Robust Computation of Boundary Path Integrals Using Kernel-Density Estimation

PEIYU XU, University of Illinois Urbana-Champaign, USA
LIFAN WU, NVIDIA, USA
BENEDIKT BITTERLI, NVIDIA, USA
RAVI RAMAMOORTHI, NVIDIA, USA and University of California, San Diego, USA
SHUANG ZHAO, University of Illinois Urbana-Champaign, USA

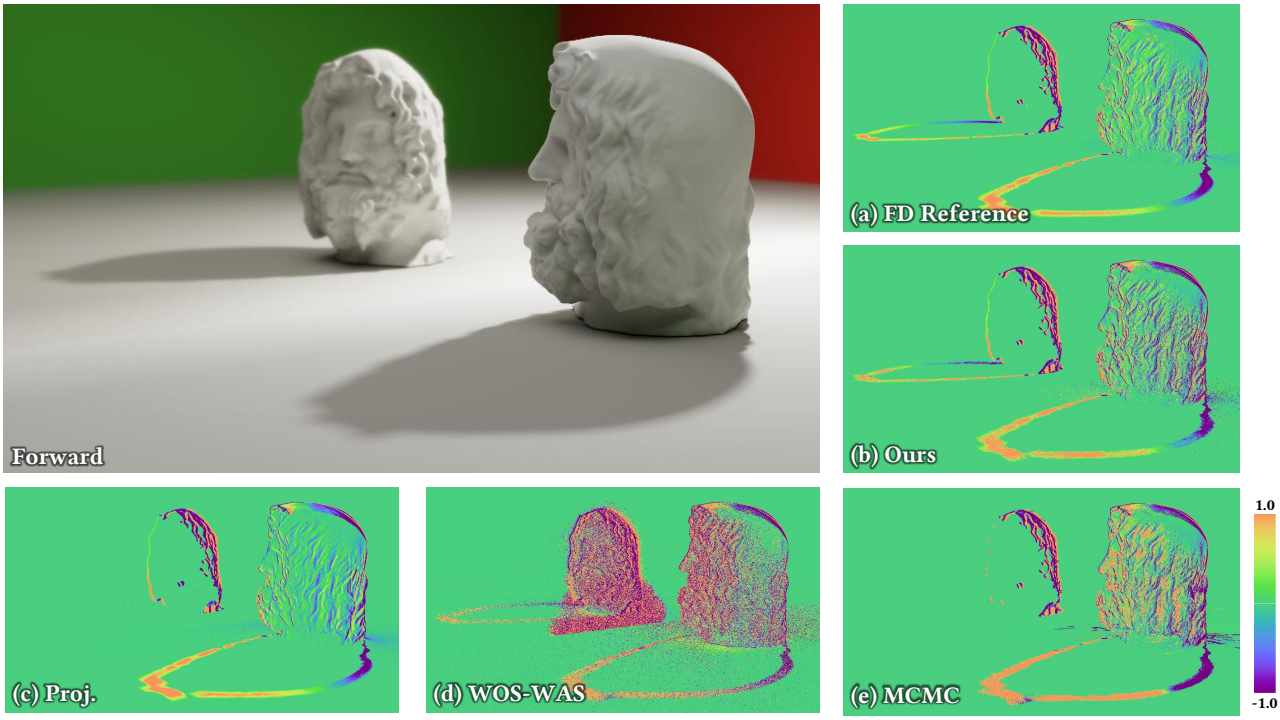

Fig. 1. We introduce a robust method to estimate *boundary* path integrals, which are crucial for computing derivatives with respect to scene geometry. In this example, a diffuse object is viewed both directly and through a mirror, and we differentiate the rendering with respect to the object's horizontal translation. At equal sample count, projective sampling [Zhang et al. 2023] misses the derivatives around the shadow in the mirror (c); Walk-on-Sphere-based warped-area sampling (WOS-WAS) [Wu et al. 2025] suffers from high overall variance (d); and MCMC [Xu et al. 2024] becomes trapped in a local mode with strong self-correlation (e). Our method (b) achieves the highest overall quality.

Authors' addresses: Peiyu Xu, peiyuxu3@illinois.edu, University of Illinois Urbana-Champaign, USA; Lifan Wu, lifanw@nvidia.com, NVIDIA, USA; Benedikt Bitterli, bbitterli@nvidia.com, NVIDIA, USA; Ravi Ramamoorthi, ravir@cs.ucsd.edu, NVIDIA, USA and University of California, San Diego, USA; Shuang Zhao, shzhao@illinois.edu, University of Illinois Urbana-Champaign, USA.

ACM 0730-0301/2026/7-ART137
https://doi.org/10.1145/3811325

To enable differentiation with respect to object geometries, boundary path integrals—central to physics-based differentiable rendering—must be estimated numerically. Although their mathematical formulation is well established, designing efficient and robust numerical estimators remains challenging. Most state-of-the-art boundary sampling methods rely on primary-sample-space guiding, which tends to break down on finely tessellated geometries; reparameterization-based alternatives, meanwhile, often incur high variance and/or significant computational overhead.

In this paper, we introduce a simple, robust, and consistent solution to this problem. At the core of our approach is a novel formulation of the boundary integral based on kernel-density estimation. Much like photon mapping, we slightly expand the measure-zero domain of integration with a kernel, sidestepping the need to sample directly on a delta-function region in path space. To our knowledge, this is the first such application of kernel-density methods for boundary integral evaluation. We validate our method

by comparing its derivative estimates against finite differences (FD), and further demonstrate its practical utility by benchmarking against several state-of-the-art baselines in synthetic inverse-rendering scenarios.

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

Additional Key Words and Phrases: differentiable rendering, boundary path integral, Monte Carlo estimation, kernel-density estimation

ACM Reference Format:
Peiyu Xu, Lifan Wu, Benedikt Bitterli, Ravi Ramamoorthi, and Shuang Zhao. 2026. Robust Computation of Boundary Path Integrals Using Kernel-Density Estimation . *ACM Trans. Graph.* 45, 4, Article 137 (July 2026), 12 pages. https://doi.org/10.1145/3811325

1 INTRODUCTION

Many inverse rendering, optimization, and learning tasks in computer graphics today rely on differentiable rendering for estimating the derivatives of the rendering process with respect to scene parameters such as camera poses, object geometries, and material properties. Recently, significant theoretical progress has been made [Li et al. 2018; Zhang et al. 2019, 2020; Bangaru et al. 2020; Xu et al. 2023; Wang et al. 2024; Xu et al. 2024], enabling differentiation of full light-transport effects (i.e., global illumination) with respect to arbitrary scene parameters including those controlling global object geometry (e.g., the positions of mesh vertices).

Mathematically, physics-based differentiable rendering generally requires evaluating *interior* and *boundary* integrals. The former share the same domain as forward-rendering integrals and can therefore be estimated by repurposing existing forward-rendering sampling strategies. To further improve the efficiency in estimating the *interior* integrals for differentiable rendering, several specialized techniques have been introduced that utilize new importance sampling strategies [Zeltner et al. 2021; Fan et al. 2024], antithetic sampling [Zhang et al. 2021], or spatiotemporal reuse [Chang et al. 2023; Wang et al. 2023].

Unique to differentiable rendering, the *boundary* integrals emerge from discontinuities such as visibility boundaries evolving with the differentiation parameter. Previously, these integrals were primarily estimated in two ways—either *directly* [Li et al. 2018; Zhang et al. 2020; Yan et al. 2022; Zhang et al. 2023; Xu et al. 2024] or after being *reparameterized* into *interior* integrals [Loubet et al. 2019; Bangaru et al. 2020; Xu et al. 2023].

Unfortunately, as Figure 1 illustrates, neither family is robust or efficient enough to handle highly tessellated scenes with complex light-transport effects such as specular interreflections. To overcome these limitations, we make the following contributions:

- We devise a new formulation that rewrites *boundary* path integrals as *interior* ones using kernel-density estimation (§4.1).
- Based on this formulation, we introduce a consistent Monte Carlo estimator to efficiently estimate the rewritten integrals (§4.2).

Our method is consistent, robust, and simple to deploy—requiring only a slight modification to unidirectional path tracing. We demonstrate the effectiveness of our technique using several differentiable and inverse rendering examples (Figures 1 and 9–12).

2 RELATED WORK

In what follows, we provide a brief recap of prior works related to our technique. In §2.1, we revisit recent differentiable rendering techniques for estimating *boundary* integrals. Then, in §2.2, we discuss techniques that utilize kernel-density estimation.

2.1 Boundary Integral Estimation

Direct Monte Carlo sampling of discontinuities. Li et al. [2018] introduced the first technique that directly estimates *boundary* integrals for differentiable rendering. Unfortunately, its reliance on silhouette detection with respect to arbitrary shading locations can be prohibitively expensive for complex scenes. To address this, Zhang et al. [2020] proposed a new path-space formulation that enables multi-directional sampling of boundary paths without explicit silhouette detection. Yan et al. [2022] further improved sampling efficiency by introducing guiding in the primary-sample space, and Zhang et al. [2023] extended this approach with a projection process that re-uses information from forward path tracing to guide *boundary* path sampling. While effective for relatively simple geometries, these guiding-based methods degrade on highly tessellated scenes that overwhelm their acceleration structures. More recently, Xu et al. [2024] introduced a Markov-Chain Monte Carlo (MCMC) technique that offers improved robustness for sampling *boundary* light paths but, like many MCMC rendering methods, is complex, exhibits non-uniform convergence, and can introduce undesired correlations.

Our method does not require MCMC while remaining robust and efficient even for highly tessellated geometries with complex light-transport effects.

Reparameterization-based methods. Another class of methods [Bangaru et al. 2020; Loubet et al. 2019; Xu et al. 2023] reparameterizes *boundary* integrals into *interior* ones, thereby avoiding direct sampling of discontinuities and enabling the reuse of forward-rendering sampling strategies. However, the divergence term they introduce is nontrivial to evaluate and, more importantly, to importance sample. Furthermore, as we demonstrate in the results, these methods tend to produce notably biased gradient estimates that can lead to overly smooth reconstructions.

Our method, on the contrary, leverages a novel kernel-density estimation process which avoids reparameterization and divergence estimation, and leads to better convergence in inverse rendering.

Reformulated integral for SDF discontinuity. Recently, Wang et al. [2024] proposed a method dedicated to differentiable rendering of implicit surfaces represented as signed-distance functions (SDFs). Their formulation relaxes the *boundary* integrals by allowing *boundary* paths to contain a segment passing through a thin shell around the surface, rather than being strictly tangent to it.

While conceptually related, our technique differs in two fundamental ways: their method targets smooth implicit geometry whereas ours focuses on piecewise planar explicit shapes; and our method builds on Zhang et al.'s path-space formulation [2020], while theirs relies on the standard rendering equation.

