

Path-Space Differentiable Rendering of Participating Media

CHENG ZHANG*, ZIHAN YU*, and SHUANG ZHAO, University of California, Irvine, USA

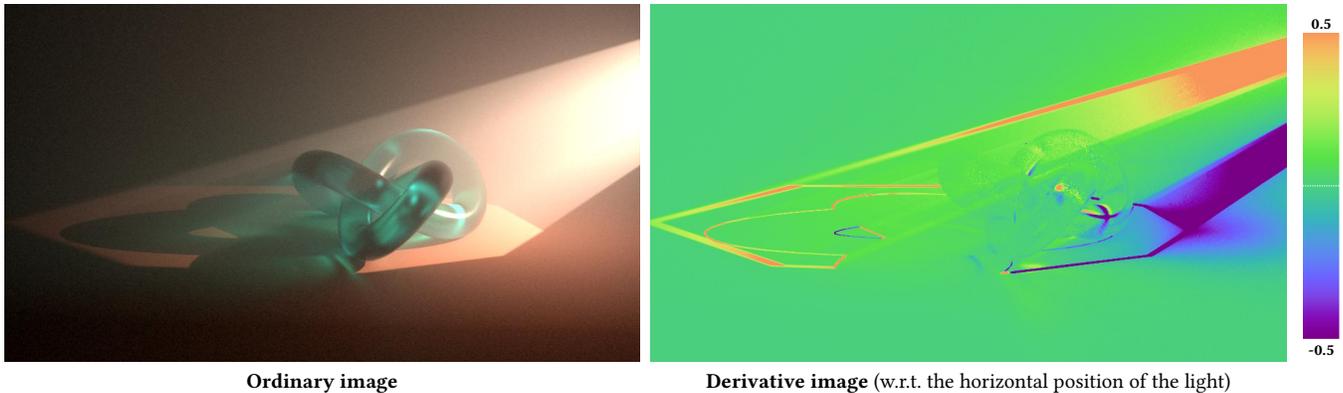


Fig. 1. In this paper, we introduce the theoretical framework of **generalized differential path integrals**. Our technique enjoys the generality of differentiating both surface and volumetric light transport with respect to arbitrary scene parameters (such as material optical properties and object geometries). Further, it offers the flexibility of designing new unbiased Monte Carlo methods capable of efficiently estimating derivatives in virtual scenes with complex geometry and light transport effects. This example shows a transparent knot inside a homogeneous participating media lit by a spotlight. On the right, we show the corresponding derivative image with respect to the horizontal location of the light.

Physics-based differentiable rendering—which focuses on estimating derivatives of radiometric detector responses with respect to arbitrary scene parameters—has a diverse array of applications from solving analysis-by-synthesis problems to training machine-learning pipelines incorporating forward-rendering processes. Unfortunately, existing general-purpose differentiable rendering techniques lack either the generality to handle volumetric light transport or the flexibility to devise Monte Carlo estimators capable of handling complex geometries and light transport effects.

In this paper, we bridge this gap by showing how generalized path integrals can be differentiated with respect to arbitrary scene parameters. Specifically, we establish the mathematical formulation of generalized differential path integrals that capture both interfacial and volumetric light transport. Our formulation allows the development of advanced differentiable rendering algorithms capable of efficiently handling challenging geometric discontinuities and light transport phenomena such as volumetric caustics.

We validate our method by comparing our derivative estimates to those generated using the finite differences. Further, to demonstrate the effectiveness of our technique, we compare both differentiable rendering and inverse rendering performance with state-of-the-art methods.

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

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

*Both authors contributed equally to the paper.

Authors' address: Cheng Zhang, chengz20@uci.edu; Zihan Yu, zihay19@uci.edu; Shuang Zhao, shz@ics.uci.edu, University of California, Irvine, USA.

Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for third-party components of this work must be honored. For all other uses, contact the owner/author(s).

