Differentiating Variance for Variance-Aware Inverse Rendering

KAI YAN, University of California Irvine, United States of America and Weta FX, New Zealand
VINCENT PEGORARO, Weta FX, New Zealand
MARC DROSKE, Weta FX, New Zealand
JIŘÍ VORBA, Weta FX, New Zealand
SHUANG ZHAO, University of California Irvine, United States of America

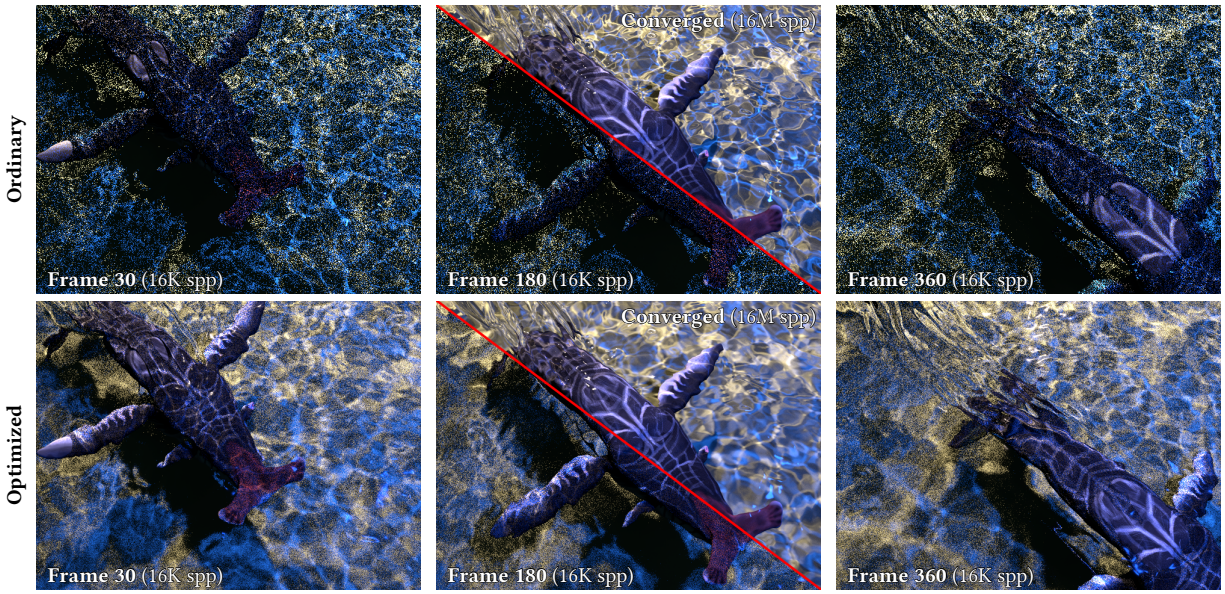


Fig. 1. We introduce a technique that estimates and differentiates the variance of Monte Carlo rendering processes. This example features an animated scene with a whale swimming underwater, lit by one environmental and two area lights. Based on our technique, we jointly optimize the roughness of the water surface as well as the sampling probability of the three light sources (using one frame of the animation). When rendered using a unidirectional path tracer with 16384 samples per pixel, the optimized configuration produces significantly better results. Converged renderings for Frame 180 demonstrate the roughening of the water surface produced by our optimization. Please refer to the supplement to view the full animations.

Monte Carlo methods have been widely adopted in physics-based rendering. A key property of a Monte Carlo estimator is its variance, which dictates the convergence rate of the estimator. In this paper, we devise a mathematical formulation for derivatives of rendering variance with respect to not only scene parameters (e.g., surface roughness) but also sampling probabilities. Based on this formulation, we introduce unbiased Monte Carlo estimators for those derivatives. Our theory and algorithm enable variance-aware inverse rendering which alters a virtual scene and/or an estimator in an optimal way to offer a good balance between bias and variance. We evaluate our technique using several synthetic examples.

CCS Concepts: • **Computing methodologies** → **Rendering**.

Additional Key Words and Phrases: Differentiable rendering, differential path integral, Monte Carlo variance

ACM Reference Format:
Kai Yan, Vincent Pegoraro, Marc Droske, Jiří Vorba, and Shuang Zhao. 2024. Differentiating Variance for Variance-Aware Inverse Rendering. In *SIGGRAPH Asia 2024 Conference Papers (SA Conference Papers '24), December 03–06, 2024, Tokyo, Japan.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3680528.3687603

Authors' Contact Information: Kai Yan, University of California Irvine, United States of America and Weta FX, New Zealand, kyan8@uci.edu; Vincent Pegoraro, Weta FX, New Zealand, vpegoraro@wetafx.co.nz; Marc Droske, Weta FX, New Zealand, mdroske@wetafx.co.nz; Jiří Vorba, Weta FX, New Zealand, jvorba@wetafx.co.nz; Shuang Zhao, University of California Irvine, United States of America, shz@ics.uci.edu.

SA Conference Papers '24, December 03–06, 2024, Tokyo, Japan

ACM ISBN 979-8-4007-1131-2/24/12
https://doi.org/10.1145/3680528.3687603

1 Introduction

Physically-based *forward* rendering focuses on numerically computing radiometric measurements in fully described virtual scenes. To this end, Monte Carlo methods (e.g., path tracing), which work by averaging contributions of randomly traced light paths have long been a "gold standard", thanks to their capabilities of accurately

simulating complex light transport effects (e.g., soft shadow and interreflection).

Differentiable rendering, on the other hand, concerns with estimating derivatives of forward-rendering results. Recently, great progress has been made in differentiable rendering [Li et al. 2018; Zhang et al. 2020; Bangaru et al. 2020], resulting in efficient and general-purpose differentiable rendering techniques capable of differentiating with respect to a wide range of scene parameters including material properties and object geometries.

A key property of any Monte Carlo forward or differentiable rendering technique is its *variance*—which largely determines the number of random samples needed for producing clean results. Consequently, variance reduction has been a central research topic for both forward and differentiable rendering [Zhang et al. 2021; Vicini et al. 2021; Zhang et al. 2023].

Due to the importance of variance, several methods [Durand 2011; Subr and Kautz 2013; Pilleboue et al. 2015]—most of which leverage Monte Carlo techniques themselves—have been introduced to estimate the variance of Monte Carlo rendering processes. On the other hand, estimating the derivatives of rendering variance has been largely under-explored—As we will demonstrate, naïvely applying automatic differentiation (AD) to (forward) variance estimation procedures generally yields incorrect results, also known as *bias in derivatives*.

In this paper, we bridge this gap by establishing a new differential formulation for rendering variance. An important application of our technique is *variance-aware inverse rendering*, which enables the altering of virtual scenes and/or Monte Carlo estimators to offer a good balance between *rendering variance* and *rendering bias*.

Concretely, our contributions include:

- Devising a novel mathematical formulation for derivatives of rendering variances with respect to both scene parameters and sampling probabilities (§4);
- Discussing the estimation and differentiation of variance-aware losses and introducing the concept of variance-aware inverse rendering (§5).

We validate our technique by comparing our variance derivative estimates to references obtained using finite differences. Further, we demonstrate the usefulness of our technique using several synthetic inverse-rendering examples.

2 Related Works

Differentiable rendering. Recently, great progresses have been made in physics-based differentiable rendering. We discuss most relevant techniques in the following and refer the readers to online course materials (e.g., [Zhao et al. 2020]) for a more comprehensive overview.

A main challenge toward the development of general-purpose differentiable rendering techniques has been the differentiation with respect to scene geometry—which generally requires calculating additional boundary integrals.

To address this problem, two classes of techniques have been introduced. The first class directly samples discontinuity boundaries [Li et al. 2018; Zhang et al. 2020]. Specifically, Zhang et al. [2020] have introduced the formulation of *differential path integrals* where discontinuities are tracked and handled at the path level. Recently, specialized methods [Yan et al. 2022; Zhang et al. 2023] have been developed to better importance sample these path integrals. The second class of methods [Loubet et al. 2019; Bangaru et al. 2020; Xu et al. 2023] reparameterize boundary integrals and avoid explicit handling of discontinuity boundaries altogether.

Unfortunately, most, if not all, existing differentiable rendering techniques focus on derivatives of (converged) rendering estimates (e.g., pixel values) rather than their variances. In this paper, we bridge this gap by adopting the formulation by Zhang et al. [2020] to establish a general mathematical framework for the derivatives of rendering variances.

Variance-aware forward rendering. Variance reduction has been a long-standing topic in Monte Carlo forward rendering. Among the very large body of works on this topic, several recent methods leverage variance information to enhance various aspects including multiple importance sampling [Grittmann et al. 2019] and path guiding [Rath et al. 2020]. Our technique is largely orthogonal to these methods.