2.2 Kernel-Density Estimation

Forward rendering. Photon mapping, introduced by Jensen [1996], has been a key technique for efficiently simulating light transport. Prior work in forward rendering uses photon beams to query photons [Jarosz et al. 2008], represent photons [Jarosz et al. 2011b], or both [Jarosz et al. 2011a; Bitterli and Jarosz 2017]. Although our estimator also uses beam-like queries, it differs fundamentally from these forward methods since it estimates derivatives given by *boundary* path integrals rather than radiance from *interior* ones. VCM/UPS [Hachisuka et al. 2012; Georgiev et al. 2012] further developed an extended path-space formulation bridging photon mapping and path-space methods; while our method could fit within this framework, we do not explicitly perform vertex merging in our kernel density estimation.

Photon derivatives for forward rendering. Derivatives of photons have been previously used to accelerate forward rendering, such as by shaping the photon kernels [Schjøth et al. 2007; Frisvad et al. 2014], estimating motion blur [Schjøth et al. 2010], or in gradient domain methods [Gruson et al. 2018; Hua et al. 2017]. These improve forward rendering efficiency, while we focus on inverse rendering instead.

Photon derivatives for inverse rendering. A series of works inspired by optimal transport [Xing et al. 2022, 2023, 2024] use a different form of loss and formulate the gradient without boundary integrals. In particular, Xing et al. [2024] also use photon mapping for gradient computation, but their approach adopts a different theory and does not consider visibility; it is therefore orthogonal to our work.

Estimating query area for bias reduction. Using an incorrect query area introduces additional bias in photon-based methods, including ours. Prior works address this with Bernoulli trials for unbiased area estimation [Qin et al. 2015] or geometric methods [Perrot et al. 2019]. These are orthogonal to our work and could further reduce bias; for simplicity and performance, we use a constant approximation of the query area in this paper.

Consistency for photon mapping. A large body of prior works have investigated and developed consistent versions of photon mapping [Hachisuka et al. 2008; Hachisuka and Jensen 2009; Knaus and Zwicker 2011]. Following existing works, we also derive a consistent version of our estimator.

3 PRELIMINARIES

We now provide a brief summary of path-space forward rendering (§3.1) and differentiable rendering (§3.2).

3.1 Forward-Rendering Path Integrals

Many, if not most, advanced rendering algorithms express the response I of a radiometric detector as a **path integral** [Veach 1997]:

$$I = \int_\Omega f(\bar{\boldsymbol{x}})\, \mathrm{d}\mu(\bar{\boldsymbol{x}}), \tag{1}$$

where: $\Omega := \bigcup_{N=1}^{\infty} \mathcal{M}^{N+1}$ is the **path space** with $\mathcal{M}$ being the surfaces of all objects; $\bar{\boldsymbol{x}} = (\boldsymbol{x}_0, \ldots, \boldsymbol{x}_N) \in \Omega$ are **light paths** with $\boldsymbol{x}_0$ on a light source and $\boldsymbol{x}_N$ on the detector; and μ is the corresponding area-product measure. Further, the integrand f is the **measurement contribution function** given by

$$f(\bar{\boldsymbol{x}}) := L_\mathrm{e}(\boldsymbol{x}_0 \to \boldsymbol{x}_1)\, W_\mathrm{e}(\boldsymbol{x}_{N-1} \to \boldsymbol{x}_N) \left[\prod_{n=1}^{N-1} f_\mathrm{s}(\boldsymbol{x}_{n-1} \to \boldsymbol{x}_n \to \boldsymbol{x}_{n+1})\right] \left[\prod_{n=1}^{N} G(\boldsymbol{x}_{n-1} \leftrightarrow \boldsymbol{x}_n)\right], \tag{2}$$

where L_e and W_e are the **source emission** and **detector importance**, f_s is the **bidirectional scattering distribution function** (BSDF), and G is the **geometric term**.

Material-form reparameterization. To facilitate the differentiation of the path integral (1) with respect to parameters θ controlling object geometry (i.e., the surfaces $\mathcal{M}$), Zhang et al. [2020] have proposed to reparameterize the evolving object surfaces $\mathcal{M}(\theta)$ using a one-to-one mapping $\mathsf{X}(\cdot, \theta) : \mathcal{B} \mapsto \mathcal{M}(\theta)$ where $\mathcal{B}$ is some fixed **reference surface**. To distinguish points $\boldsymbol{p} \in \mathcal{B}$ on the reference surface and those $\boldsymbol{x} \in \mathcal{M}(\theta)$ on the evolving surface, we call the former **material points** and the latter **spatial points**.

Let $\bar{\boldsymbol{p}} = (\boldsymbol{p}_0, \ldots, \boldsymbol{p}_N)$ be a **material light path** with each vertex $\boldsymbol{p}_n$ on the reference surface $\mathcal{B}$. Then, the mapping $\mathsf{X}(\cdot, \theta)$ induces a change of variable from $\bar{\boldsymbol{p}}$ to a (spatial) light path $\bar{\boldsymbol{x}} = (\boldsymbol{x}_0, \ldots, \boldsymbol{x}_N)$ with $\boldsymbol{x}_n = \mathsf{X}(\boldsymbol{p}_n, \theta)$ for all $0 \le n \le N$. Applying this change of variable to the path integral (1) yields

$$I = \int_{\hat{\Omega}} f(\bar{\boldsymbol{x}}) \left\| \frac{\mathrm{d}\mu(\bar{\boldsymbol{x}})}{\mathrm{d}\mu(\bar{\boldsymbol{p}})} \right\| \mathrm{d}\mu(\bar{\boldsymbol{p}}), \tag{3}$$

where the domain of integration is the **material path space** $\hat{\Omega} := \bigcup_{N=1}^{\infty} \mathcal{B}^{N+1}$ independent of the parameter θ, f is the measurement contribution from Eq. (2), and the Jacobian term emerging from the change of variable is given by

$$\left\| \frac{\mathrm{d}\mu(\bar{\boldsymbol{x}})}{\mathrm{d}\mu(\bar{\boldsymbol{p}})} \right\| = \prod_{n=0}^{N} \left\| \frac{\mathrm{d}A(\boldsymbol{x}_n)}{\mathrm{d}A(\boldsymbol{p}_n)} \right\|. \tag{4}$$

Choice of the reference. When estimating derivatives at some fixed $\theta = \theta_0$, it is convenient to choose the reference surface as $\mathcal{B} = \mathcal{M}(\theta_0)$. Then, at $\theta = \theta_0$, the mapping $\mathsf{X}(\cdot, \theta)$ reduces to the identity, causing the material path space $\hat{\Omega}$ to coincide with the path space $\Omega(\theta_0)$ and the Jacobian term defined in Eq. (4) to reduce to one. We will use this choice of reference in the rest of this paper.

3.2 Differential Path Integrals

Zhang et al. [2020] have differentiated the material-form path integral of Eq. (3) with respect to the parameter θ using Reynolds transport theorem [Reynolds 1903]. The result can generally be expressed as **material-form differential path integrals**:

$$\partial_\theta I = \underbrace{\int_{\hat{\Omega}} \partial_\theta \left(f(\bar{\boldsymbol{x}}) \left\| \frac{\mathrm{d}\mu(\bar{\boldsymbol{x}})}{\mathrm{d}\mu(\bar{\boldsymbol{p}})} \right\| \right) \mathrm{d}\mu(\bar{\boldsymbol{p}})}_{\text{interior}} + \underbrace{\int_{\partial\hat{\Omega}} f(\bar{\boldsymbol{p}}_\partial)\, V(\boldsymbol{p}_K)\, \mathrm{d}\dot{\mu}(\bar{\boldsymbol{p}}_\partial)}_{\text{boundary}}, \tag{5}$$

where ∂_θ denotes the derivative with respect to θ evaluated at θ_0. In this equation, the *interior* component is an integral over the same material path space $\hat{\Omega}$ as the material path integral (3), while the *boundary* component captures the shift of discontinuous boundaries of the measurement contribution f with respect to the parameter θ.

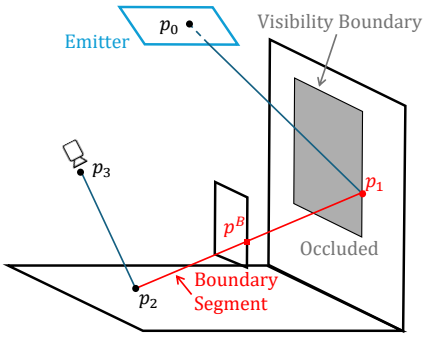

Fig. 2. **Boundary path and visibility boundary:** A boundary light path $\bar{\boldsymbol{p}}_\partial = (\boldsymbol{p}_0, \boldsymbol{p}_1, \boldsymbol{p}_2, \boldsymbol{p}_3)$ contains exactly one boundary segment $\overline{\boldsymbol{p}_1\,\boldsymbol{p}_2}$ (shown in red) that is tangent to the scene geometry at an additional point $\boldsymbol{p}^{\mathrm{B}}$.

Boundary path integral. In this paper, we focus on the estimation of the *boundary* integral in Eq. (5), which we now explain in more detail.

This integral is over the **boundary path space** $\partial\hat{\Omega}$ containing **boundary paths** $\bar{\boldsymbol{p}}_\partial = (\boldsymbol{p}_0, \boldsymbol{p}_1, \ldots, \boldsymbol{p}_N) \in \hat{\Omega}$ such that exactly one vertex $\boldsymbol{p}_K$ (for some $0 \le K < N$) is a jump discontinuity of the *mutual visibility* $\mathbb{V}(\boldsymbol{p}_K \leftrightarrow \boldsymbol{p}_{K+1})$ with $\boldsymbol{p}_{K+1}$ fixed. In this case, the **boundary segment** $\overline{\boldsymbol{p}_K\,\boldsymbol{p}_{K+1}}$ is tangent to the scene geometry $\mathcal{B} = \mathcal{M}(\theta_0)$ at a single point $\boldsymbol{p}^{\mathrm{B}}$ (see Figure 2). Additionally, the measure $\dot{\mu}$ associated with the boundary path space satisfies that $\mathrm{d}\dot{\mu}(\bar{\boldsymbol{p}}_\partial) = \mathrm{d}\ell(\boldsymbol{p}_K) \prod_{n\neq K} \mathrm{d}A(\boldsymbol{p}_n)$, where ℓ denotes the curve-length measure.

Lastly, $V(\boldsymbol{p}_K)$ is the **scalar normal velocity** defined as

$$V(\boldsymbol{p}_K) = \boldsymbol{n}_\partial(\boldsymbol{p}_K) \cdot \partial_\theta \boldsymbol{p}_K, \tag{6}$$

where $\boldsymbol{n}_\partial(\boldsymbol{p}_K)$ denotes the unit normal of the visibility boundary at $\boldsymbol{p}_K$. Please refer to the work by Zhang et al. [2020] for more details on the evaluation of the derivative $\partial_\theta \boldsymbol{p}_K$.

4 OUR METHOD

We now present our technique. In §4.1, we derive our formulation that expresses *boundary* path integrals as *interior* ones using kernel-density estimation. Then, in §4.2, we introduce a consistent Monte Carlo estimator based on the new formulation.