© 2021 Copyright held by the owner/author(s).
0730-0301/2021/8-ART76

<https://doi.org/10.1145/3450626.3459782>

ACM Reference Format:

Cheng Zhang, Zihan Yu, and Shuang Zhao. 2021. Path-Space Differentiable Rendering of Participating Media. *ACM Trans. Graph.* 40, 4, Article 76 (August 2021), 15 pages. <https://doi.org/10.1145/3450626.3459782>

1 INTRODUCTION

Physics-based *forward rendering*—a core research topic in computer graphics—focuses on numerically estimating radiometric sensor responses in fully specified virtual scenes. Previous research efforts have led to mature algorithms capable of efficiently and accurately simulating light transport in virtual environments with high complexities.

Differentiable rendering, on the other hand, concerns with estimating derivatives of radiometric measurements with respect to differential changes of a scene. These techniques have a wide range of applications by facilitating, for instance, the use of gradient-based optimization for solving *inverse-rendering* problems, and the integration of physics-based rendering in *machine-learning* and *probabilistic-inference* pipelines.

Despite great progresses made by recent works, unique theoretical and practical challenges remain for differentiable rendering. One of them is lacking support for participating media and translucent materials, which are ubiquitous in the real world and crucial to many applications such as computational fabrication, remote sensing, and biomedical imaging. Most existing general-purpose differentiable rendering techniques [Li et al. 2018; Loubet et al. 2019; Zhang et al. 2020; Bangaru et al. 2020]—which offer the generality to differentiate with respect to scene geometry—consider only interfacial reflection and refraction of light.

Another challenge is the lack of robust Monte Carlo estimation techniques. Unlike the forward-rendering case where many sophisticated algorithms such as bidirectional path tracing and Markov-Chain Monte Carlo (MCMC) methods have been developed (e.g., [Veach and Guibas 1997; Pauly et al. 2000; Jakob and Marschner 2012a]), most recent differentiable rendering techniques [Li et al. 2018; Zhang et al. 2019; Loubet et al. 2019; Bangaru et al. 2020] rely on unidirectional path tracing and have difficulties handling complex light transport effects such as caustics.

To overcome the first challenge, the *differential radiative transfer* framework [Zhang et al. 2019] has been introduced, which offers the same level of generality as the radiative transfer theory [Chandrasekhar 1960] for forward rendering by differentiating the radiative transfer equation (RTE).

To address the second problem, the *differential path integral* formulation [Zhang et al. 2020] has been derived. Similar to Veach’s path-integral formulation [1997] for forward rendering, this technique has enabled the development of differentiable rendering algorithms beyond unidirectional path tracing.

Unfortunately, each methods have their limitations. The *differential radiative transfer* method [Zhang et al. 2019] is limited to unidirectional volumetric path tracing. Further, it requires detecting object silhouettes at each light-scattering event, which can become extremely expensive for scenes with complex geometries. The *differential path integral* formulation [Zhang et al. 2020], on the other hand, neglects volumetric scattering of light.

In this paper, we bridge the gap by introducing the mathematical formulation of *generalized differential path integral* that offers the same level of generality as differential radiative transfer [Zhang et al. 2019] and the same flexibility as differential path integral [Zhang et al. 2020]. In other words, our techniques allows differentiating volumetric light transport with respect to arbitrary scene parameters. Meanwhile, it enables the development of Monte Carlo methods that (i) go beyond unidirectional path tracing, and (ii) avoid expensive silhouette detections.

Similar to most prior works in physics-based differentiable rendering [Li et al. 2018; Loubet et al. 2019; Zhang et al. 2019, 2020; Bangaru et al. 2020], we focus on scene geometries depicted using polygonal meshes and assume the absence of perfectly specular interfaces.

Concretely, our contributions include:

- Establishing the formulation of *generalized differential path integrals* (§4) by differentiating *generalized path integrals* [Pauly et al. 2000].
- Discussing how zero-measure detectors (e.g., pinhole cameras) and sources (e.g., point lights), which are largely neglected by previous differentiable rendering techniques, can be handled in our framework (§4.3).
- Developing new unbiased and consistent Monte Carlo methods that estimate our *generalized differential path integrals* in an unbiased fashion (§5). Our estimators greatly outperform the volumetric path tracing method developed by Zhang et al. [2019] for complex scene geometries and light transport effects.

