampling weights change in each iteration of gradient descent; we visualize weights from one of the iterations as an example. In (b), the differential sampling and the combined differential-adaptive sampling methods have nearly identical performance.*

[NDVZJ19] NIMIER-DAVID M., VICINI D., ZELTNER T., JAKOB W.: Mitsuba 2: A retargetable forward and inverse renderer. *ACM Transactions on Graphics (TOG) 38*, 6 (2019), 1–17. 2

[NRN*23] NICOLET B., ROUSSELLE F., NOVAK J., KELLER A., JAKOB W., MÜLLER T.: Recursive control variates for inverse rendering. *ACM Transactions on Graphics (TOG) 42*, 4 (2023), 1–13. 2, 9

[RGH*22] RATH A., GRITTMANN P., HERHOLZ S., WEIER P., SLUSALLEK P.: EARS: Efficiency-aware Russian roulette and splitting. *ACM Transactions on Graphics (TOG) 41*, 4 (2022), 1–14. 4

[STBLG20] SHEM-TOV K., BANGARU S. P., LEVIN A., GKIOULEKAS I.: Towards reflectometry from interreflections. In *2020 IEEE International Conference on Computational Photography (ICCP)* (2020), IEEE, pp. 1–12. 3

[VG95a] VEACH E., GUIBAS L.: Bidirectional estimators for light transport. In *Photorealistic Rendering Techniques*. Springer, 1995, pp. 145–167. 5

[VG95b] VEACH E., GUIBAS L. J.: Optimally combining sampling techniques for Monte Carlo rendering. In *Proceedings of the 22nd annual conference on Computer graphics and interactive techniques* (1995), pp. 419–428. 5

[VK16] VORBA J., KŘIVÁNEK J.: Adjoint-driven Russian roulette and splitting in light transport simulation. *ACM Transactions on Graphics (TOG) 35*, 4 (2016), 1–11. 4

[VSJ21] VICINI D., SPEIERER S., JAKOB W.: Path replay backpropagation: Differentiating light paths using constant memory and linear time. *ACM Transactions on Graphics (TOG) 40*, 4 (2021), 1–14. 2, 4

[VSJ22] VICINI D., SPEIERER S., JAKOB W.: Differentiable signed distance function rendering. *ACM Transactions on Graphics (TOG) 41*, 4 (2022), 1–18. 2

[WND*14] WILKIE A., NAWAZ S., DROSKE M., WEIDLICH A., HANIKA J.: Hero wavelength spectral sampling. In *Computer Graphics Forum* (2014), vol. 33, Wiley Online Library, pp. 123–131. 9

[XBLZ23] XU P., BANGARU S., LI T.-M., ZHAO S.: Warped-area reparameterization of differential path integrals. *ACM Transactions on Graphics (TOG) 42*, 6 (2023), 1–18. 2

[ZDDZ21] ZHANG C., DONG Z., DOGGETT M., ZHAO S.: Antithetic sampling for Monte Carlo differentiable rendering. *ACM Transactions on Graphics (TOG) 40*, 4 (2021), 1–12. 2, 9

[ZMY*20] ZHANG C., MILLER B., YAN K., GKIOULEKAS I., ZHAO S.: Path-space differentiable rendering. *ACM Trans. Graph. 39*, 4 (2020), 143:1–143:19. 2, 9

[ZSGJ21] ZELTNER T., SPEIERER S., GEORGIEV I., JAKOB W.: Monte Carlo estimators for differential light transport. *ACM Transactions on Graphics (TOG) 40*, 4 (2021), 1–16. 2, 5, 9

[ZWDR16] ZHAO S., WU L., DURAND F., RAMAMOORTHI R.: Downsampling scattering parameters for rendering anisotropic media. *ACM Transactions on Graphics (TOG) 35*, 6 (2016), 1–11. 2, 3

[ZWZ*19] ZHANG C., WU L., ZHENG C., GKIOULEKAS I., RAMAMOORTHI R., ZHAO S.: A differential theory of radiative transfer. *ACM Transactions on Graphics (TOG) 38*, 6 (2019), 1–16. 2

Appendix A: Equivalence of recursive and path-based formulations of differential light transport.