4.1 Our Formulation

As stated in §3, the *boundary* integral in Eq. (5) is defined over the boundary path space $\partial\hat{\Omega}$, which consists of boundary paths in which exactly one segment is restricted to a visibility boundary.

We decompose the boundary path space into disjoint subspaces $\{\partial\hat{\Omega}_{N,K} : N > 0 \text{ and } 0 \le K < N\}$ where $\partial\hat{\Omega}_{N,K}$ contains boundary paths $\bar{\boldsymbol{p}}_\partial = (\boldsymbol{p}_0, \ldots, \boldsymbol{p}_N)$ with N segments and $\overline{\boldsymbol{p}_K\boldsymbol{p}_{K+1}}$ being the boundary segment. Then, it holds that

$$\begin{aligned} I_{\mathrm{bnd}} &:= \int_{\partial\hat{\Omega}} f(\bar{\boldsymbol{p}}_\partial)\, V(\boldsymbol{p}_K)\, \mathrm{d}\dot{\mu}(\bar{\boldsymbol{p}}_\partial) \\ &= \sum_{N=0}^{\infty} \sum_{K=0}^{N-1} \int_{\partial\hat{\Omega}_{N,K}} f(\bar{\boldsymbol{p}}_\partial)\, V(\boldsymbol{p}_K)\, \mathrm{d}\dot{\mu}(\bar{\boldsymbol{p}}_\partial). \end{aligned} \tag{7}$$

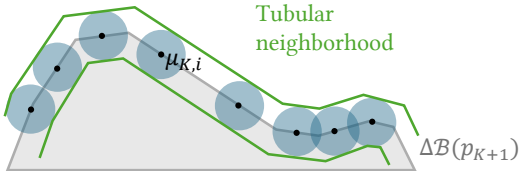

Fig. 3. We approximate the general impulse function $\delta_{\Delta\mathcal{B}(\boldsymbol{p}_{K+1})}$, which restricts surface integrals to the visibility-discontinuity curves $\Delta\mathcal{B}(\boldsymbol{p}_{K+1})$ (shown in dark gray), by a sum of 2D kernels $\mathcal{K}(\cdot;\boldsymbol{\mu}_{K,i})$ centered at points $\boldsymbol{\mu}_{K,i}$ on the boundary (shown as blue discs) for $i = 1, 2, \ldots, M$.

In the following, we introduce a consistent approximation for individual integrals on the right-hand side of this equation.

Specifically, we first rewrite each *boundary* integral as an *interior* one with a generalized impulse function:

$$\begin{aligned} &\int_{\partial\hat{\Omega}_{N,K}} f(\bar{\boldsymbol{p}}_\partial)\, V(\boldsymbol{p}_K)\, \mathrm{d}\dot{\mu}(\bar{\boldsymbol{p}}_\partial) \\ &\qquad = \int_{\mathcal{B}^{N+1}} f(\bar{\boldsymbol{p}})\, V(\boldsymbol{p}_K)\, \delta_{\Delta\mathcal{B}(\boldsymbol{p}_{K+1})}(\boldsymbol{p}_K)\, \mathrm{d}\mu(\bar{\boldsymbol{p}}), \end{aligned} \tag{8}$$

where $\Delta\mathcal{B}(\boldsymbol{p}_{K+1}) \subset \mathcal{B}$ denotes the visibility boundaries on which $\boldsymbol{p}_K$ must reside, and $\delta_{\Delta\mathcal{B}(\boldsymbol{p}_{K+1})}$ is the generalized impulse function that turns an area integral of $\boldsymbol{p}_K$ into one over the curves $\Delta\mathcal{B}(\boldsymbol{p}_{K+1})$.

In what follows, we express the generalized impulse function as conventional Dirac delta functions before softening them. We consider a tubular neighborhood (see Figure 3) of the curves $\Delta\mathcal{B}(\boldsymbol{p}_{K+1})$ parameterized with $\boldsymbol{q}$ defined on $\mathbb{R} \times (-\epsilon, \epsilon)$ for some small $\epsilon > 0$ such that $\boldsymbol{q}(s, 0)$ is an arc-length parameterization of $\Delta\mathcal{B}(\boldsymbol{p}_{K+1})$. Then, since the impulse function $\delta_{\Delta\mathcal{B}(\boldsymbol{p}_{K+1})}$ restricts the integral over points $\boldsymbol{q}(s,t)$ with $t = 0$, it holds that

$$\delta_{\Delta\mathcal{B}(\boldsymbol{p}_{K+1})}(\boldsymbol{q}(s,t)) = \delta(t)\, J(s,t) = \int_{\mathbb{R}} \delta(t)\, \delta(s'-s)\, J(s,t)\mathrm{d}s', \tag{9}$$

where $J(s,t) := \|\mathrm{d}A(\boldsymbol{q}(s,t))/\mathrm{d}s\,\mathrm{d}t\|$ is the Jacobian term associated with the parameterization.

In this paper, we focus on scene geometries expressed explicitly using polygonal meshes. In this case, the visibility boundaries $\Delta\mathcal{B}(\boldsymbol{p}_{K+1})$ are piecewise linear, allowing their tubular neighborhood to be parameterized with the Jacobian term $J(s,t) \equiv 1$ for all s and t. Under this setting, we now soften the Dirac delta functions on the right-hand side of Eq. (9) using some smooth 1D kernel (e.g., Gaussian) $\mathcal{K}_1$ that is centered at the origin, resulting in

$$\int_{\mathbb{R}} \delta(t)\, \delta(s'-s)\, \mathrm{d}s' \xrightarrow{\text{soften}} \int_{\mathbb{R}} \mathcal{K}_1(t)\, \mathcal{K}_1(s'-s)\, \mathrm{d}s'. \tag{10}$$

Estimating the right-hand side using Monte Carlo integration yields

$$\int_{\mathbb{R}} \delta(t)\, \delta(s'-s)\, \mathrm{d}s' \approx \frac{1}{M} \sum_{i=1}^{M} \frac{1}{P_{K,i}}\, \mathcal{K}_1(t)\, \mathcal{K}_1(s_{K,i}-s), \tag{11}$$

where $s_{K,i}$ are randomly sampled with the probability $P_{K,i}$ for $i = 1, 2, \ldots, M$. Let $\boldsymbol{p} = \boldsymbol{q}(s,t)$ be a point within the tubular neighborhood for some $s \in \mathbb{R}$ and $t \in (-\epsilon, \epsilon)$. Substituting Eq. (11) into Eq. (9) leads to an approximation of the general impulse function

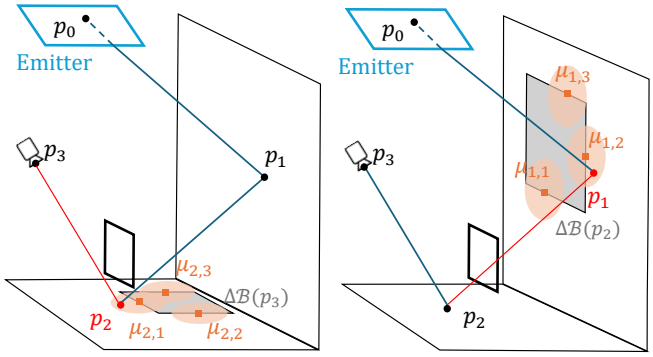

Fig. 4. The integrand of our reformulated *boundary* path integral (15) involves a summation over all path segments. In this example, for a path $\bar{\boldsymbol{p}} = (\boldsymbol{p}_0, \boldsymbol{p}_1, \boldsymbol{p}_2, \boldsymbol{p}_3)$, we illustrate the segment contributions for $\overline{\boldsymbol{p}_2\,\boldsymbol{p}_3}$ (left) and $\overline{\boldsymbol{p}_1\,\boldsymbol{p}_2}$ (right) using $M = 3$ kernels. For the segment $\overline{\boldsymbol{p}_2\,\boldsymbol{p}_3}$, the kernels (shown as orange discs) are centered at $\boldsymbol{\mu}_{2,1}, \boldsymbol{\mu}_{2,2}$, and $\boldsymbol{\mu}_{2,3}$, randomly drawn from the visibility boundary $\Delta\mathcal{B}(\boldsymbol{p}_3)$ and evaluated at $\boldsymbol{p}_2$. Similarly, the segment $\overline{\boldsymbol{p}_1\,\boldsymbol{p}_2}$ uses kernels centered at $\boldsymbol{\mu}_{1,1}, \boldsymbol{\mu}_{1,2}$, and $\boldsymbol{\mu}_{1,3}$, randomly drawn from the visibility boundary $\Delta\mathcal{B}(\boldsymbol{p}_2)$ and evaluated at $\boldsymbol{p}_1$.

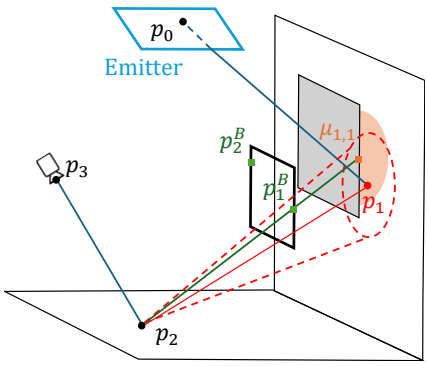

Fig. 5. To sample points $\boldsymbol{\mu}_{K,i}$ along the visibility boundaries $\Delta\mathcal{B}(\boldsymbol{p}_{K+1})$, we instead draw points $\boldsymbol{p}_i^{\mathrm{B}}$ along face edges during pre-computation and obtain $\boldsymbol{\mu}_{K,i}$ by tracing a ray (shown in green) from $\boldsymbol{p}_{K+1}$ through $\boldsymbol{p}_i^{\mathrm{B}}$. At runtime, to quickly identify the edge points $\boldsymbol{p}_i^{\mathrm{B}}$ near the segment $\overline{\boldsymbol{p}_{K+1}\,\boldsymbol{p}_K}$, we build an acceleration structure (e.g., a BVH) over these points and query for those lying within a cone (shown in red).

$\delta_{\Delta\mathcal{B}(\boldsymbol{p}_{K+1})}$:

$$\delta_{\Delta\mathcal{B}(\boldsymbol{p}_{K+1})}(\boldsymbol{p}) \approx \frac{1}{M}\sum_{i=1}^{M}\frac{\mathcal{K}(\boldsymbol{p};\boldsymbol{\mu}_{K,i})}{P_{K,i}}, \tag{12}$$

where $\boldsymbol{\mu}_{K,i} := \boldsymbol{q}(s_{K,i}, 0)$ are points on the boundary $\Delta\mathcal{B}(\boldsymbol{p}_{K+1})$, and

$$\mathcal{K}(\boldsymbol{q}(s,t);\boldsymbol{\mu}_{K,i}) := \mathcal{K}_1(t)\,\mathcal{K}_1(s_{K,i} - s) \tag{13}$$

is an isotropic 2D kernel centered at $\boldsymbol{\mu}_{K,i}$ (see Figure 3).