Path-space regularization. As a potential application of our theory, path-space regularization concerns with modifying path sampling strategies so that rendering variance is greatly reduced while introducing minimal bias. Previously, several techniques [Kaplanyan and Dachsbacher 2013; Jendersie and Grosch 2019] have been introduced to regularize the sampling of micro-facet distributions by leveraging strategic use of mollifiers for Dirac deltas.

The path-space regularization technique that is most relevant to ours is the one by Weier et al. [2021]—which reduces rendering variance by altering surface roughness at selected vertices of each light path. This technique formulates the search of optimal roughness values as a variance-aware inverse rendering problem and leverages differentiable rendering to solve the optimization efficiently. Unfortunately, they rely on highly biased variance estimates which, as we will show in §4 and §6, can lead to unreliable optimization.

3 Preliminaries

We now revisit some preliminaries in Monte Carlo rendering (§3.1) and the estimation of rendering variance (§3.2).

3.1 Path Integrals

In physics-based (forward) rendering, the response of a radiometric detector is typically formulated as a **path integral** [Veach 1997] of the form:

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

where:

- $\bar{\boldsymbol{x}} = (\boldsymbol{x}_0, \boldsymbol{x}_1, \ldots, \boldsymbol{x}_N)$ is a **light path** where $\boldsymbol{x}_0$ and $\boldsymbol{x}_N$ reside on a light source and the detector, respectively;
- Ω is the **path space** comprised of all finite-length light paths;
- μ is the area-product measure.

Additionally, f is the **measurement contribution function** defined as

$$f(\bar{\boldsymbol{x}}) = L_\mathrm{e}(\boldsymbol{x}_0 \to \boldsymbol{x}_1)\left(\prod_{n=0}^{N-1} G(\boldsymbol{x}_n \leftrightarrow \boldsymbol{x}_{n+1})\right)\left(\prod_{n=1}^{N-1} f_\mathrm{s}(\boldsymbol{x}_{n-1} \to \boldsymbol{x}_n \to \boldsymbol{x}_{n+1})\right) W_\mathrm{e}(\boldsymbol{x}_{N-1} \to \boldsymbol{x}_N), \quad (2)$$

where: L_e and W_e denote the **source emission** and **detector response** functions, respectively; G is the **geometric term**; f_s indicates the **bidirectional scattering distribution function** (BSDF).

Based on the path-integral formulation (1), the detector response can be estimated using Monte Carlo estimation in an unbiased fashion via

$$\langle I \rangle := \frac{f(\bar{\boldsymbol{x}})}{q(\bar{\boldsymbol{x}})}, \quad (3)$$

where $\bar{\boldsymbol{x}}$ is light path randomly drawn with the probability density q (such that $q(\bar{\boldsymbol{x}}) > 0$ for all $\bar{\boldsymbol{x}}$ with $f(\bar{\boldsymbol{x}}) \neq 0$).

3.2 Estimating Rendering Variance

The variance $\mathbb{V}[\langle I \rangle]$ of the estimator $\langle I \rangle$ from Eq. (3) satisfies that [Pegoraro 2016, §4.4.6]

$$\mathbb{V}[\langle I \rangle] = \mathbb{E}\left[\langle I \rangle^2\right] - \mathbb{E}^2[\langle I \rangle] = \mathbb{E}\left[\langle I \rangle^2\right] - I^2, \quad (4)$$

where the second moment $\mathbb{E}[\langle I \rangle^2]$ is given by

$$\mathbb{E}\left[\langle I \rangle^2\right] = \int_\Omega \left(\frac{f(\bar{\boldsymbol{x}})}{q(\bar{\boldsymbol{x}})}\right)^2 q(\bar{\boldsymbol{x}})\,\mathrm{d}\mu(\bar{\boldsymbol{x}}) = \int_\Omega h(\bar{\boldsymbol{x}})\,\mathrm{d}\mu(\bar{\boldsymbol{x}}), \quad (5)$$

with f being the measurement contribution defined in Eq. (2) and

$$h(\bar{\boldsymbol{x}}) := \frac{f^2(\bar{\boldsymbol{x}})}{q(\bar{\boldsymbol{x}})}. \quad (6)$$

Given Eq. (5), the second moment $\mathbb{E}[\langle I \rangle^2]$ can be estimated using

$$\left\langle \mathbb{E}\left[\langle I \rangle^2\right] \right\rangle = \frac{h(\bar{\boldsymbol{x}})}{q(\bar{\boldsymbol{x}})} = \frac{f^2(\bar{\boldsymbol{x}})}{q^2(\bar{\boldsymbol{x}})}, \quad (7)$$

where $\bar{\boldsymbol{x}}$ is a path randomly drawn with the probability density q. Further, when the first moment $\mathbb{E}[\langle I \rangle] = I$ is known, the variance $\mathbb{V}[\langle I \rangle]$ can be estimated using

$$\langle \mathbb{V}[\langle I \rangle] \rangle = \left\langle \mathbb{E}\left[\langle I \rangle^2\right] \right\rangle - I^2 = \frac{f^2(\bar{\boldsymbol{x}})}{q^2(\bar{\boldsymbol{x}})} - I^2. \quad (8)$$

On the other hand, when I itself needs to be estimated using Eq. (3), we will need two *independent* path samples $\bar{\boldsymbol{x}}_1, \bar{\boldsymbol{x}}_2$ to obtain an unbiased estimate of I^2. When reusing these samples for the second moment $\mathbb{E}[\langle I \rangle^2]$, we obtain the following Monte Carlo estimator:

$$\langle \mathbb{V}[\langle I \rangle] \rangle = \frac{1}{2}\left(\frac{f^2(\bar{\boldsymbol{x}}_1)}{q^2(\bar{\boldsymbol{x}}_1)} + \frac{f^2(\bar{\boldsymbol{x}}_2)}{q^2(\bar{\boldsymbol{x}}_2)}\right) - \frac{f(\bar{\boldsymbol{x}}_1)}{q(\bar{\boldsymbol{x}}_1)}\frac{f(\bar{\boldsymbol{x}}_2)}{q(\bar{\boldsymbol{x}}_2)}. \quad (9)$$

4 Differentiating Rendering Variance

Let $\langle I \rangle$ be a Monte Carlo estimator defined in Eq. (3) and $\theta \in \mathbb{R}$ be a parameter[1] controlling the scene (that is being rendered by $\langle I \rangle$) or the path sampling process. We consider the problem of differentiating the rendering variance $\mathbb{V}[\langle I \rangle]$ defined in Eq. (4) with respect to θ, which largely amounts to deriving the derivatives of I and the second moment $\mathbb{E}[\langle I \rangle^2]$:

$$\boxed{\partial_\theta\, \mathbb{V}\left[\langle I \rangle\right] = -2I\left(\partial_\theta I\right) + \left(\partial_\theta\, \mathbb{E}\left[\langle I \rangle^2\right]\right).} \quad (10)$$

Smoothness assumptions. To facilitate the derivation of the derivatives $\partial_\theta I$ and $\partial_\theta\, \mathbb{E}[\langle I \rangle^2]$, we assume the source emission $L_\mathrm{e}(\boldsymbol{x}_0 \to \boldsymbol{x}_1)$, detector response $W_\mathrm{e}(\boldsymbol{x}_{N-1} \to \boldsymbol{x}_N)$, and BSDF $f_\mathrm{s}(\boldsymbol{x}_{n-1} \to \boldsymbol{x}_n \to \boldsymbol{x}_{n+1})$ to all be C^0 with respect to the path vertices $\boldsymbol{x}_0, \boldsymbol{x}_1, \ldots, \boldsymbol{x}_N$. This assumption has been made by most, if not all, state-of-the-art differentiable rendering techniques (e.g., [Zhang et al. 2020; Bangaru et al. 2020; Vicini et al. 2021, 2022]).

In addition, we assume that the probability density q is C^0 except across visibility boundaries. This is a mild assumption that usually holds in practice, and we will further discuss in §4.2.

4.1 Our Derivation

We now derive Eq. (10) when the parameter $\theta \in \mathbb{R}$ controls the scene (e.g., the roughness of a surface or the size of an area light) and/or the sampling process (e.g., the probability mass for sampling a specific light source). In general, θ can affect the measurement contribution f and the path space Ω in Eq. (1), as well as the probability q in Eq. (3).