The value I of each pixel in the rendered image equals:

$$\begin{aligned} I &= \int_{\mathcal{S}^2} W_e(\mathbf{x}_0, \omega_{\mathbf{x}_1,\mathbf{x}_0}) G(\mathbf{x}_0, \mathbf{x}_1) L_i(\mathbf{x}_0, \omega_{\mathbf{x}_1,\mathbf{x}_0}) \mathrm{d}\omega_{\mathbf{x}_1,\mathbf{x}_0} && (25)\\ &= \int_{\mathcal{S}^2} f_0(\bar{\mathbf{x}}) L_i(\mathbf{x}_0, \omega_{\mathbf{x}_1,\mathbf{x}_0}) \mathrm{d}\omega_{\mathbf{x}_1,\mathbf{x}_0}. && (26)\end{aligned}$$

For all intermediate vertices on a path, we have:

$$L_i(\mathbf{x}_i, \omega_{\mathbf{x}_{i+1},\mathbf{x}_i}) = L_o(\mathbf{x}_{i+1}, \omega_{\mathbf{x}_{i+1},\mathbf{x}_i}), \tag{27}$$

and at the end of the path, we have

$$L_o(\mathbf{x}_N, \omega_{\mathbf{x}_N,\mathbf{x}_{N-1}}) = L_e(\mathbf{x}_N, \omega_{\mathbf{x}_N,\mathbf{x}_{N-1}}) = f_N(\bar{\mathbf{x}}). \tag{28}$$

We differentiate Equation (26) and recursively expand using the rendering equation (15), its derivative (16), and Equations (27)–(28):

$$\begin{aligned}
\partial_\theta I &= \partial_\theta \left[\int_{\mathcal{S}^2} f_0(\bar{\mathbf{x}}) L_i(\mathbf{x}_0, \omega_{\mathbf{x}_1,\mathbf{x}_0}) \mathrm{d}\omega_{\mathbf{x}_1,\mathbf{x}_0} \right] && (29)\\
&= \int_{\mathcal{S}^2} \partial_\theta \left[f_0(\bar{\mathbf{x}}) L_o(\mathbf{x}_1, \omega_{\mathbf{x}_1,\mathbf{x}_0}) \right] \mathrm{d}\omega_{\mathbf{x}_1,\mathbf{x}_0} && (30)\\
&= \int_{\mathcal{S}^2} \left[\partial_\theta f_0(\bar{\mathbf{x}}) L_o(\mathbf{x}_1, \omega_{\mathbf{x}_1,\mathbf{x}_0}) + f_0(\bar{\mathbf{x}}) \partial_\theta L_o(\mathbf{x}_1, \omega_{\mathbf{x}_1,\mathbf{x}_0}) \right] \mathrm{d}\omega_{\mathbf{x}_1,\mathbf{x}_0} && (31)\\
&= \int_{\mathcal{S}^2} \Big[\partial_\theta f_0(\bar{\mathbf{x}}) \int_{\mathcal{S}^2} f_1(\bar{\mathbf{x}}) L_i(\mathbf{x}_1, \omega_{\mathbf{x}_2,\mathbf{x}_1}) \mathrm{d}\omega_{\mathbf{x}_2,\mathbf{x}_1} \\
&\qquad + f_0(\bar{\mathbf{x}}) \int_{\mathcal{S}^2} \Big[\partial_\theta f_1(\bar{\mathbf{x}}) L_i(\mathbf{x}_1, \omega_{\mathbf{x}_2,\mathbf{x}_1}) \\
&\qquad\qquad + f_1(\bar{\mathbf{x}}) \partial_\theta L_i(\mathbf{x}_1, \omega_{\mathbf{x}_2,\mathbf{x}_1}) \Big] \mathrm{d}\omega_{\mathbf{x}_2,\mathbf{x}_1} \Big] \mathrm{d}\omega_{\mathbf{x}_1,\mathbf{x}_0} && (32)\\
&= \int_{\mathcal{S}^2} \int_{\mathcal{S}^2} \Big[\partial_\theta f_0(\bar{\mathbf{x}}) f_1(\bar{\mathbf{x}}) L_o(\mathbf{x}_2, \omega_{\mathbf{x}_2,\mathbf{x}_1}) \\
&\qquad + f_0(\bar{\mathbf{x}}) \partial_\theta f_1(\bar{\mathbf{x}}) L_o(\mathbf{x}_2, \omega_{\mathbf{x}_2,\mathbf{x}_1}) \\
&\qquad + f_0(\bar{\mathbf{x}}) f_1(\bar{\mathbf{x}}) \partial_\theta L_o(\mathbf{x}_2, \omega_{\mathbf{x}_2,\mathbf{x}_1}) \Big] \mathrm{d}\omega_{\mathbf{x}_2,\mathbf{x}_1} \mathrm{d}\omega_{\mathbf{x}_1,\mathbf{x}_0} && (33)\\
&= \ldots && (34)\\
&= \int_{\mathcal{S}^2} \cdots \int_{\mathcal{S}^2} \Big[\partial_\theta f_0(\bar{\mathbf{x}}) \Big[\prod_{i=1}^{N} f_i(\bar{\mathbf{x}}) \Big] + f_0(\bar{\mathbf{x}}) \partial_\theta f_1(\bar{\mathbf{x}}) \Big[\prod_{i=2}^{N} f_i(\bar{\mathbf{x}}) \Big] \\
&\qquad + \ldots + \Big[\prod_{i=0}^{N-1} f_i(\bar{\mathbf{x}}) \Big] \partial_\theta f_N(\bar{\mathbf{x}}) \Big] \mathrm{d}\omega_{\mathbf{x}_N,\mathbf{x}_{N-1}} \ldots \mathrm{d}\omega_{\mathbf{x}_1,\mathbf{x}_0} && (35)\\
&= \int_{\mathcal{P}} \left[\sum_{i=0}^{N} g(\bar{\mathbf{x}}, i; \theta) \right] \mathrm{d}\bar{\mathbf{x}}. && (36)
\end{aligned}$$