Then, substituting Eq. (12) into Eq. (8) produces:

$$\begin{aligned}&\int_{\partial\hat{\Omega}_{N,K}} f(\bar{\boldsymbol{p}}_\partial)\,V(\boldsymbol{p}_K)\,\mathrm{d}\dot{\mu}(\bar{\boldsymbol{p}}_\partial)\\ &\approx \int_{\mathcal{B}^{N+1}} f(\bar{\boldsymbol{p}})\left(\frac{1}{M}\sum_{i=1}^{M} V(\boldsymbol{\mu}_{K,i})\,\frac{\mathcal{K}(\boldsymbol{p}_K;\boldsymbol{\mu}_{K,i})}{P_{K,i}}\right)\mathrm{d}\mu(\bar{\boldsymbol{p}}),\end{aligned} \tag{14}$$

where $V(\boldsymbol{p}_K)$ on the right-hand side is replaced with $V(\boldsymbol{\mu}_{K,i})$ since the scalar normal velocity V is well defined only on the visibility boundary $\Delta\mathcal{B}(\boldsymbol{p}_{K+1})$.

Lastly, according to Eqs. (7) and (14), we can approximate the full *boundary* integral in Eq. (5) as an *interior* one:

$$\boxed{I_{\mathrm{bnd}} \approx \int_{\hat{\Omega}} f(\bar{\boldsymbol{p}})\left(\frac{1}{M}\sum_{K=0}^{N-1}\sum_{i=1}^{M} V(\boldsymbol{\mu}_{K,i})\,\frac{\mathcal{K}(\boldsymbol{p}_K;\boldsymbol{\mu}_{K,i})}{P_{K,i}}\right)\mathrm{d}\mu(\bar{\boldsymbol{p}}).} \tag{15}$$

This reformulated integral (15) is defined over the *interior* of the material path space $\hat{\Omega} = \Omega(\theta_0)$. For each path $\bar{\boldsymbol{p}} = (\boldsymbol{p}_0, \ldots, \boldsymbol{p}_N)$, the integrand is given by its measurement contribution $f(\bar{\boldsymbol{p}})$ modulated by a summation over all path segments (as illustrated in Figure 4).

4.2 Our Estimator

Evaluating the right-hand side of Eq. (15) requires sampling points $\{\boldsymbol{\mu}_{K,i} : i = 1, 2, \ldots, M\}$ along the visibility boundaries $\Delta\mathcal{B}(\boldsymbol{p}_{K+1})$ for each $0 \le K < N$, which is a nontrivial task.

To address this problem, we note that the visibility boundaries $\Delta\mathcal{B}(\boldsymbol{p}_{K+1})$ are the projection of some silhouettes with respect to the point $\boldsymbol{p}_{K+1}$. Under our assumption of polygonal-mesh-based scene geometry, such silhouettes consist of edges separating front- and back-facing faces. We observe that the point $\boldsymbol{p}_K$ is close to the visibility boundary $\Delta\mathcal{B}(\boldsymbol{p}_{K+1})$ if and only if the segment $\overline{\boldsymbol{p}_K\,\boldsymbol{p}_{K+1}}$ is close to some point $\boldsymbol{p}^{\mathrm{B}}$ on a silhouette face edge. Thus, for each i, we instead sample $\boldsymbol{p}_i^{\mathrm{B}}$ uniformly over the face edges and let $\boldsymbol{\mu}_{K,i} = \mathrm{rayIntersect}(\boldsymbol{p}_{K+1} \to \boldsymbol{p}_i^{\mathrm{B}})$ be the intersection between the ray from $\boldsymbol{p}_{K+1}$ toward $\boldsymbol{p}_i^{\mathrm{B}}$ and the scene geometry $\mathcal{B}$ (see Figure 5). Then, using a relation derived by Zhang et al. [2019], it holds that

$$P_{K,i} = \frac{1}{E}\,\frac{\|\boldsymbol{p}_i^{\mathrm{B}} - \boldsymbol{p}_{K+1}\|}{\|\boldsymbol{\mu}_{K,i} - \boldsymbol{p}_{K+1}\|}\,\frac{\sin\theta^{\mathrm{B}}}{\sin\theta^{\mathrm{S}}|\cos\theta^{\mathrm{D}}|}, \tag{16}$$

where E denotes the total length of face edges. Additionally, θ^{B} and θ^{S} denote the angles between the direction $\overrightarrow{\boldsymbol{\mu}_{K,i}\boldsymbol{p}_i^{\mathrm{B}}}$ and, respectively, the edge containing $\boldsymbol{p}_i^{\mathrm{B}}$ and the tangent of visibility boundary $\Delta\mathcal{B}(\boldsymbol{p}_{K+1})$ at $\boldsymbol{\mu}_{K,i}$; θ^{D} indicates the angle between $\overrightarrow{\boldsymbol{\mu}_{K,i}\boldsymbol{p}_i^{\mathrm{B}}}$ and the surface normal at $\boldsymbol{\mu}_{K,i}$.

In practice, we set the kernel as a uniform disc function:

$$\mathcal{K}(\boldsymbol{p}_K;\boldsymbol{\mu}_{K,i}) := \begin{cases} 1/\pi r^2 & \text{if } \|\boldsymbol{p}_K - \boldsymbol{\mu}_{K,i}\| < r,\\ 0 & \text{otherwise,}\end{cases} \tag{17}$$

where r is a hyperparameter indicating the disc radius. In addition, to accelerate the computation of $\boldsymbol{\mu}_{K,i}$, we intersect the ray $\boldsymbol{p}_{K+1} \to \boldsymbol{p}_i^{\mathrm{B}}$ against only the plane containing $\boldsymbol{p}_K$ (instead of the entire scene geometry $\mathcal{B}$). Since the scene geometry is piecewise planar and $\boldsymbol{p}_K$ needs to be very close to $\boldsymbol{\mu}_{K,i}$ to have a non-negligible contribution, we did not observe any significant bias introduced by this approximation.

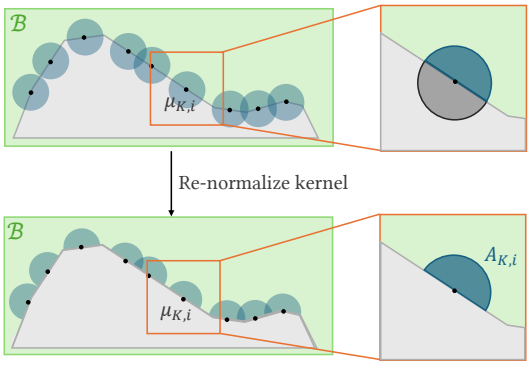

Fig. 6. **Kernel re-normalization:** (Top) Our derivation in §4.1 assumes that each vertex $\boldsymbol{p}_K$ of a light path is sampled over the entire scene geometry $\mathcal{B}$. (Bottom) In contrast, practical path-sampling strategies such as unidirectional path tracing typically draw $\boldsymbol{p}_K$ only from the sub-region visible to $\boldsymbol{p}_{K+1}$ (shown in light green). In this case, we re-normalize each kernel $\mathcal{K}(\cdot;\boldsymbol{\mu}_{K,i})$ using the area $A_{K,i}$ of the intersection between the disc and this visible sub-region (shown in dark green).

Our full algorithm. Based on the discussions above, we introduce a two-stage algorithm to estimate the *boundary* integral using Eq. (15) as follows.

During the first *pre-computation* stage, we sample M points $\{\boldsymbol{p}_i^{\mathrm{B}} : i = 1, 2, \ldots, M\}$ uniformly over all face edges, and then build an acceleration structure (e.g., Kd-tree) for these points.

During the second *run-time* stage, we perform standard algorithms (e.g., unidirectional path tracing) to sample light paths. For each path $\bar{\boldsymbol{p}} = (\boldsymbol{p}_0, \ldots, \boldsymbol{p}_N)$, we evaluate the right-hand side of Eq. (15) by examining each segment $\overline{\boldsymbol{p}_K\boldsymbol{p}_{K+1}}$ (for $0 \le K < N$). Since iterating over all the edge points $\boldsymbol{p}_i^{\mathrm{B}}$ generated in the first stage is usually impractical, as illustrated in Figure 5, we search for those inside a cone with the apex $\boldsymbol{p}_{K+1}$ and axis $\overrightarrow{\boldsymbol{p}_{K+1}\boldsymbol{p}_K}$ using the acceleration structure (and compute the corresponding $\boldsymbol{\mu}_{K,i}$ using ray intersection). The angle of this cone is determined based on the length of the segment $\overline{\boldsymbol{p}_K\boldsymbol{p}_{K+1}}$ and the hyperparameter r in Eq. (17).

Silhouette checks. We recall that, to obtain the points $\boldsymbol{\mu}_{K,i}$ over the visibility boundaries $\Delta\mathcal{B}(\boldsymbol{p}_{K+1})$, we need the corresponding $\boldsymbol{p}_i^{\mathrm{B}}$ to be located on silhouettes with respect to $\boldsymbol{p}_{K+1}$. To ensure that this is indeed the case, for each $\boldsymbol{p}_i^{\mathrm{B}}$ inside the cone, we only consider it valid if

$$\left(\boldsymbol{n}_1 \cdot \overrightarrow{\boldsymbol{p}_{K+1}\boldsymbol{p}_K}\right)\left(\boldsymbol{n}_2 \cdot \overrightarrow{\boldsymbol{p}_{K+1}\boldsymbol{p}_K}\right) < 0, \tag{18}$$

where $\boldsymbol{n}_1$ and $\boldsymbol{n}_2$ are, respectively, the outward normals of the two faces that share the edge containing $\boldsymbol{p}_i^{\mathrm{B}}$.

Kernel normalization. Our previous derivation integrates $\boldsymbol{p}_K$ over the entirety of the scene geometry $\mathcal{B}$. On the other hand, when sampled using standard methods like path tracing, $\boldsymbol{p}_K$ will be restricted to points visible to $\boldsymbol{p}_{K+1}$. To ensure that our kernel $\mathcal{K}(\cdot;\boldsymbol{\mu}_{K,i})$ remains normalized under this setting, we re-normalize it by letting

$$\mathcal{K}(\boldsymbol{p};\boldsymbol{\mu}_{K,i}) = \begin{cases} A_{K,i}^{-1} & \text{if } \|\boldsymbol{p} - \boldsymbol{\mu}_{K,i}\| < r, \\ 0 & \text{otherwise,} \end{cases} \tag{19}$$

where $A_{K,i}$ denotes the area of the sub-region inside the r-radius disc centered at $\boldsymbol{\mu}_{K,i}$ that is visible to $\boldsymbol{p}_{K+1}$, as illustrated in Figure 6. In practice, since the visibility boundary $\Delta\mathcal{B}(\boldsymbol{p}_{K+1})$ is piecewise linear, we approximate $A_{K,i}$ using half the area of the disc (i.e., $A_{K,i} \approx \pi r^2/2$). Although this approximation breaks down when $\boldsymbol{\mu}_{K,i}$ is located near a vertex of the visibility boundary $\Delta\mathcal{B}(\boldsymbol{p}_{K+1})$, we found the resulting error to vanish as long as the kernel size r is sufficiently small, as we will demonstrate in Figure 7.