To validate our theory and algorithms, we compare our derivative estimates with those produced using finite differences (Figure 10). To demonstrate the effectiveness of our method, we compare (i) derivative images generated with our technique and differential radiative transfer (Figure 10); and (ii) inverse-rendering performance using gradients estimated with these methods (Figure 11).

2 RELATED WORK

Forward volume rendering. Monte Carlo methods have been the “gold standard” for accurately simulating photon and neutron transport in complex environments [Spanier and Gelbard 1969]. In computer graphics, volumetric path tracing and its variations (e.g., [Kajiya and Von Herzen 1984; Cerezo et al. 2005]) produce unbiased and consistent estimates of radiometric measures. Later, based on the path-integral formulation [Veach 1997] and its generalization to volumetric light transport [Pauly et al. 2000], bidirectional path tracing [Lafortune and Willems 1996] and Markov-Chain Monte Carlo (MCMC) methods (e.g., [Pauly et al. 2000; Kelemen et al. 2002; Jakob and Marschner 2012b]) have been introduced to enable efficient simulation of challenging effects such as caustics.

For a comprehensive survey on Monte Carlo volume rendering techniques, we refer to the survey by Novák et al. [2018].

Differentiable surface-only rendering. A main challenge towards developing general-purpose differentiable rendering engines has been the differentiation with respect to scene geometry, which generally requires calculating additional boundary integrals. To address this problem, Li et al. [2018] introduced a Monte Carlo edge-sampling method that provides unbiased estimates of these boundary integrals but requires detection of object silhouettes, which can be computationally expensive for complex scenes. Later, reparameterization-based methods [Loubet et al. 2019; Bangaru et al. 2020] have been introduced to avoid computing boundary integrals altogether. Despite their ability to differentiate with respect to arbitrary scene parameterizations, all these methods are obtained by differentiating the rendering equation [Kajiya 1986] and rely on unidirectional path tracing for derivative estimations, which can be inefficient when handling complex light transport effects.

By differentiating Veach’s path integrals [1997], Zhang et al. [2020] derived the formulation of differential path integrals, enabling Monte Carlo differentiable rendering beyond unidirectional path tracing. Despite its flexibility, this formulation still neglects all volumetric light transport effects. Our theory subsumes this work by showing how to differentiate full generalized path integrals.

Differentiable volume rendering. Specialized differentiable volume rendering has been used to solve analysis-by-synthesis problems in volumetric scattering [Gkioulekas et al. 2013], prefiltering of high-resolution volumes [Zhao et al. 2016], and fabrication of translucent materials [Sumin et al. 2019]. All these methods compute radiance derivatives with respect to specific material properties like optical density.

For general-purpose differentiable volume rendering, Che et al. [2020] developed a system capable of computing derivatives with respect to optical material and local normal properties. Nimier-David et al. [2019] introduced the Mitsuba 2 system that enables

differentiable volume rendering with millions of parameters. Unfortunately, these methods cannot differentiate with respect to scene geometry.

By differentiating both the rendering equation [Kajiya 1986] and the radiative transfer equation [Chandrasekhar 1960], the theory of differential radiative transfer introduced recently by Zhang et al. [2019] offers the most general differentiable rendering theory to date. However, this technique still relies on unidirectional path tracing with Monte Carlo edge sampling [Li et al. 2018] and cannot efficiently handle complex geometry or light transport effects.

Our technique overcomes these limitations by leveraging more advanced Monte Carlo estimators while offering the same level of generality as differential radiative transfer.

3 PRELIMINARIES

We now briefly review the mathematical preliminaries on forward rendering of participating media using the path-space formulation [Veach 1997; Pauly et al. 2000].