Material-form reparameterization. When the path space Ω depends on θ, differentiating path integrals over this space with respect to θ becomes more challenging since discontinuities of the measurement contribution can evolve with θ. To address this problem, Zhang et al. [2020] have proposed to reparameterize the evolving path space $\Omega(\theta)$ with a fixed **reference path space** $\hat{\Omega}$ using a predetermined one-to-one mapping $\bar{\mathsf{X}}(\theta) : \hat{\Omega} \mapsto \Omega(\theta)$ which transforms each **material path** $\bar{\boldsymbol{p}} \in \hat{\Omega}$ into a **spatial path** $\bar{\boldsymbol{x}} = \bar{\mathsf{X}}(\bar{\boldsymbol{p}}, \theta) \in \Omega(\theta)$.

To this end, given a material path $\bar{\boldsymbol{p}} = (\boldsymbol{p}_0, \ldots, \boldsymbol{p}_N)$, its spatial representation $\bar{\boldsymbol{x}} = \bar{\mathsf{X}}(\bar{\boldsymbol{p}}, \theta) = (\boldsymbol{x}_0, \ldots, \boldsymbol{x}_N)$ is obtained by transforming each path vertex using $\boldsymbol{x}_n = \mathsf{X}(\boldsymbol{p}_n, \theta)$, where $\mathsf{X}(\cdot, \theta)$ is a one-to-one mapping from some *reference surface* (which defines the fixed reference space $\hat{\Omega}$) to the *evolving surface* of the scene (which defines the path space $\Omega(\theta)$).

In practice, when evaluating derivatives at $\theta = \theta_0$ (for some fixed θ_0), the reference surface is normally chosen as the evolving surface at $\theta = \theta_0$. This causes the reference path space $\hat{\Omega}$ to coincide with evolving one Ω at $\theta = \theta_0$: That is, $\hat{\Omega} = \Omega(\theta_0)$. For more details, please refer to the work by Zhang et al. [2020].

Differentiating I. The derivative $\partial_\theta I$ can be obtained using the differential path integral formulation [Zhang et al. 2020]—which we briefly revisit in the following for completeness.

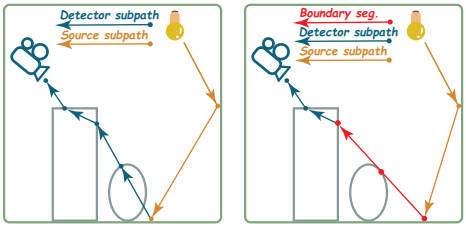
Applying the material-form reparameterization to the path integral in Eq. (1) produces[2]

$$I = \int_{\hat{\Omega}} \underbrace{f(\bar{\mathsf{X}}(\bar{\boldsymbol{p}}, \theta))\, J_{\bar{\mathsf{X}}}(\bar{\boldsymbol{p}})}_{=: \hat{f}(\bar{\boldsymbol{p}})}\,\mathrm{d}\mu(\bar{\boldsymbol{p}}), \quad (11)$$

where $J_{\bar{\mathsf{X}}}$ is the Jacobian term emerging from the change of variable $\bar{\boldsymbol{x}} = \bar{\mathsf{X}}(\bar{\boldsymbol{p}}, \theta)$.

[1] We assume the parameter θ to be scalar-valued for notational simplicity. All results derived in this paper generalize naturally to vector-valued parameters.

[2] We omit the dependencies of $J_{\bar{\mathsf{X}}}$, $\hat{f}$, and $\hat{h}$ on the parameter θ for notational simplicity.



(a) Ordinary path (b) Boundary path

Fig. 2. Ordinary and boundary paths.

The derivative of Eq. (11) with respect to θ can be expressed as *differential path integrals*:

$$\partial_\theta I = \underbrace{\int_{\hat{\Omega}} \partial_\theta \hat{f}(\bar{\boldsymbol{p}})\, \mathrm{d}\mu(\bar{\boldsymbol{p}})}_{\text{interior}} + \underbrace{\int_{\partial\hat{\Omega}} \hat{f}(\bar{\boldsymbol{p}})\, V(\bar{\boldsymbol{p}})\, \mathrm{d}\dot{\mu}(\bar{\boldsymbol{p}})}_{\text{boundary}}, \tag{12}$$

where the *interior* integral is defined over the same referenced path space $\hat{\Omega}$ as Eq. (11).

The *boundary* integral in Eq. (12) is defined over the **boundary path space** $\partial\hat{\Omega}$ comprised of **material boundary paths** each containing exact one **boundary segment** (see Figure 2). For a boundary path $\bar{\boldsymbol{p}}$ with the boundary segment $\overline{\boldsymbol{p}_{K-1}\,\boldsymbol{p}_K}$, the vertex $\boldsymbol{p}_K$ is constrained over discontinuity boundaries[3] of the mutual visibility $\mathbb{V}(\mathsf{X}(\boldsymbol{p}_{K-1}, \theta) \leftrightarrow \mathsf{X}(\boldsymbol{p}_K, \theta))$ with $\boldsymbol{p}_{K-1}$ fixed, and $V(\bar{\boldsymbol{p}})$ captures the "normal velocity" of these boundary curves at $\boldsymbol{p}_K$ (with respect to θ). Please refer to prior works [Zhang et al. 2020, 2023] for the exact forms of $V(\bar{\boldsymbol{p}})$.

Differentiating $\mathbb{E}[\langle I\rangle^2]$. We now discuss how Zhang et al.'s [2020] differential path integral formulation can be adopted for the second moment $\mathbb{E}[\langle I\rangle^2]$.

Leveraging the material-form reparameterization, we rewrite $\mathbb{E}[\langle I\rangle^2]$ in Eq. (5) as a *material-form path integral*:

$$\mathbb{E}\left[\langle I\rangle^2\right] = \int_{\hat{\Omega}} \underbrace{h\left(\bar{\mathsf{X}}(\bar{\boldsymbol{p}}, \theta)\right) J_{\bar{\mathsf{X}}}(\bar{\boldsymbol{p}})}_{=:\hat{h}(\bar{\boldsymbol{p}})}\, \mathrm{d}\mu(\bar{\boldsymbol{p}}), \tag{13}$$

which is identical to the forward rendering variant (11) except with the measurement contribution f replaced with h defined in Eq. (6).

Based on the assumption that the probability density q is C^0 except across visibility boundaries, we make an important observation that the function h shares the same θ-dependent discontinuities as f—which we will elaborate in §4.2. This allows us to use another form of differential path integrals to express the derivative of Eq. (13) with respect to θ:

$$\partial_\theta\, \mathbb{E}\left[\langle I\rangle^2\right] = \underbrace{\int_{\hat{\Omega}} \partial_\theta \hat{h}(\bar{\boldsymbol{p}})\, \mathrm{d}\mu(\bar{\boldsymbol{p}})}_{\text{interior}} + \underbrace{\int_{\partial\hat{\Omega}} \hat{h}(\bar{\boldsymbol{p}})\, V(\bar{\boldsymbol{p}})\, \mathrm{d}\dot{\mu}(\bar{\boldsymbol{p}})}_{\text{boundary}}, \tag{14}$$

[3]We assume without loss of generality that all visibility boundaries have normals pointing toward the occluded sides. When using normals toward the visible sides, the integrands of the *boundary* components in Eqs. (12) and (14) become $-\hat{f}(\bar{\boldsymbol{p}})\, V(\bar{\boldsymbol{p}})$ and $-\hat{h}(\bar{\boldsymbol{p}})\, V(\bar{\boldsymbol{p}})$, respectively.

which is identical to Eq. (12) except with $\hat{f}$ replaced by $\hat{h}$ for both the *interior* and the *boundary* components.

Monte Carlo estimation. With the derivatives in Eqs. (12) and (14) derived, the derivative in Eq. (10) at some $\theta = \theta_0$ can be estimated via:

$$\langle \partial_\theta\, \mathbb{V}\left[\langle I\rangle\right]\rangle = -2\, \frac{f(\bar{\boldsymbol{x}}_1)}{q_1(\bar{\boldsymbol{x}}_1)} \left(\frac{\partial_\theta \hat{f}(\bar{\boldsymbol{p}}_2)}{q_2(\bar{\boldsymbol{p}}_2)} + \frac{\hat{f}(\bar{\boldsymbol{p}}_3)\, V(\bar{\boldsymbol{p}}_3)}{q_3(\bar{\boldsymbol{p}}_3)} \right) + \left(\frac{\partial_\theta \hat{h}(\bar{\boldsymbol{p}}_2)}{q_2(\bar{\boldsymbol{p}}_2)} + \frac{\hat{h}(\bar{\boldsymbol{p}}_4)\, V(\bar{\boldsymbol{p}}_4)}{q_4(\bar{\boldsymbol{p}}_4)} \right), \tag{15}$$

where:

- $\bar{\boldsymbol{x}}_1$ is a (spatial) light path drawn from the path space Ω with the probability density q_1;
- $\bar{\boldsymbol{p}}_2$ is a material path drawn from the fixed reference path space $\hat{\Omega}$ (that is typically set to $\Omega(\theta_0)$) with the probability density q_2;
- $\bar{\boldsymbol{p}}_3$ and $\bar{\boldsymbol{p}}_4$ are boundary paths drawn from the boundary path space $\partial\hat{\Omega}$ with the probability densities q_3 and q_4, respectively.