Full estimator. Putting everything together, we obtain the following estimator for the *boundary* integral:

$$\boxed{\langle I_{\mathrm{bnd}}\rangle = \frac{f(\bar{\boldsymbol{p}})}{\mathrm{pdf}(\bar{\boldsymbol{p}})}\left(\frac{1}{M}\sum_{K=0}^{N-1}\sum_{\boldsymbol{p}_i^{\mathrm{B}}} V(\boldsymbol{\mu}_{K,i})\,\frac{\mathcal{K}(\boldsymbol{p}_K;\boldsymbol{\mu}_{K,i})}{P_{K,i}}\right),} \tag{20}$$

where: the light path $\bar{\boldsymbol{p}}$ is sampled using standard methods (e.g., unidirectional path tracing) with the probability density $\mathrm{pdf}(\bar{\boldsymbol{p}})$; $P_{K,i}$ and $\mathcal{K}(\boldsymbol{p}_K;\boldsymbol{\mu}_{K,i})$ are given by Eqs. (16) and (19), respectively; and the summation of $\boldsymbol{p}_i^{\mathrm{B}}$ is over all valid pre-sampled edge points within the cone with the apex $\boldsymbol{p}_{K+1}$ and axis $\overrightarrow{\boldsymbol{p}_{K+1}\boldsymbol{p}_K}$.

Consistent Estimator. Following Knaus and Zwicker [2011], we obtain a consistent version of our estimator by running multiple iterations of the algorithm while progressively reducing the kernel size. Specifically, we update the kernel size following

$$r_{t+1} = r_t\sqrt{\frac{t+\alpha}{t+1}} \tag{21}$$

for each iteration t, where $\alpha \in (0, 1)$ is a user-defined hyperparameter to control the rate of radius decay. At each iteration t, we apply the aforementioned algorithm with radius r_t to compute Eq. (20) as $\langle I_{\mathrm{bnd}}\rangle_t$, and the final boundary integral is computed as an average across all iterations

$$\langle I_{\mathrm{bnd}}\rangle_{\mathrm{final}} = \frac{1}{T}\sum_{t=0}^{T}\langle I_{\mathrm{bnd}}\rangle_t. \tag{22}$$

In practice, this is computed by keeping a running average and progressively updating the gradient values. We further provide a proof of consistency in the supplemental materials, and a validation experiment in §5.1.

5 RESULTS

In the following, we describe our implementation details and then, in §5.1, conduct ablation studies and validate our method. Additional results are presented in §5.2.

Implementation Details. We implement our technique presented in §4 on the CPU using the Enzyme automatic differentiation framework [Moses and Churavy 2020], and on the GPU based on Falcor [Kallweit et al. 2022]. The algorithm runs in two stages. In the pre-computation stage, we build a simple edge distribution similar to prior methods, and sample a set of edge points. The sampled points are stored as spheres with a radius equal to the search radius in a BVH structure. In our implementation, we use the BVH interface provided by Embree and a custom intersection filter for this purpose. In the run-time stage, we search for all edge points inside a cone by

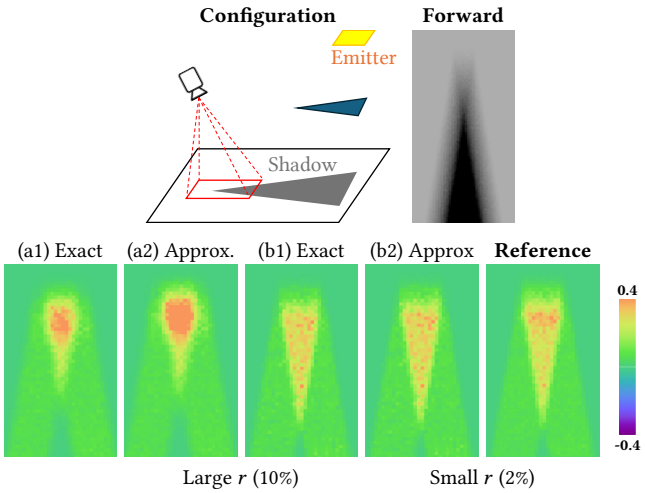

Fig. 7. We conduct an **ablation** to study the effect of the kernel size r on the accuracy of our approximation $A_{K,i} \approx \pi r^2/2$: (a1) large r (10% of the triangle size) with exact $A_{K,i}$; (a2) large r with the approximated $A_{K,i}$; (b1) small r (2% of the triangle size) with exact $A_{K,i}$; (b2) small r with the approximated $A_{K,i}$. The approximation error vanishes as r decreases.

running an Any-Hit intersection query to collect all intersections between a path segment and the points (spheres) generated in the first stage. Intersections that do not lie inside the cone or fail the silhouette test are excluded, and we use the remaining valid particles to compute the edge contribution.

Baselines. We compare against four state-of-the-art baselines: walk-on-sphere-based warped-area sampling (WOS-WAS) [Wu et al. 2025], projective sampling [Zhang et al. 2023], quadric-based silhouette sampling [Soroka et al. 2025], and MCMC-based sampling [Xu et al. 2024]. Of these, WOS-WAS reparameterizes the *boundary* path integral using the divergence theorem, while the other three directly sample boundary paths. For WOS-WAS, we only compare the gradient images, as the inverse rendering code was not provided. We also observe that WOS-WAS suffers from abnormally high variance when handling secondary boundaries in mirrors.

Equal-sample comparisons. The baselines are implemented on very different rendering systems: projective sampling uses Mitsuba 3 [Jakob et al. 2022]; quadric-based silhouette sampling uses Redner [Li et al. 2018]; WOS-WAS uses Falcor [Kallweit et al. 2022]; and MCMC uses the same Enzyme-based CPU codebase as our method. Therefore, we opt for equal-sample comparisons to better compare the algorithms (rather than rendering systems).

We note that, if implemented on the same system, our method would most likely cost the least amount of time per path sample. To provide an example, the teaser scene took projective sampling, WOS-WAS and our method 2.9s, 4.8s, and 0.4s to render at 4spp on the GPU, respectively.

5.1 Ablations and Evaluations

Ablations. We conduct an ablation (Figure 7) to study how the kernel radius r affects the accuracy of our approximation $A_{K,i} \approx \pi r^2/2$ used when re-normalizing the kernels via Eq. (19). To this end, we compare derivative images of the cast shadow of a simple triangle (shown at the top of the figure), focusing on the shadow cast by a sharp corner, where the approximation is most likely to break down.

When the kernel size is large (10% relative to the triangle size), using our technique outlined in Eq. (20) with the ground-truth (i.e., non-approximated) area $A_{K,i}$ suffers from high bias emerging from the kernel-density estimation (Figure 7-a1). When using our approximated $A_{K,i}$, the bias further increases (Figure 7-a2). This is expected because larger kernels are more likely to contain vertices of the visibility boundary.

When the kernel size is small (2% relative to the triangle size), the derivatives contain significantly lower bias (Figure 7-b1). With the approximated $A_{K,i}$, the bias remains roughly at the same level (Figure 7-b2), indicating that our approximation of $A_{K,i}$ performs well.

Validation. In Figure 8, we validate the consistency of our estimator. Starting from a large kernel size (10% of the object size), we run our consistent estimator from §4.2 for multiple iterations at 4 spp per iteration, and observe convergence to the reference image as the number of iterations increases.

5.2 Additional Results

To further demonstrate the effectiveness of our method, we compare parameter-space gradients and inverse-rendering results generated with our method and the baselines.

Configurations. We use two scene setups, shadow and mirror, for the experiments in this subsection. In the shadow configuration, the camera observes the cast shadow of the object. The mirror configuration, on the other hand, has a diffuse object located next to a mirror. The camera observes the specular reflection of the object. Both setups follow existing works [Zhang et al. 2023; Soroka et al. 2025; Xu et al. 2024] to emphasize the secondary boundary integral.

For all experiments in this subsection, we use $\alpha = 0.3$ and $T = 4$ for the consistent estimator, and set the initial kernel radius to be around 2% of the size of the object. We find this hyperparameter choice to be robust across all the scenes that we have tested.

Gradient Images. We visualize the gradient images $\mathrm{d}I/\mathrm{d}\boldsymbol{\theta}$ produced by our method and baselines in Figure 9. The experiment is conducted on both shadow and mirror setups with different geometries.

We further test our algorithm in complex indoor scenes in Figure 10. The scenes require multiple bounces and contain complex geometries, yet our method remains robust and produces high-quality gradients.

Parameter-space gradients. We evaluate parameter-space gradients $\mathrm{d}\mathcal{L}/\mathrm{d}\boldsymbol{\theta}$ of some loss $\mathcal{L}$ estimated by our method and the baselines in Figure 11, using the shadow configuration. In this experiment, the parameters $\boldsymbol{\theta}$ represent individual vertex positions of the object mesh.

We note that, according to the chain rule, it holds that $\mathrm{d}\mathcal{L}/\mathrm{d}\boldsymbol{\theta} = (\mathrm{d}\mathcal{L}/\mathrm{d}\boldsymbol{I})\,(\mathrm{d}\boldsymbol{I}/\mathrm{d}\boldsymbol{\theta})$, where $\boldsymbol{I}$ denotes the image rendered using $\boldsymbol{\theta}$. Since we do not yet have specified target images needed to evaluate the

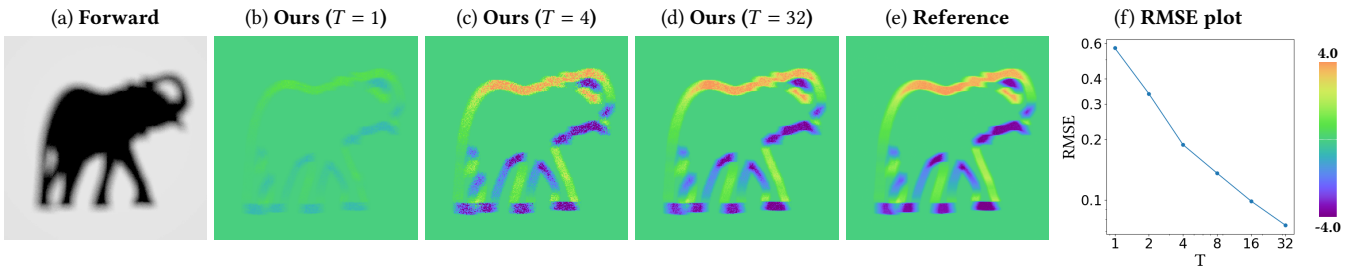

Fig. 8. We validate the consistency of our method by gradually increasing the number of iterations T and tracking the RMSE. The log-log plot (f) shows that our estimate converges to the reference (e) as T increases. Panels (b)–(d) visualize the corresponding decrease in bias.