This result is the same as the path space integral (4).

Appendix B: Derivation of multiple importance sampling weights for next event estimation.

For a path $\mathbf{x}_0\mathbf{x}_1 \ldots \mathbf{x}_j\mathbf{x}_{e,j}$ where the first $j+1$ vertices are from the path $\bar{\mathbf{x}}$ but the last vertex $\mathbf{x}_{e,j}$ is sampled from the emitters with probability $p_e(\mathbf{x}_j\mathbf{x}_{e,j})$, the probability of generating the path with differential vertex $\mathbf{x}_i$ (where $1 \leq i < j$) is:

$$p_{\text{nee}}(\bar{\mathbf{x}},\, \mathbf{x}_j\mathbf{x}_{e,j},\, i) = \left[\prod_{k=0}^{j-1} p_f(\mathbf{x}_k) \right] p_e(\mathbf{x}_j\mathbf{x}_{e,j}) \frac{p_{\partial f}(\mathbf{x}_i)}{p_f(\mathbf{x}_i)} (1-q)^{i-1} q. \tag{37}$$

Generating this path without a differential vertex has probability:

$$p_{\text{nee}}(\bar{\mathbf{x}},\, \mathbf{x}_j\mathbf{x}_{e,j}) = \left[\prod_{k=0}^{j-1} p_f(\mathbf{x}_k) \right] p_e(\mathbf{x}_j\mathbf{x}_{e,j}) (1-q)^{j-1}. \tag{38}$$

We could also generate the paths above using BRDF or differential sampling for $\mathbf{x}_{e,j}$, which would result in the following probabilities:

$$p(\bar{\mathbf{x}},\, \mathbf{x}_j\mathbf{x}_{e,j},\, i) = \begin{cases} p_{\text{nee}}(\bar{\mathbf{x}},\, \mathbf{x}_j\mathbf{x}_{e,j},\, i) \dfrac{p_f(\mathbf{x}_{j-1}\mathbf{x}_j\mathbf{x}_{e,j})}{p_e(\mathbf{x}_j\mathbf{x}_{e,j})}, & \text{if } 1 \leq i < j, \\ p_{\text{nee}}(\bar{\mathbf{x}},\, \mathbf{x}_j\mathbf{x}_{e,j}) \dfrac{p_{\partial f}(\mathbf{x}_{j-1}\mathbf{x}_j\mathbf{x}_{e,j})}{p_e(\mathbf{x}_j\mathbf{x}_{e,j})}, & \text{if } i = j, \end{cases} \tag{39}$$

$$p(\bar{\mathbf{x}},\, \mathbf{x}_j\mathbf{x}_{e,j}) = p_{\text{nee}}(\bar{\mathbf{x}},\, \mathbf{x}_j\mathbf{x}_{e,j}) \frac{p_f(\mathbf{x}_{j-1}\mathbf{x}_j\mathbf{x}_{e,j})}{p_e(\mathbf{x}_j\mathbf{x}_{e,j})}. \tag{40}$$