In practice, we use unidirectional path tracing (PT) with next-event estimation (NEE) to sample the ordinary paths $\bar{\boldsymbol{x}}_1$ and $\bar{\boldsymbol{p}}_2$. The boundary paths $\bar{\boldsymbol{p}}_3$ and $\bar{\boldsymbol{p}}_4$, on the other hand, can be importance sampled using several recent techniques [Zhang et al. 2020; Yan et al. 2022; Zhang et al. 2023]. In this paper, we adopt the basic approach introduced by Zhang et al. [2020].

Simple case. When the parameter θ controls only material properties (e.g., surface roughness) but not object geometries, the *boundary* components in the differential path integrals (12, 14) vanish, leading to:

$$\partial_\theta I = \int_{\Omega} \partial_\theta f(\bar{\boldsymbol{x}})\, \mathrm{d}\mu(\bar{\boldsymbol{x}}), \tag{16}$$

$$\partial_\theta\, \mathbb{E}\left[\langle I\rangle^2\right] = \int_{\Omega} \partial_\theta h(\bar{\boldsymbol{x}})\, \mathrm{d}\mu(\bar{\boldsymbol{x}}). \tag{17}$$

This further causes the estimator in Eq. (15) to simplify to:

$$\langle \partial_\theta\, \mathbb{V}\left[\langle I\rangle\right]\rangle = -2\, \frac{f(\bar{\boldsymbol{x}}_1)}{q_1(\bar{\boldsymbol{x}}_1)}\, \frac{\partial_\theta f(\bar{\boldsymbol{x}}_2)}{q_1(\bar{\boldsymbol{x}}_2)} + \frac{\partial_\theta h(\bar{\boldsymbol{x}}_2)}{q_1(\bar{\boldsymbol{x}}_2)}, \tag{18}$$

where $\bar{\boldsymbol{x}}_1$ and $\bar{\boldsymbol{x}}_2$ are *independent* path samples drawn from q_1.

4.2 Path-Sampling Probability

A significant difference between our newly introduced differential path integrals (14) and the ordinary (12) is the need to differentiate the probability density $q(\bar{\boldsymbol{x}})$ for sampling a path $\bar{\boldsymbol{x}}$—which is a component of the function h defined in Eq. (6).

In what follows, we discuss the expression of q resulting from the widely adopted (unidirectional) path tracing (PT) with next-event estimation (NEE). Starting with the segment $(\boldsymbol{x}_N, \boldsymbol{x}_{N-1})$ with $\boldsymbol{x}_N$ on a detector, this process traces the path backward by iteratively adding vertices $\boldsymbol{x}_{N-2}, \boldsymbol{x}_{N-3}, \ldots$ until getting $\boldsymbol{x}_0$ on a light source. Then, the probability density $q(\bar{\boldsymbol{x}})$ of a path $\bar{\boldsymbol{x}} = (\boldsymbol{x}_0, \ldots, \boldsymbol{x}_N)$ satisfies that

$$q(\bar{\boldsymbol{x}}) = q(\boldsymbol{x}_N, \boldsymbol{x}_{N-1}) \left(\prod_{n=1}^{N-2} q^{\text{bsdf}}(\boldsymbol{x}_n \mid \boldsymbol{x}_{n+1}, \boldsymbol{x}_{n+2}) \right) q^{\text{light}}(\boldsymbol{x}_0 \mid \boldsymbol{x}_1), \tag{19}$$

where the joint probability $q(\boldsymbol{x}_N, \boldsymbol{x}_{N-1})$ emerges from camera-ray sampling, and the conditional probabilities $q^{\text{bsdf}}(\boldsymbol{x}_n \mid \boldsymbol{x}_{n+1}, \boldsymbol{x}_{n+2})$ and $q^{\text{light}}(\boldsymbol{x}_0 \mid \boldsymbol{x}_1)$ emerge from BSDF and light sampling, respectively. We note that all the probability densities on the right-hand side of Eq. (19) use the surface-area measure.

Continuity. We recall that our derivations in §4.1 assume that q is C^0 except across visibility boundaries. In practice, this is a mild assumption because:

- The probability $q^{\text{bsdf}}(\boldsymbol{x}_n \mid \boldsymbol{x}_{n+1}, \boldsymbol{x}_{n+2})$ resulting from BSDF sampling is normally C^0 for all $\boldsymbol{x}_n$ that is visible to $\boldsymbol{x}_{n+1}$ (i.e., with $\mathbb{V}(\boldsymbol{x}_n \leftrightarrow \boldsymbol{x}_{n+1}) = 1$).
- The probability $q^{\text{light}}(\boldsymbol{x}_0 \mid \boldsymbol{x}_1)$ given by light sampling—which typically amounts to drawing $\boldsymbol{x}_0$ from surfaces of light sources without considering the visibility $\mathbb{V}(\boldsymbol{x}_0 \leftrightarrow \boldsymbol{x}_1)$—is typically C^0 with respect to $\boldsymbol{x}_0$.

Therefore, θ-dependent discontinuities of the path-sampling probability $q(\bar{\boldsymbol{x}})$ in Eq. (19) are naturally aligned with those of the measurement contribution f, allowing the differentiation of Eq. (13) using Eq. (14).

Russian roulette. A common technique to avoid generating infinitely long light paths is *Russian roulette*: At each intermediate vertex $\boldsymbol{x}_n$ of a path, the sampling process conducts a binomial trial on the "survival" of the path with the probability $q^{\text{survival}}(\boldsymbol{x}_n)$. If the path survives, the tracing process continues and the next vertex $\boldsymbol{x}_{n-1}$ is drawn. Otherwise, the sampling process terminates, and the (partially sampled) path is discarded. This results in the following probability density:

$$q_{\text{RR}}(\bar{\boldsymbol{x}}) \coloneqq q(\bar{\boldsymbol{x}}) \prod_{n=1}^{N-1} q^{\text{survival}}(\boldsymbol{x}_n), \tag{20}$$

where $q(\bar{\boldsymbol{x}})$ is defined in Eq. (19).

Several heuristics [Vorba and Křivánek 2016; Rath et al. 2022] have been introduced in forward rendering to set the survival probabilities q^{survival}. Our technique is largely orthogonal to these methods and validated in Fig.4.

Multiple importance sampling. Another commonly adopted technique for robust Monte Carlo rendering is multiple importance sampling (MIS). When using M path sampling strategies with corresponding probability densities $q_1, q_2, \ldots, q_M$, an MIS estimator takes the form:

$$\langle I \rangle_{\text{MIS}} = \sum_{m=1}^{M} w_m(\bar{\boldsymbol{x}}_m) \frac{f(\bar{\boldsymbol{x}}_m)}{q_m(\bar{\boldsymbol{x}}_m)}, \tag{21}$$

where $\bar{\boldsymbol{x}}_m$ is a light path drawn from q_m, and w_m is a nonnegative weight function satisfying $\sum_{m=1}^{M} w_m(\bar{\boldsymbol{x}}) = 1$ for all $\bar{\boldsymbol{x}}$.

In practice, a unidirectional path tracer typically use $M = 2$ strategies that are identical except for the sampling of the light vertex $\boldsymbol{x}_0$ where they use light and BSDF sampling, respectively.

The second moment of this MIS estimator in Eq. (21) equals

$$\mathbb{E}\left[\langle I \rangle_{\text{MIS}}^2\right] = \int_{\Omega^M} h_{\text{MIS}}(\bar{\boldsymbol{x}}_1, \ldots, \bar{\boldsymbol{x}}_M) \prod_{m=1}^{M} \mathrm{d}\mu(\bar{\boldsymbol{x}}_m), \tag{22}$$

where

$$h_{\text{MIS}}(\bar{\boldsymbol{x}}_1, \ldots, \bar{\boldsymbol{x}}_M) \coloneqq \left(\sum_{m=1}^{M} w_m(\bar{\boldsymbol{x}}_m) \frac{f(\bar{\boldsymbol{x}}_m)}{q_m(\bar{\boldsymbol{x}}_m)} \right)^2 q(\bar{\boldsymbol{x}}_1, \ldots, \bar{\boldsymbol{x}}_M), \tag{23}$$

with $q(\bar{\boldsymbol{x}}_1, \ldots, \bar{\boldsymbol{x}}_M)$ denoting the joint probability for sampling light paths $\bar{\boldsymbol{x}}_1, \ldots, \bar{\boldsymbol{x}}_M$.