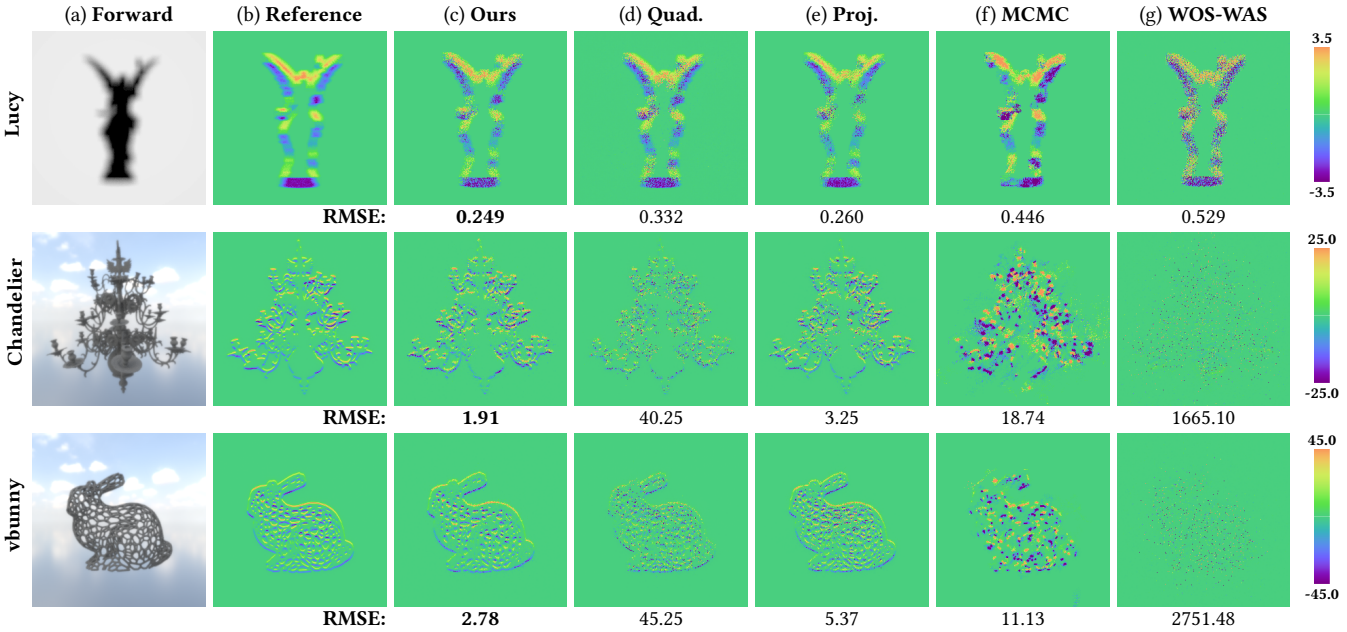

Fig. 9. We evaluate the efficiency of our method with an equal-sample differentiable rendering experiment. The **lucy** experiment uses the shadow setup, while **chandelier** and **vbunny** use the mirror setup. Despite the complex geometry and visibility, our method consistently outperforms the baselines across all three experiments.

loss $\mathcal{L}$ and its gradient $\mathrm{d}\mathcal{L}/\mathrm{d}I$, we instead set $\mathrm{d}\mathcal{L}/\mathrm{d}I$ to an all-one vector and estimate $\mathrm{d}I/\mathrm{d}\boldsymbol{\theta}$ using our method and the baselines.

We visualize in Figure 11 the magnitude of gradients with respect to per-vertex positions on a UV map. Our method produces low-variance gradients with almost negligible bias, whereas the baseline methods exhibit substantially larger noise.

Inverse-rendering results. We compare inverse-rendering results generated by projective sampling [Zhang et al. 2023], quadric-based silhouette sampling [Soroka et al. 2025], and our method under the mirror configuration with several target shapes in Figure 12. We initialize the shape of the object using a sphere with 40,000 vertices and use 40 target images for the optimizations. Remeshing is applied every 500 iterations for all methods. Although our method runs faster than the baselines, we use equal sample counts here to give a conservative assessment of the convergence behavior; timing statistics are reported in Figure 12 for reference. The mirror setting poses a significant challenge due to the use of higher-tessellated geometries and the presence of specular inter-reflection. In this regime, our method offers a new level of robustness, consistently achieving faster convergence than the baselines.

6 DISCUSSION AND CONCLUSION

Limitations and future work. As shown in Figure 13, our method can exhibit high variance when derivatives are dominated by extremely soft shadows or rough reflections. Using the mirror setting, we render gradient images with increasing roughness: our advantage gradually vanishes and eventually falls behind the projective sampling [Zhang et al. 2023] baseline. Fortunately, practical inverse-rendering configurations rarely rely on such effects, which typically carry too little information for reconstruction.

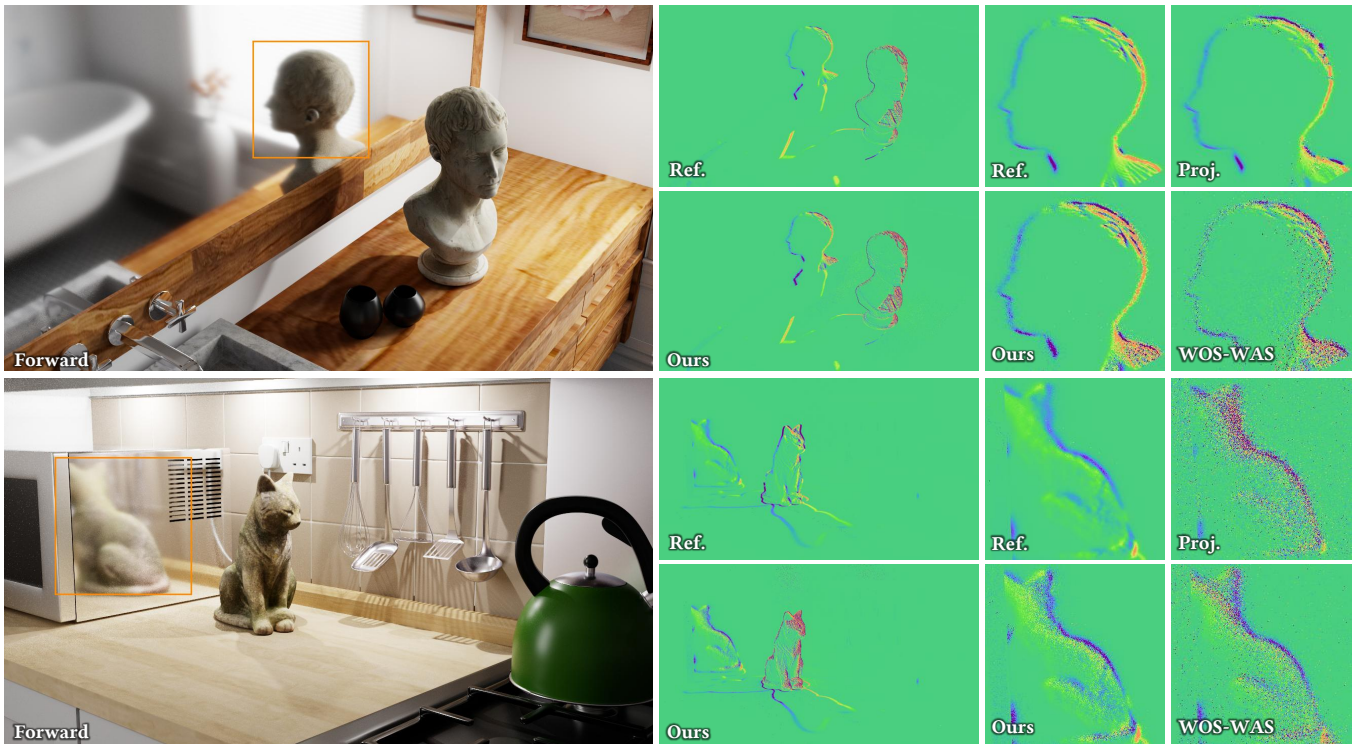

Fig. 10. We show the performance of our method on complex indoor scenes. Our method produces gradient images that closely match the finite-difference reference. Note that the high-variance region are on the surface of the moving object, which is typically handled by the interior term or pixel boundary and is orthogonal to our method. We further highlight regions dominated by the boundary integral and compare our result against the reference, projective sampling [Zhang et al. 2023], and WOS-WAS [Wu et al. 2025].

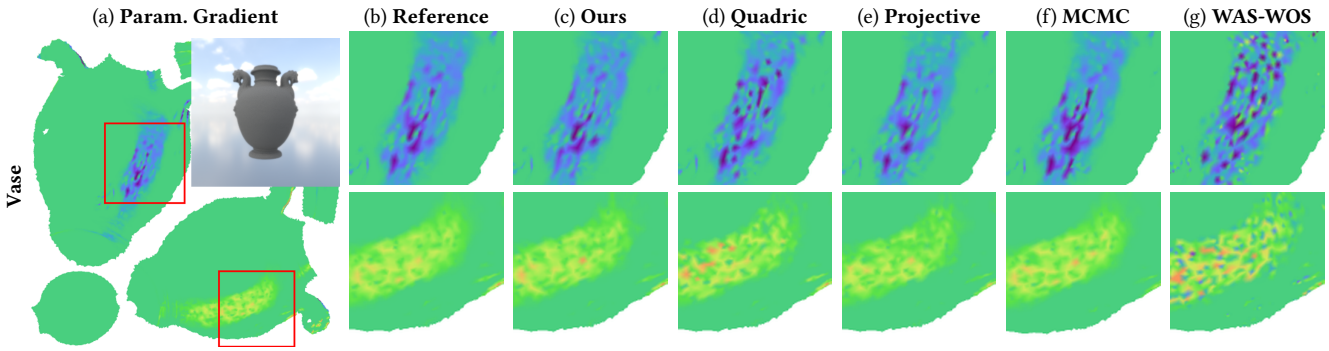

Fig. 11. We compare **gradients $\mathrm{d}\mathcal{L}/\mathrm{d}\boldsymbol{\theta}$ with respect to individual vertex positions**, obtained (with $\mathrm{d}\mathcal{L}/\mathrm{d}I$ set to the all-ones vector) using our method (c), Quadric-based Sampling [Soroka et al. 2025] (d), Projective Sampling [Zhang et al. 2023] (e), MCMC-based Sampling [Xu et al. 2024] (f), and WOS-WAS [Wu et al. 2025] (g). We visualize the magnitude of the gradient vector at each vertex of the sphere by projecting it onto a UV map.

Additionally, our formulation assumes piecewise planar geometry; generalizing it to smooth surfaces is an interesting direction for future research. Lastly, our estimator relies on unidirectional path tracing; extending it to bidirectional path tracing (BDPT) would further improve robustness for complex effects such as caustics.

Conclusion. We introduced a new formulation that leverages kernel-density estimation to rewrite *boundary* path integrals as *interior* ones, and developed a consistent Monte Carlo estimator for the rewritten integrals. We demonstrated its effectiveness against state-of-the-art direct sampling [Zhang et al. 2023; Xu et al. 2024; Soroka et al. 2025] and reparameterization [Wu et al. 2025] methods on a variety of synthetic differentiable and inverse rendering examples.