By considering both BRDF sampling and emitter sampling with varying locations of the differential vertex, we obtain the following mixture probabilities for generating the paths $\overline{\mathbf{x}}$ and $\mathbf{x}_0\mathbf{x}_1\ldots\mathbf{x}_j\mathbf{x}_{e,j}$:

$$p_{\mathrm{mix}}(\overline{\mathbf{x}}) = p(\overline{\mathbf{x}}) + \left[\sum_{i=1}^{N-1} p(\overline{\mathbf{x}}, i)\right] + p_{\mathrm{nee}}(\overline{\mathbf{x}}, \mathbf{x}_{N-1}\mathbf{x}_N) + \left[\sum_{i=1}^{N-1} p_{\mathrm{nee}}(\overline{\mathbf{x}}, \mathbf{x}_{N-1}\mathbf{x}_N, i)\right], \tag{41}$$

$$p_{\mathrm{mix}}(\overline{\mathbf{x}},\, \mathbf{x}_j\mathbf{x}_{e,j}) = p(\overline{\mathbf{x}},\, \mathbf{x}_j\mathbf{x}_{e,j}) + \left[\sum_{i=1}^{j} p(\overline{\mathbf{x}},\, \mathbf{x}_j\mathbf{x}_{e,j},\, i)\right] + p_{\mathrm{nee}}(\overline{\mathbf{x}},\, \mathbf{x}_j\mathbf{x}_{e,j}) + \left[\sum_{i=1}^{j-1} p_{\mathrm{nee}}(\overline{\mathbf{x}},\, \mathbf{x}_j\mathbf{x}_{e,j},\, i)\right].. \tag{42}$$

The differential measurement contribution of the path $\mathbf{x}_0\mathbf{x}_1\ldots\mathbf{x}_j\mathbf{x}_{e,j}$ with differential vertex $\mathbf{x}_i$ is the same as $g(\overline{\mathbf{x}}, i; \theta)$ from Equation (5), except that we need to adjust the contributions at the end of the path:

$$g(\overline{\mathbf{x}},\, \mathbf{x}_j\mathbf{x}_{e,j},\, i;\, \theta) :- \left[\prod_{k=1}^{j-1} f_k(\overline{\mathbf{x}})\right] G(\mathbf{x}_j, \mathbf{x}_{e,j})\, \rho_s(\mathbf{x}_{j-1}, \mathbf{x}_j, \mathbf{x}_{e,j}) \cdot L_e(\mathbf{x}_{e,j}) \frac{\partial_\theta f_i(\overline{\mathbf{x}})}{f_i(\overline{\mathbf{x}})}. \tag{43}$$

The final estimator for the differential path integral with MIS and NEE combines contributions from the standard path $\overline{\mathbf{x}}$ and paths $\left\{\mathbf{x}_0\mathbf{x}_1\ldots\mathbf{x}_j\mathbf{x}_{e,j}\right\}_{j=1}^{N-1}$ that are directly connected to an emitter:

$$\left\langle \partial_\theta I \right\rangle^{\mathrm{NEE}} := \frac{1}{M}\sum_{m=1}^{M}\left[\frac{\sum_{i=0}^{N^m} g(\overline{\mathbf{x}}^m, i;\, \theta)}{p_{\mathrm{mix}}(\overline{\mathbf{x}}^m)} + \sum_{j=1}^{N^m-1} \frac{\sum_{i=0}^{j} g(\overline{\mathbf{x}}^m,\, \mathbf{x}^m_j\mathbf{x}^m_{e,j},\, i;\, \theta)}{p_{\mathrm{mix}}(\overline{\mathbf{x}}^m,\, \mathbf{x}^m_j\mathbf{x}^m_{e,j})}\right]. \tag{44}$$