Lastly, the derivative $\partial_\theta \mathbb{E}[\langle I \rangle_{\text{MIS}}^2]$ of Eq. (22) can be obtained using the same approach described in §4.1.

Relation With Prior Work

Previously, Weier et al. [2021] have also attempted to differentiate rendering variance (4) with respect to surface roughness. Their publicly available implementation [Weier 2021] is built upon Mitsuba3 [Jakob et al. 2022] and uses the moment integrator—which implements Eq. (7)—to obtain unbiased estimates of the second moment $\mathbb{E}[\langle I \rangle^2]$.

When estimating the derivative $\partial_\theta \mathbb{E}[\langle I \rangle^2]$, Weier et al. [2021] have simply applied automatic differentiation (AD) to the moment integrator. Unfortunately, since Mitsuba3 uses *detached* sampling [Zeltner et al. 2021] (i.e, does not differentiate probability densities), this leads to an estimator

$$\frac{\partial_\theta f^2(\bar{\boldsymbol{x}})}{q^2(\bar{\boldsymbol{x}})}, \tag{24}$$

which mismatches our unbiased variant based on Eq. (17):

$$\frac{\partial_\theta h(\bar{\boldsymbol{x}})}{q(\bar{\boldsymbol{x}})} = \frac{1}{q(\bar{\boldsymbol{x}})} \partial_\theta \left(\frac{f^2(\bar{\boldsymbol{x}})}{q(\bar{\boldsymbol{x}})} \right). \tag{25}$$

As we will demonstrate in §6, using the estimator in Eq. (24) can lead to very high bias in derivative estimates that significantly reduces the quality of inverse rendering results.

Compared with Weier et al.'s method [2021], our technique not only enjoys unbiased derivative estimates but also offers the generality to differentiate with respect to object geometries and sampling probabilities.

5 Variance-Aware Inverse Rendering

Leveraging recent advances in differentiable rendering [Zhang et al. 2020; Bangaru et al. 2020], most existing physics-based inverse-rendering pipelines [Luan et al. 2021; Sun et al. 2023; Yan et al. 2023] seek for scene parameters θ that minimize some **rendering bias** $\mathcal{L}_{\text{bias}}(I, I_0)$ capturing the difference between the converged rendering $I = \mathbb{E}[\langle I \rangle]$ of the scene and some predetermined target I_0.

With the variance derivative $\partial_\theta \mathbb{V}[\langle I \rangle]$ derived in §4, our technique enables the differentiation of *variance-aware* losses that, in turn, lead to an important application we call *variance-aware inverse rendering*. In what follows, we discuss these aspects in more details.

Variance-aware losses. A *variance-aware loss* takes the form:

$$\mathcal{L}[\langle I \rangle] \coloneqq \mathcal{L}_{\text{bias}}(I,\ I_0) + \lambda\, \mathbb{V}[\langle I \rangle], \tag{26}$$

where $\lambda \in \mathbb{R}_{>0}$ is a weight for the variance component.

The commonly used *Mean Square Error* (MSE) [Pegoraro 2016, §4.4.9] is essentially a specific form of Eq. (26) with the rendering bias $\mathcal{L}_{\text{bias}}$ set to squared L_2 and $\lambda = 1$:

$$MSE[\langle I \rangle] = \mathbb{E}[(\langle I \rangle - I_0)^2] \coloneqq (I - I_0)^2 + \mathbb{V}[\langle I \rangle]. \tag{27}$$

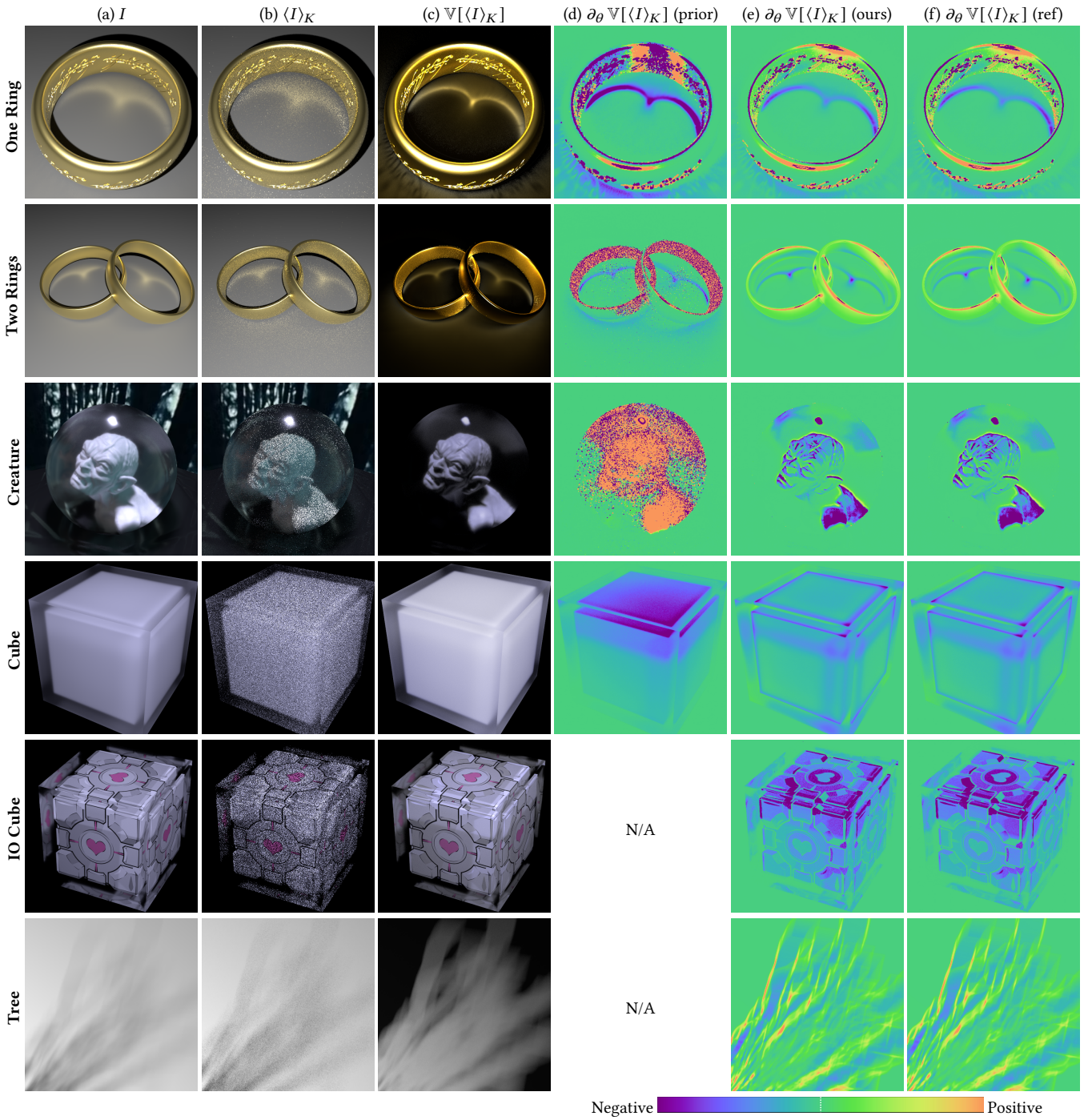


Fig. 3. **Validation:** We compare derivative estimates of rendering variances. (a) Converged path-tracing results. (b) Path-tracing results using $K = 64$ samples per pixel. (c) Variance of the estimator used in (b). For better visualization, we apply a `sigmoid` remapping to all pixel values. (d, e) Variance derivatives estimated (in equal sample and time) using the baseline [Weier et al. 2021] and our methods, respectively. (f) finite-difference references (computed in significantly more time).

In this case, the rendering bias $(I - I_0)^2$ can be estimated in an unbiased fashion using the dual-buffer method [Deng et al. 2022]:

$$\langle (I - I_0)^2 \rangle = (\langle I_1 \rangle - I_0)(\langle I_2 \rangle - I_0), \tag{28}$$

where $\langle I_1 \rangle$ and $\langle I_2 \rangle$ are *independent* Monte Carlo renderings.

Modeling rendering time. When the parameter θ strongly affects the computational speed of the estimator $\langle I \rangle$, we further generalize Eq. (26) to

$$\mathcal{L}^*[\langle I \rangle] := \mathcal{L}_{\text{bias}}(I,\ I_0) + \lambda^*\, t[\langle I \rangle]\, \mathbb{V}[\langle I \rangle], \tag{29}$$

where $t[\langle I \rangle]$ predicts the execution time of the estimator $\langle I \rangle$.