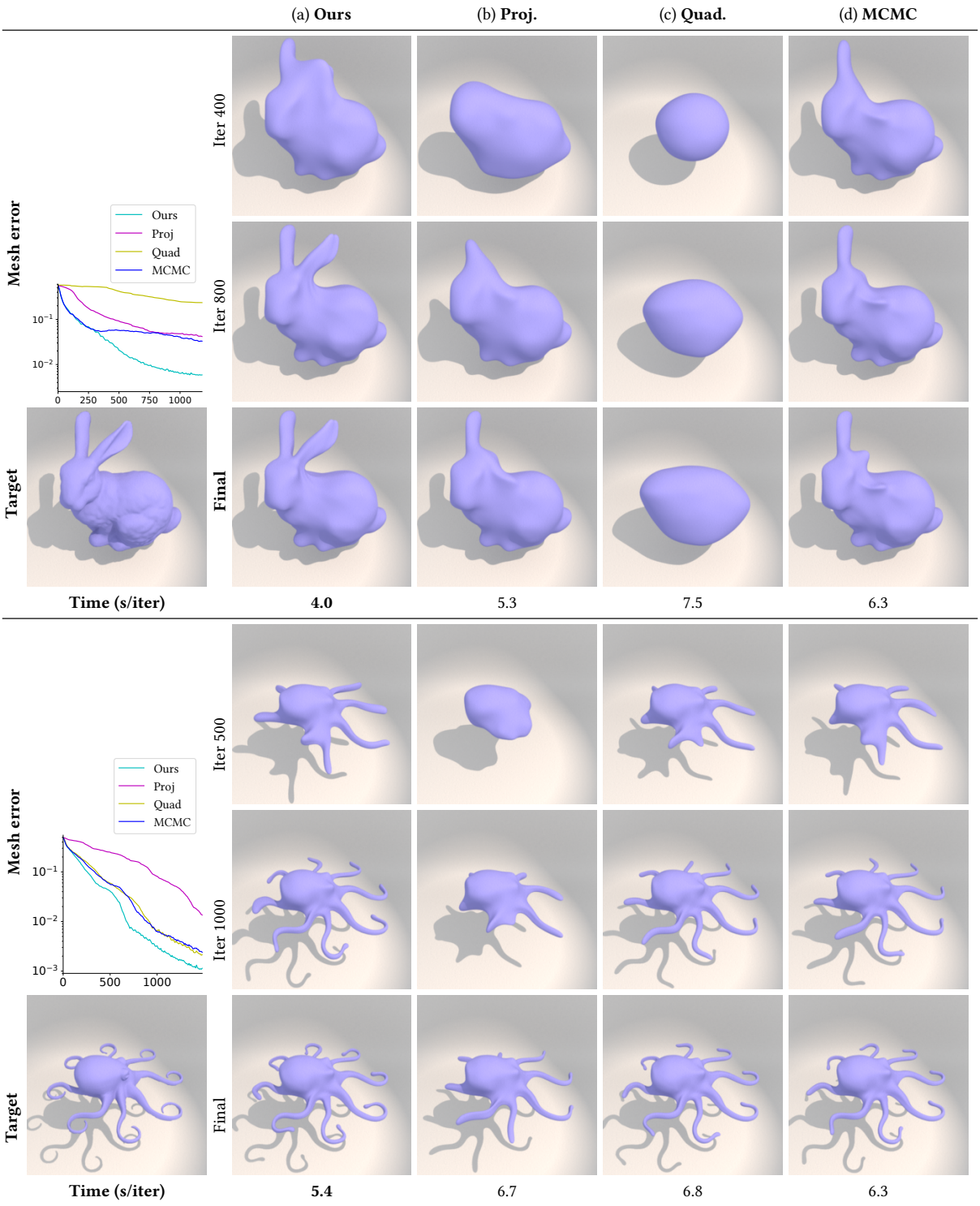

Fig. 12. We compare **inverse-rendering results** generated using our method (a), projective sampling [Zhang et al. 2023] (b), Quadric-based Sampling [Soroka et al. 2025] (c), and MCMC-based Sampling [Xu et al. 2024] (d) under the mirror configuration. The mesh-error plot reports the Chamfer distance between the optimized mesh and the target shape as a function of iteration count. We also report per-iteration timings, with the fastest method shown in bold. For each example, we show two intermediate stages of the shape optimization and the final result. Our method converges faster than the baselines.

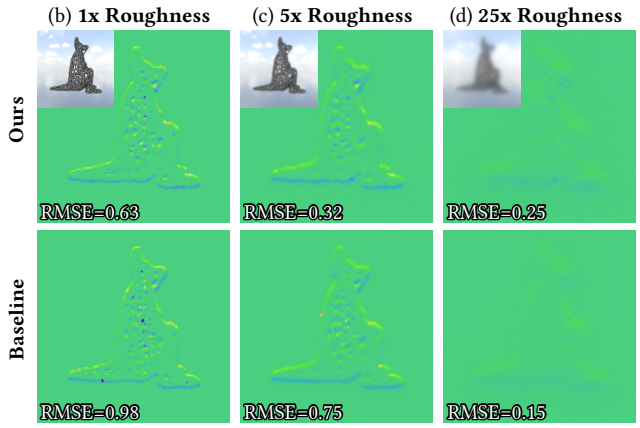

Fig. 13. We illustrate a limitation of our method by comparing gradient images against the projective-sampling baseline [Zhang et al. 2023] on the mirror setup with increasing surface roughness. Columns (b)–(d) show the gradient image and RMSE relative to the finite-difference reference at each roughness level. Our advantage over the baseline diminishes as roughness grows.

ACKNOWLEDGMENTS

We thank the anonymous reviewers for their constructive suggestions. This work was partially supported by NSF grant 2553564.

REFERENCES

Sai Praveen Bangaru, Tzu-Mao Li, and Frédo Durand. 2020. Unbiased Warped-Area Sampling for Differentiable Rendering. *ACM Trans. Graph.* 39, 6 (2020), 245:1–245:18.

Benedikt Bitterli and Wojciech Jarosz. 2017. Beyond Points and Beams: Higher-Dimensional Photon Samples for Volumetric Light Transport. *ACM Transactions on Graphics (Proceedings of SIGGRAPH)* 36, 4 (July 2017). https://doi.org/10/gfznbr

Wesley Chang, Venkataram Sivaram, Derek Nowrouzezahrai, Toshiya Hachisuka, Ravi Ramamoorthi, and Tzu-Mao Li. 2023. Parameter-space ReSTIR for Differentiable and Inverse Rendering. In *ACM SIGGRAPH 2023 Conference Proceedings (SIGGRAPH '23)*. Association for Computing Machinery, New York, NY, USA, 18:1–18:10.

Zhimin Fan, Pengcheng Shi, Mufan Guo, Ruoyu Fu, Yanwen Guo, and Jie Guo. 2024. Conditional Mixture Path Guiding for Differentiable Rendering. *ACM Trans. Graph.* 43, 4 (jul 2024). https://doi.org/10.1145/3658133

Jeppe Revall Frisvad, Lars Schjøth, Kenny Erleben, and Jon Sporring. 2014. Photon differential splatting for rendering caustics. In *Computer Graphics Forum*, Vol. 33. Wiley Online Library, 252–263.

Iliyan Georgiev, Jaroslav Křivánek, Tomáš Davidovič, and Philipp Slusallek. 2012. Light Transport Simulation with Vertex Connection and Merging. *ACM Trans. Graph.* 31, 6, Article 192 (Nov. 2012), 10 pages. https://doi.org/10.1145/2366145.2366211

Adrien Gruson, Binh-Son Hua, Nicolas Vibert, Derek Nowrouzezahrai, and Toshiya Hachisuka. 2018. Gradient-domain volumetric photon density estimation. *ACM Transactions on Graphics (TOG)* 37, 4 (2018), 1–13.

Toshiya Hachisuka and Henrik Wann Jensen. 2009. Stochastic Progressive Photon Mapping. *ACM Trans. Graph.* 28, 5, Article 141 (Dec. 2009), 8 pages. https://doi.org/10.1145/1661412.1618487

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

Toshiya Hachisuka, Jacopo Pantaleoni, and Henrik Wann Jensen. 2012. A path space extension for robust light transport simulation. *ACM Trans. Graph.* 31, 6, Article 191 (Nov. 2012), 10 pages. https://doi.org/10.1145/2366145.2366210

Binh-Son Hua, Adrien Gruson, Derek Nowrouzezahrai, and Toshiya Hachisuka. 2017. Gradient-domain photon density estimation. In *Computer Graphics Forum*, Vol. 36. Wiley Online Library, 31–38.

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

Wojciech Jarosz, Derek Nowrouzezahrai, Iman Sadeghi, and Henrik Wann Jensen. 2011a. A Comprehensive Theory of Volumetric Radiance Estimation Using Photon Points and Beams. *ACM Transactions on Graphics* 30, 1 (2011), 5:1–5:19. https://doi.org/10.1145/1899404.1899409

Wojciech Jarosz, Derek Nowrouzezahrai, Robert Thomas, Peter-Pike Sloan, and Matthias Zwicker. 2011b. Progressive Photon Beams. *ACM Transactions on Graphics (Proceedings of SIGGRAPH Asia)* 30, 6 (Dec. 2011). https://doi.org/10/fn5xzj

Wojciech Jarosz, Matthias Zwicker, and Henrik Wann Jensen. 2008. The Beam Radiance Estimate for Volumetric Photon Mapping. *Computer Graphics Forum (Proceedings of Eurographics)* 27, 2 (April 2008), 557–566. https://doi.org/10/bjsfsx

Henrik Wann Jensen. 1996. Global Illumination using Photon Maps. In *Rendering Techniques '96*. Springer, 21–30.

Simon Kallweit, Petrik Clarberg, Craig Kolb, Tom'aš Davidovič, Kai-Hwa Yao, Theresa Foley, Yong He, Lifan Wu, Lucy Chen, Tomas Akenine-Möller, Chris Wyman, Cyril Crassin, and Nir Benty. 2022. The Falcor Rendering Framework. https://github.com/NVIDIAGameWorks/Falcor https://github.com/NVIDIAGameWorks/Falcor.

Claude Knaus and Matthias Zwicker. 2011. Progressive Photon Mapping: A Probabilistic Approach. *ACM Transactions on Graphics* 30, 4 (July 2011), 25:1–25:13. https://doi.org/10.1145/2010324.1964920

Tzu-Mao Li, Miika Aittala, Frédo Durand, and Jaakko Lehtinen. 2018. Differentiable Monte Carlo ray tracing through edge sampling. *ACM Trans. Graph.* 37, 6 (2018), 222:1–222:11.

Guillaume Loubet, Nicolas Holzschuch, and Wenzel Jakob. 2019. Reparameterizing discontinuous integrands for differentiable rendering. *ACM Transactions on Graphics (TOG)* 38, 6 (2019), 1–14.

William Moses and Valentin Churavy. 2020. Instead of Rewriting Foreign Code for Machine Learning, Automatically Synthesize Fast Gradients. In *Advances in Neural Information Processing Systems*, H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin (Eds.), Vol. 33. Curran Associates, Inc., 12472–12485.