We note that, when $t[\langle I \rangle]$ is a constant, Eqs. (26) and (29) are equivalent with $\lambda = \lambda^*\, t[\langle I \rangle]$.

Modeling multi-sample estimators. Our derivations in §4 focus on the estimator $\langle I \rangle$ expressed in Eq. (3) that uses one path sample $\bar{\boldsymbol{x}}$. On the other hand, most practical Monte Carlo rendering settings use K independent and identically distributed (i.i.d.) samples $\{\bar{\boldsymbol{x}}_k : k = 1, 2, \ldots, K\}$:

$$\langle I \rangle_K := \frac{1}{K} \sum_{k=1}^{K} \frac{f(\bar{\boldsymbol{x}}_k)}{q(\bar{\boldsymbol{p}}_k)}. \tag{30}$$

To estimate the variance and its derivative of the K-sample estimator $\langle I \rangle_K$, the number of sample paths needed by a naïve method scales with K—which can lead to slow performance when K is large. To address this problem, we leverage the relations

$$\mathbb{E}[\langle I \rangle_K] = \mathbb{E}[\langle I \rangle] = I, \qquad \mathbb{V}[\langle I \rangle_K] = \frac{1}{K}\, \mathbb{V}[\langle I \rangle], \tag{31}$$

allowing the variance-aware losses in Eqs. (26) and (29) to be evaluated using only the single-sample estimator $\langle I \rangle$:

$$\mathcal{L}[\langle I \rangle_K] = \mathcal{L}_{\text{bias}}(\mathbb{E}[\langle I \rangle],\ I_0) + \frac{\lambda}{K}\, \mathbb{V}[\langle I \rangle], \tag{32}$$

$$\mathcal{L}^*[\langle I \rangle_K] = \mathcal{L}_{\text{bias}}(\mathbb{E}[\langle I \rangle],\ I_0) + \lambda^*\, t[\langle I \rangle]\, \mathbb{V}[\langle I \rangle], \tag{33}$$

where Eq. (33) further assumes $t[\langle I \rangle_K]$ to be proportional to $t[\langle I \rangle]$ with respect to the parameter θ.

It is worth noting that, the number of samples (per pixel) used for estimating Eqs. (32) and (33) and their derivatives—which we term as the **training spp**—does not have to equal K. We set the training spp to no more than 32 for all results in this paper.

Variance-aware inverse rendering. Leveraging our technique expressed in §4, we can now differentiate the variance-aware losses in Eqs. (32) and (33):

$$\partial_\theta \mathcal{L}[\langle I \rangle_K] = \frac{\partial \mathcal{L}_{\text{bias}}}{\partial I}(\partial_\theta I) + \frac{\lambda}{K}\, \partial_\theta\, \mathbb{V}[\langle I \rangle], \tag{34}$$

$$\partial_\theta \mathcal{L}^*[\langle I \rangle_K] = \frac{\partial \mathcal{L}_{\text{bias}}}{\partial I}(\partial_\theta I) + \lambda^*\, \partial_\theta\, (t[\langle I \rangle]\, \mathbb{V}[\langle I \rangle]), \tag{35}$$

where $\partial_\theta I$ follows in Eq. (12) and can be estimated using previous differentiable rendering methods [Zhang et al. 2020, 2023].

This leads to an important application which we call *variance-aware inverse rendering*: Given a Monte Carlo estimator $\langle I \rangle_K$ and a virtual scene (that determines the path space $\boldsymbol{\Omega}$), the goal is to slightly alter the estimator and/or the scene (by tuning the value of some parameter θ) so that some predetermined variance-aware loss is minimized. The resulting altered estimator and/or scene offer a good balance between rendering bias (with respect to the unaltered configuration) and rendering variance.

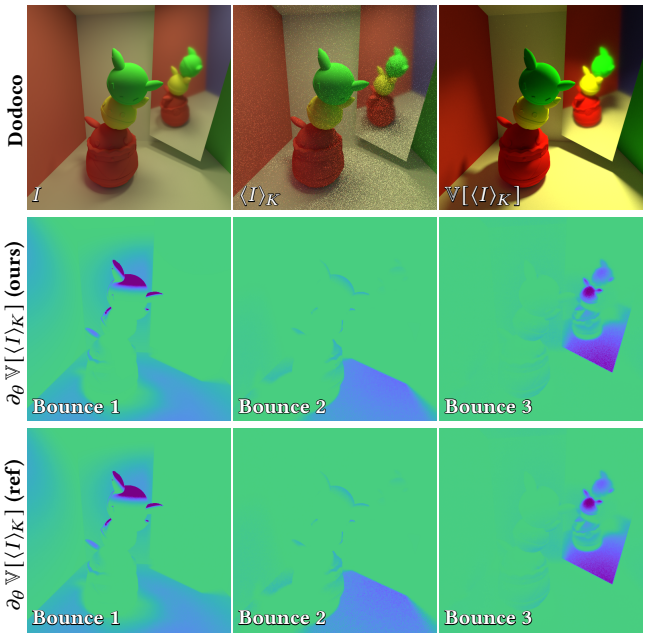


Fig. 4. **Validation:** We validate our estimates of the derivatives $\partial_\theta \mathbb{V}[\langle I \rangle_K]$ with respect to the Russian-roulette survival probability at first three bounces. The references are obtained using finite differences.

6 Results

We implement our Monte Carlo estimators described in §4 and §5 on the GPU using the Dr.Jit numerical backend [Jakob et al. 2022]. In the following, we show differentiable-rendering and inverse-rendering results in §6.1 and §6.2, respectively.

6.1 Validations and Evaluations

Validations. To validate our technique, we compare in Figure 3 as well as in the supplement our estimates of the derivatives $\partial_\theta \mathbb{V}[\langle I \rangle_K]$—where $\langle I \rangle_K$ is a unidirectional path tracer using next-event estimation (NEE) and $K = 64$ sample paths per pixel—to references computed using finite differences (FD).

The One Ring and Two Rings examples contain glossy rings on a diffuse surface lit by an area light, and derivatives are estimated with respect to the surface roughness of the rings. Creature and Cube involve two diffuse objects encapsulated inside glass sphere and cube, respectively, under area lighting. We differentiate with respect to the surface roughness of the glass containers. For all three examples, our technique produces derivative estimates closely matching the references. The method used by Weier et al. [2021], as discussed at the end of §4, suffers from very high bias in derivative estimates.

The remaining examples in Figure 3 use differentiation beyond surface roughness—which is not supported by Weier et al.'s [2021]

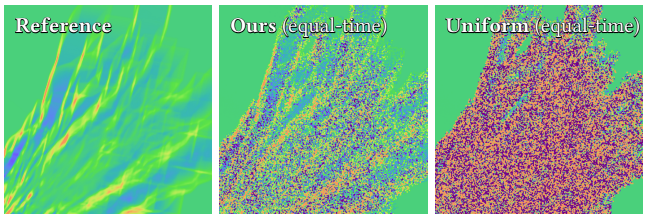


Fig. 5. **Ablation:** We compare variance derivatives for the Tree scene estimated using different sampling methods. In this case, the derivatives are emerge solely from the *boundary* component of Eq. (14).

method. Our derivative estimates closely match the finite-difference references for all examples.

IO Cube uses a similar configuration as the Cube scene except for using a textured object. Tree contains a tree-like object lit by a small area light, casting soft shadows on the ground. The derivatives of this example emerge solely from the *boundary* integrals in Eqs. (12) and (14). For both examples, we differentiate with respect to a parameter that controls the geometric scaling and emission of the light simultaneously so that its total power remains constant.

Lastly, the Dodoco scene in Figure 4 contains three diffuse objects and a mirror inside a Cornell box. We differentiate the rendering variance with respect to the Russian-roulette survival possibilities. Specifically, we use a simplified scenario where these probabilities are set solely based on the number of bounces: $q^{\text{survival}}(\boldsymbol{x}_n) := q^{\text{survival}}_{N-n}$ for any $\boldsymbol{x}_n$. In other words, we use $q^{\text{survival}}_k \in [0, 1)$ to control the survival probability at the k-th bounce from the camera for $k = 1, 2, \ldots$. Based on this setting, we estimate derivatives of rendering variance with respect to q^{survival}_1, q^{survival}_2, and q^{survival}_3, respectively. Our results match the finite-difference references closely.

Estimating boundary integrals. We demonstrate in Figure 5 the effectiveness of importance sampling techniques for estimating the *boundary* integral in Eq. (14). At equal time, our technique built upon the previous guiding method by Zhang et al. [2020] significantly outperforms simple uniform sampling. We note that the effectiveness can be further improved by adopting more recent techniques [Yan et al. 2022; Zhang et al. 2023].

6.2 Variance-Aware Inverse Rendering Results

We now show variance-aware inverse rendering results. Our optimization time ranges from 20 seconds to 10 minutes on an RTX 4090 GPU. The performance is mainly affected by the number of differentiated parameters, the image resolution, and the depth of the light path. Differentiating the geometry can take more time due to the need of estimating additional *boundary* integrals.

Ablation. To demonstrate the effect of different K values, we show variance-aware inverse rendering results of the Two Ring scene in Figure 6. The optimizations use the variance-aware loss defined in Eq. (32) and the training spp set to 16. With lower K values, the loss is dominated by the variance term, resulting in more drastic modifications of the scene (and thus higher rendering bias). With higher K values, in contrast, the loss is dominated by the bias term, leading to smaller changes.

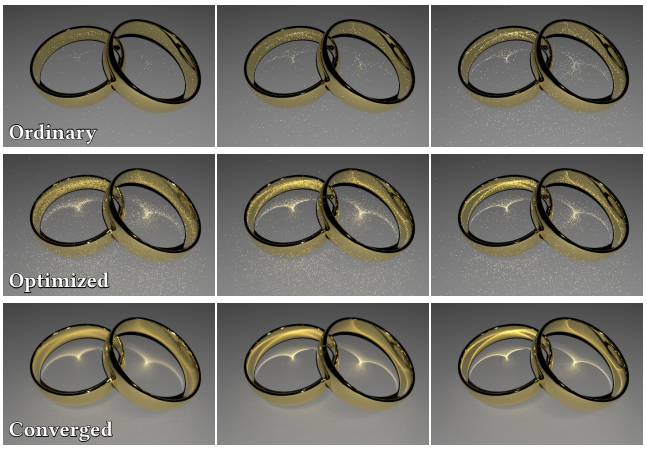


Fig. 6. **Ablation:** Variance-aware inverse rendering results of the Two Rings scene obtained by altering surface roughness of the rings so that the variance-aware loss (32) with different sample counts (i.e., K values) is minimized. Images in the first two rows are rendered with the corresponding sample counts; Those in the last row are converged renderings of the optimized scenes.

In addition, we demonstrate in Figure 7 the importance of unbiased derivative estimates by comparing variance-aware inverse rendering results. Both examples use identical settings as Figure 3 except for using lower (ordinary) object roughness and $K = 512$. The inverse rendering optimizations use the loss defined in Eq. (32) with $\mathcal{L}_{\text{bias}}$ being the L^2 function and $\lambda = 1$. In the Two Rings example, the optimization alters the spatially varying roughness of the rings with the training spp set to 16. The baseline method [Weier et al. 2021] suffers from over roughening and, thus, a higher variance-aware loss. For Creature—where the optimization alters a single roughness for the glass sphere and uses a training spp of 32—the baseline method gets stuck in the initial state. In contrast, our method allows the optimizations to converge nicely for both examples, and the resulting altered scenes offer lower loss than the baseline—as shown in column (e) of the figure.

Additional results. In Figure 8, we show four more inverse rendering results using the same variance-aware loss as Figure 7.

The Creature uses the same setting as Figure 7 except that the optimization adjusts a parameter controlling the size of the area light and its brightness (so that the total power of the light remains constant). The optimized scene, when rendered using the same path tracer with $K = 512$, offers better overall quality compared with the Creature result in Figure 7. This demonstrates the practical usefulness of going beyond surface roughening.

The Tree example uses a similar setting as in Figure 3. The optimization searches for the optimal light size and brightness for a path tracer using one sample per pixel. With only $K = 1$ sample, the loss is dominated by variance, forcing the optimizing to shrink the area light significantly.

The Spotlight example shows a textured diffuse object lit by a bright spotlight and a dim ambient light. Initialized with a constant

0.5, we optimize the sampling probability of the spotlight individually for each pixel. For this example, we visualize the optimized sampling probabilities in column (e).

Russian roulette. Lastly, in Figure 9, we use the same Dodoco scene as Figure 4 and show a proof-of-concept example where Russian-roulette survival probabilities are optimized. To drive the optimization, we use the loss expressed in Eq. (33) with the execution-time-predicting function set as

$$t[\langle I \rangle] := t_0 + \sum_{k>0} t_k \prod_{k'=1}^{k} q_{k'}^{\text{survival}}, \tag{36}$$

where $t_k \in \mathbb{R}_{>0}$ is a pre-computed constant measuring the fraction of render time needed by all paths with exactly k bounces (without Russian roulette).

At equal time, rendering using our optimized survival probabilities produces notably cleaner result.

7 Discussion and Conclusion

Limitations and future work. Our derivations in §4.1 rely on the assumption that the probability density q shares the same visibility-driven discontinuity points as the measurement contribution f. Although this is a mild assumption, it may need to be relaxed to handle some complex sampling procedures like delta tracking [Woodcock et al. 1965]. In addition, although our theory is not restricted to any specific sampling method, our analysis in §4.2 and implementation focus on unidirectional path tracing. Thus, applying our theory to more general cases such as bidirectional path tracing (BDPT) could be an interesting future topic.

Conclusion. In this paper, we derived a path-integral formulation for derivatives of rendering variance with respect to not only scene parameters but also sampling probabilities. Based on this formulation, we introduced unbiased Monte Carlo estimators for the derivatives. In addition, we discussed an important application of our technique—variance-aware inverse rendering—which concerns with altering a virtual scene and/or an estimator to offer a good balance between bias and variance. We validated our technique by comparing derivative estimates to finite-difference references and demonstrated the effectiveness of our method using several synthetic variance-aware inverse rendering examples.

Acknowledgments

We would like to thank Qianhui Wu for her artistic support in this work and Weizhen Huang discussions related to the rendering process. This work started when Kai Yan was an intern at Wētā FX.

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.

Xi Deng, Fujun Luan, Bruce Walter, Kavita Bala, and Steve Marschner. 2022. Reconstructing Translucent Objects Using Differentiable Rendering. In *ACM SIGGRAPH 2022 Conference Proceedings (SIGGRAPH '22)*. Article 38, 10 pages.

Frédo Durand. 2011. A Frequency Analysis of Monte-Carlo and other Numerical Integration Schemes. *MIT CSAIL Technical Report* (12 2011), TR–2011–052.

Pascal Grittmann, Iliyan Georgiev, Philipp Slusallek, and Jaroslav Křivánek. 2019. Variance-aware multiple importance sampling. *ACM Trans. Graph.* 38, 6 (2019), 152:1–152:9.

Wenzel Jakob, Sébastien Speierer, Nicolas Roussel, and Delio Vicini. 2022. Dr.Jit: A Just-In-Time Compiler for Differentiable Rendering. *Transactions on Graphics (Proceedings of SIGGRAPH)* 41, 4 (July 2022). https://doi.org/10.1145/3528223.3530099

Johannes Jendersie and Thorsten Grosch. 2019. Microfacet Model Regularization for Robust Light Transport. *Computer Graphics Forum (Proc. of EGSR)* 38, 4 (July 2019), 39–47. https://doi.org/10.1111/cgf.13768

Anton S. Kaplanyan and Carsten Dachsbacher. 2013. Path Space Regularization for Holistic and Robust Light Transport. *Computer Graphics Forum (Proc. of Eurographics 2013)* 32, 2 (2013), 63–72.

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 Trans. Graph.* 38, 6 (2019), 228:1–228:14.

Fujun Luan, Shuang Zhao, Kavita Bala, and Zhao Dong. 2021. Unified Shape and SVBRDF Recovery using Differentiable Monte Carlo Rendering. *Computer Graphics Forum* 40, 4 (2021), 101–113.

Vincent Pegoraro. 2016. *Handbook of Digital Image Synthesis: Scientific Foundations of Rendering.* A K Peters/CRC Press.

Adrien Pilleboue, Gurprit Singh, David Coeurjolly, Michael Kazhdan, and Victor Ostromoukhov. 2015. Variance Analysis for Monte Carlo Integration. *ACM Transactions on Graphics* 34 (08 2015). https://doi.org/10.1145/2766930

Alexander Rath, Pascal Grittmann, Sebastian Herholz, Petr Vévoda, Philipp Slusallek, and Jaroslav Křivánek. 2020. Variance-aware path guiding. *ACM Trans. Graph.* 39, 4 (2020), 151:1–151:12.