Romuald Perrot, Lilian Aveneau, Frédéric Mora, and Daniel Meneveaux. 2019. Photon mapping with visible kernel domains. *The Visual Computer* 35, 5 (May 2019). https://doi.org/10.1007/s00371-018-1505-y

Hao Qin, Xin Sun, Qiming Hou, Baining Guo, and Kun Zhou. 2015. Unbiased photon gathering for light transport simulation. *ACM Trans. Graph.* 34, 6, Article 208 (Nov. 2015), 14 pages. https://doi.org/10.1145/2816795.2818119

Osborne Reynolds. 1903. *Papers on mechanical and physical subjects: the sub-mechanics of the universe*. Vol. 3. The University Press.

Lars Schjøth, Jeppe Revall Frisvad, Kenny Erleben, and Jon Sporring. 2007. Photon differentials. In *Proceedings of the 5th international conference on Computer graphics and interactive techniques in Australia and Southeast Asia*. 179–186.

Lars Schjøth, Jeppe Revall Frisvad, Kenny Erleben, and Jon Sporring. 2010. Temporal photon differentials. In *International Conference on Computer Graphics Theory and Applications*, Vol. 2. SCITEPRESS, 54–61.

Mariia Soroka, Christoph Peters, and Steve Marschner. 2025. Quadric-Based Silhouette Sampling for Differentiable Rendering. *ACM Trans. Graph.* 44, 4 (July 2025). https://doi.org/10.1145/3731146

Eric Veach. 1997. *Robust Monte Carlo methods for light transport simulation*. Vol. 1610. Stanford University PhD thesis.

Yu-Chen Wang, Chris Wyman, Lifan Wu, and Shuang Zhao. 2023. Amortizing Samples in Physics-Based Inverse Rendering Using ReSTIR. *ACM Trans. Graph.* 42, 6 (2023), 214:1–214:17.

Zichen Wang, Xi Deng, Ziyi Zhang, Wenzel Jakob, and Steve Marschner. 2024. A Simple Approach to Differentiable Rendering of SDFs. (2024), 119:1–119:11.

Lifan Wu, Nathan Morrical, Sai Bangaru, Rohan Sawhney, Shuang Zhao, Chris Wyman, Ravi Ramamoorthi, and Aaron Lefohn. 2025. Unbiased Differential Visibility Using Fixed-Step Walk-on-Spherical-Caps and Closest Silhouettes. 44, 4 (August 2025), 79:1–79:16. https://doi.org/10.1145/3731174

Jiankai Xing, Xuejun Hu, Fujun Luan, Ling-Qi Yan, and Kun Xu. 2023. Extended Path Space Manifolds for Physically Based Differentiable Rendering. In *SIGGRAPH Asia 2023 Conference Papers*. 1–11.

Jiankai Xing, Zengyu Li, Fujun Luan, and Kun Xu. 2024. Differentiable Photon Mapping using Generalized Path Gradients. *ACM Transactions on Graphics (TOG)* 43, 6 (2024), 1–15.

Jiankai Xing, Fujun Luan, Ling-Qi Yan, Xuejun Hu, Houde Qian, and Kun Xu. 2022. Differentiable Rendering using RGBXY Derivatives and Optimal Transport. *ACM Trans. Graph.* 41, 6, Article 189 (dec 2022), 13 pages. https://doi.org/10.1145/3550454.3555479

Peiyu Xu, Sai Bangaru, Tzu-Mao Li, and Shuang Zhao. 2023. Warped-Area Reparameterization of Differential Path Integrals. *ACM Trans. Graph.* 42, 6 (2023), 213:1–213:18.

Peiyu Xu, Sai Bangaru, Tzu-Mao Li, and Shuang Zhao. 2024. Markov-Chain Monte Carlo Sampling of Visibility Boundaries for Differentiable Rendering. In *SIGGRAPH Asia 2024 Conference Papers*. 118:1–118:11.

Kai Yan, Christoph Lassner, Brian Budge, Zhao Dong, and Shuang Zhao. 2022. Efficient estimation of boundary integrals for path-space differentiable rendering. *ACM Trans. Graph.* 41, 4 (2022), 123:1–123:13.

Tizian Zeltner, Sébastien Speierer, Iliyan Georgiev, and Wenzel Jakob. 2021. Monte Carlo estimators for differential light transport. *ACM Trans. Graph.* 40, 4 (2021), 78:1–78:16.

Cheng Zhang, Zhao Dong, Michael Doggett, and Shuang Zhao. 2021. Antithetic sampling for Monte Carlo differentiable rendering. *ACM Trans. Graph.* 40, 4 (2021), 77:1–77:12.
Cheng Zhang, Bailey Miller, Kai Yan, Ioannis Gkioulekas, and Shuang Zhao. 2020. Path-space differentiable rendering. *ACM Trans. Graph.* 39, 4 (2020), 143:1–143:19.
Cheng Zhang, Lifan Wu, Changxi Zheng, Ioannis Gkioulekas, Ravi Ramamoorthi, and Shuang Zhao. 2019. A differential theory of radiative transfer. *ACM Trans. Graph.* 38, 6 (2019), 227:1–227:16.
Ziyi Zhang, Nicolas Roussel, and Wenzel Jakob. 2023. Projective Sampling for Differentiable Rendering of Geometry. *ACM Trans. Graph.* 42, 6 (2023), 212:1–212:14.

A PROOF OF CONSISTENCY

In order to prove the consistency of our method, we analyze the error of the estimator in Eq. (20) relative to the true boundary integral I_{bnd} in Eq. (5) of the main paper. We restrict our attention to paths with exactly N segments; the general result follows by summing over N. Denote the true and approximated boundary integral, respectively from Eqs. (8) and (14), as

$$I^N_{\text{bnd}} = \int_{\mathcal{B}^{N+1}} f(\bar{\boldsymbol{p}})\, V(\boldsymbol{p}_K)\, \delta_{\Delta\mathcal{B}(\boldsymbol{p}_{K+1})}(\boldsymbol{p}_K)\, \mathrm{d}\mu(\bar{\boldsymbol{p}}), \tag{23}$$

$$\tilde{I}^N_{\text{bnd}} = \int_{\mathcal{B}^{N+1}} f(\bar{\boldsymbol{p}}) \left(\frac{1}{M} \sum_{i=1}^{M} V(\boldsymbol{\mu}_{K,i})\, \frac{\mathcal{K}(\boldsymbol{p}_K; \boldsymbol{\mu}_{K,i})}{P_{K,i}} \right) \mathrm{d}\mu(\bar{\boldsymbol{p}}), \tag{24}$$

and the Monte-Carlo estimator as $\langle \tilde{I}^N_{\text{bnd}} \rangle$. Hence, the error of our estimator for paths with N segments can be written as

$$\mathbb{E}\left[\langle \tilde{I}^N_{\text{bnd}} \rangle\right] - I^N_{\text{bnd}} = \left(\mathbb{E}\left[\langle \tilde{I}^N_{\text{bnd}} \rangle\right] - \tilde{I}^N_{\text{bnd}}\right) + \left(\tilde{I}^N_{\text{bnd}} - I^N_{\text{bnd}}\right). \tag{25}$$

On the right-hand side, the first term captures the error of the Monte Carlo and kernel-density estimation, while the second term captures the bias introduced by softening the Dirac delta in Eq. (10). In the following, we show that both terms vanish as $r \to 0$.

We start by considering $\tilde{I}^N_{\text{bnd}} - I^N_{\text{bnd}}$. For brevity, we parameterize the boundary point as $\boldsymbol{p} = \boldsymbol{q}(s,t)$ and collect all remaining path variables into $\bar{\boldsymbol{p}}$. Then the smoothed integral becomes

$$I = \int_{\bar{\boldsymbol{p}}} \left(\iint f(s,t,\bar{\boldsymbol{p}})\, \mathcal{K}_1(t) \left(\int V(s',\bar{\boldsymbol{p}})\, \mathcal{K}_1(s'-s)\, \mathrm{d}s' \right) \mathrm{d}t\, \mathrm{d}s \right) \mathrm{d}\mu(\bar{\boldsymbol{p}}), \tag{26}$$

with Eq. (15) being the Monte Carlo version of this equation.

For any smooth function $g(u)$ and a symmetric, normalized 1-D kernel $\mathcal{K}_1$, the error of approximating $g(0)$ by its convolution with the kernel is given by $\int_{\mathbb{R}} g(u)\mathcal{K}_1(u)\,\mathrm{d}u - g(0)$. We introduce a canonical kernel $\mathcal{K}_{1,r}(u) = \frac{1}{r}\mathcal{K}_1(\frac{u}{r})$ to explicitly take into account the size of the kernel. Applying Taylor's expansion to g gives us

$$\int_{\mathbb{R}} g(u)\mathcal{K}_1(u)\,\mathrm{d}u = g(0) + \frac{1}{2}\int u^2 g''(\xi_u)\mathcal{K}_{1,r}(u)\,\mathrm{d}u. \tag{27}$$

Hence, for g with bounded second-order derivative and $\mathcal{K}$ with bounded second moment,

$$\begin{aligned} \left| \int_{\mathbb{R}} g(u)\mathcal{K}_1(u)\,\mathrm{d}u - g(0) \right| &\le \frac{C}{2}\int u^2 \left|\mathcal{K}_{1,r}(u)\right| \mathrm{d}u \\ &\overset{v=\frac{u}{r}}{=} \frac{C}{2} r^2 \int v^2 \left|\mathcal{K}_1(v)\right| \mathrm{d}v \\ &= O(r^2). \end{aligned} \tag{28}$$

Therefore, as the kernel size shrinks, the smoothing error decays at an $O(r^2)$ rate. Since the smoothness and boundedness assumptions are satisfied in the differentiable rendering setting, applying this bound to both the s- and t-directions in Eq. (26) shows that $(\tilde{I}^N_{\text{bnd}} - I^N_{\text{bnd}}) \to 0$ as $r \to 0$.

Next, we turn to $\mathbb{E}[\langle \tilde{I}^N_{\text{bnd}} \rangle] - \tilde{I}^N_{\text{bnd}}$. Since $\langle \tilde{I}^N_{\text{bnd}} \rangle$ is obtained by direct Monte Carlo estimation of $\tilde{I}^N_{\text{bnd}}$, the only source of bias is the re-normalization in Eq. (19). A key observation is that, for triangular meshes, this re-normalization is exact unless the kernel support overlaps an edge vertex. As the kernel size approaches 0, the probability that any sampled kernel covers a vertex vanishes, so $\mathbb{E}[\langle \tilde{I}^N_{\text{bnd}} \rangle] - \tilde{I}^N_{\text{bnd}} \to 0$ as $r \to 0$. This concludes that the Monte Carlo estimator converges to the true boundary integral as the radius shrinks to 0.

Lastly, our final estimator averages independent realizations of this consistent estimator, and therefore also converges to the true boundary integral as $T \to \infty$. This concludes the proof of the consistency of our algorithm.