Alexander Rath, Pascal Grittmann, Sebastian Herholz, Philippe Weier, and Philipp Slusallek. 2022. EARS: Efficiency-Aware Russian Roulette and Splitting. *ACM Transactions on Graphics (Proceedings of SIGGRAPH 2022)* 41, 4, Article 81 (jul 2022), 14 pages. https://doi.org/10.1145/3528223.3530168

Kartic Subr and Jan Kautz. 2013. Fourier analysis of stochastic sampling strategies for assessing bias and variance in integration. *ACM transactions on graphics* 32, 4 (7 2013), 1–12. https://doi.org/10.1145/2461912.2462013

Cheng Sun, Guangyan Cai, Zhengqin Li, Kai Yan, Cheng Zhang, Carl Marshall, Jia-Bin Huang, Shuang Zhao, and Zhao Dong. 2023. Neural-PBIR reconstruction of shape, material, and illumination. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 18046–18056.

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

Delio Vicini, Sébastien Speierer, and Wenzel Jakob. 2021. Path Replay Backpropagation: Differentiating Light Paths Using Constant Memory and Linear Time. *ACM Trans. Graph.* 40, 4, Article 108 (2021), 108:1–108:14 pages.

Delio Vicini, Sébastien Speierer, and Wenzel Jakob. 2022. Differentiable Signed Distance Function Rendering. *ACM Trans. Graph.* 41, 4 (2022), 125:1–125:18.

Jiří Vorba and Jaroslav Křivánek. 2016. Adjoint-driven Russian Roulette and Splitting in Light Transport Simulation. *ACM Trans. Graph.* 35, 4 (2016), 42:1–42:11.

Philippe Weier. 2021. Optimised Path Space Regularisation. https://github.com/WeiPhil/OptimisedPathSpaceRegularisation.

Philippe Weier, Marc Droske, Johannes Hanika, Andrea Weidlich, and Jirí Vorba. 2021. Optimised Path Space Regularisation. *Computer Graphics Forum* (2021).

E Woodcock, T Murphy, P Hemmings, and S Longworth. 1965. Techniques used in the GEM code for Monte Carlo neutronics calculations in reactors and other systems of complex geometry. In *Proc. Conf. Applications of Computing Methods to Reactor Problems*, Vol. 557.

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.

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.

Kai Yan, Fujun Luan, Miloš Hašan, Thibault Groueix, Valentin Deschaintre, and Shuang Zhao. 2023. PSDR-Room: Single Photo to Scene using Differentiable Rendering. In *ACM SIGGRAPH ASIA 2023 Conference Proceedings (SA '23)*. Article 28, 11 pages. https://doi.org/10.1145/3610548.3618165

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.

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.

Shuang Zhao, Wenzel Jakob, and Tzu-Mao Li. 2020. Physics-Based Differentiable Rendering: A Comprehensive Introduction. In *ACM SIGGRAPH 2020 Courses*. Article 14, 14:1–14:30 pages.

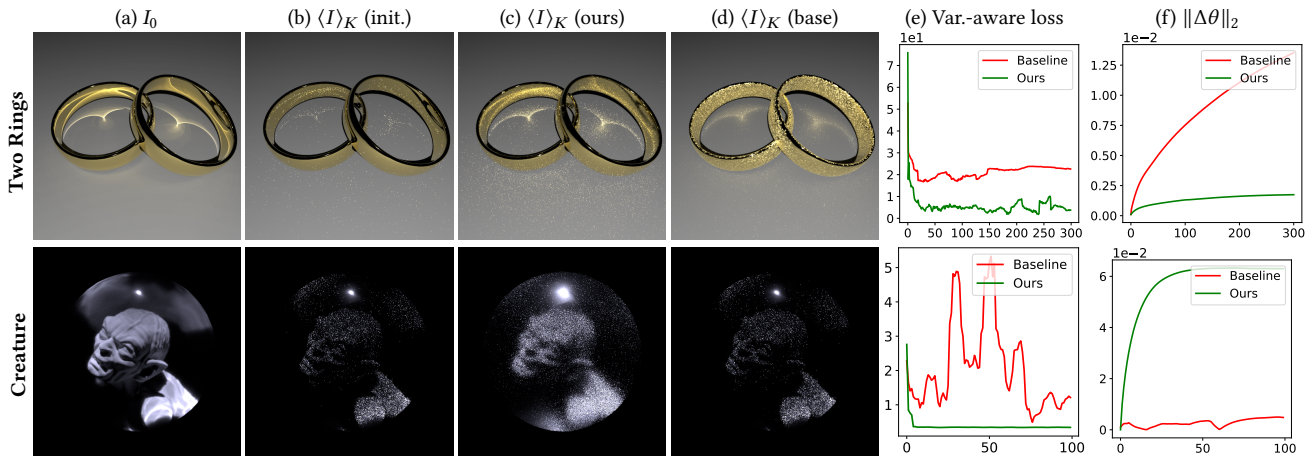


Fig. 7. **Ablation:** We demonstrate the importance of unbiased gradient estimates by comparing variance-aware inverse rendering results using gradients generated with our technique and the baseline approach [Weier et al. 2021].

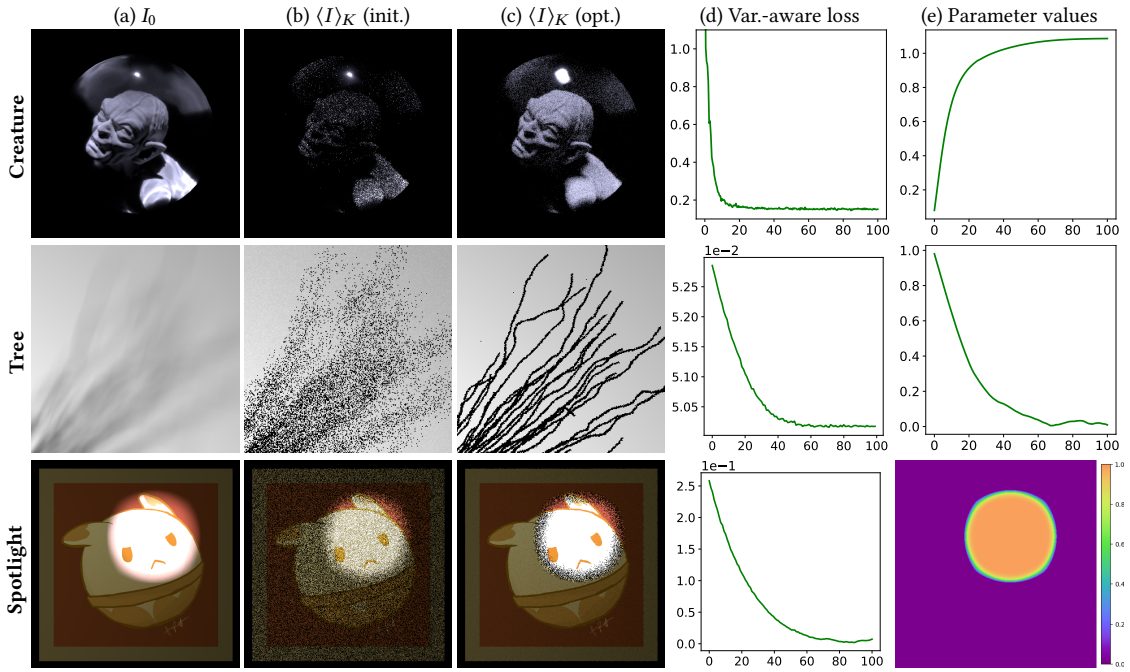


Fig. 8. **Variance-aware inverse rendering results** optimizing area-light sizes (Creature, Tree) and per-pixel sampling probability (Spotlight).

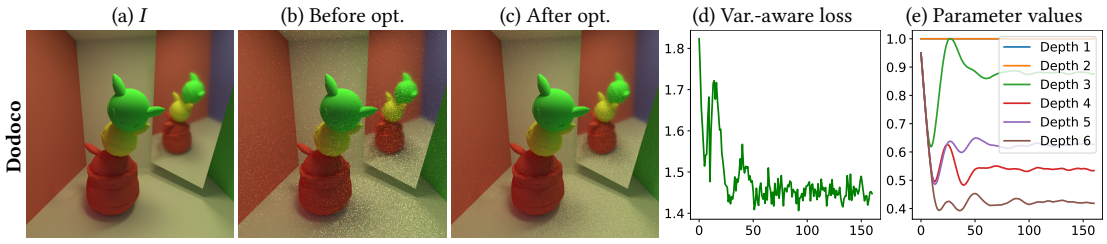


Fig. 9. **Variance-aware inverse rendering result** optimizing Russian-roulette survival probabilities.