The response $I \in \mathbb{R}$ of a radiometric detector can be expressed as a **path integral** of the form:

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

where $\bar{x} = (x_0, \dots, x_N)$ denotes a **light transport path** (with x_0 on a light source and x_N on the detector); Ω is the **path space**; f is the **measurement contribution function**; and μ is the Lebesgue measure on Ω .

Veach [1997] has shown that, for surface-only light transport (that considers only interfacial reflection and refraction of light), the path space¹ is given by $\Omega = \cup_{N=1}^{\infty} \mathcal{M}^{N+1}$, where \mathcal{M} is the union of all object surfaces, and μ is the area-product measure. This formulation has been the theoretical foundation of many sophisticated Monte Carlo rendering methods such as bidirectional path tracing [Veach and Guibas 1995] and a few Markov-Chain Monte Carlo rendering methods [Veach 1997; Jakob and Marschner 2012a].

Veach's formulation has been extended by Pauly et al. [2000] to handle volumetric light transport based on the radiative transfer theory [Chandrasekhar 1960]. In what follows, we briefly review this generalized formulation.

Path space and measure. Let $\mathcal{V} \subset \mathbb{R}^3$ be a 3D volume that encapsulates the virtual scene, $\mathcal{M} \subset \mathcal{V}$ be the union of all object surfaces in the scene, and $\mathcal{V}_0 := \mathcal{V} \setminus \mathcal{M}$. A light transport path $\bar{x} = (x_0, x_1, \dots, x_N)$ with $(N+1)$ vertices and N segments is classified with its **path characteristic** l , an $(N+1)$ -bit integer that encodes the type of individual vertices. Specifically, the n -th bit of the binary representation of l , which we denote as $b_n(l)$, equals one if x_n is a **surface vertex** (i.e., $x_n \in \mathcal{M}$) and zero if it is a **volume vertex** (i.e., $x_n \in \mathcal{V}_0$). For all $N \geq 1$ and $0 \leq l < 2^{N+1}$, the set of all paths with N segments and characteristic l is

$$\Omega_N^l := \left\{ (x_0, \dots, x_N) : x_n \in \begin{cases} \mathcal{M}, & b_n(l) = 1 \\ \mathcal{V}_0 & b_n(l) = 0 \end{cases} \right\}, \quad (2)$$

and the Lebesgue measure μ_N^l on Ω_N^l is defined by

$$d\mu_N^l(\bar{x}) := \prod_{n=0}^N d\mu_{N,n}^l(x_n), \quad (3)$$

where

$$d\mu_{N,n}^l := \begin{cases} dA, & b_n(l) = 1 \\ dV, & b_n(l) = 0 \end{cases} \quad (4)$$

with dA and dV being the surface-area and volume measures, respectively. It follows that the path space in Eq. (1) becomes

$$\Omega := \cup_{N \geq 1} \cup_{l=0}^{2^{N+1}-1} \Omega_N^l, \quad (5)$$

associated with the measure

$$\mu(D) := \sum_{N \geq 1} \sum_{l=0}^{2^{N+1}-1} \mu_N^l(D \cap \Omega_N^l). \quad (6)$$

Measurement contribution. For a given light path $\bar{x} = (x_0, \dots, x_N)$, its measurement contribution is the product of per-vertex and per-segment contributions:

$$f(\bar{x}) := \left[\prod_{n=0}^N f_v(x_{n-1} \rightarrow x_n \rightarrow x_{n+1}) \right] \left[\prod_{n=1}^N G(x_{n-1} \leftrightarrow x_n) \right]. \quad (7)$$

In this equation, the per-vertex contribution equals

$$f_v(x_{n-1} \rightarrow x_n \rightarrow x_{n+1}) := \begin{cases} f_s(x_{n-1} \rightarrow x_n \rightarrow x_{n+1}), & 0 < n < N \text{ and } x_n \in \mathcal{M} \\ \sigma_s(x_n) f_p(x_{n-1} \rightarrow x_n \rightarrow x_{n+1}), & 0 < n < N \text{ and } x_n \in \mathcal{V}_0 \\ L_e(x_0 \rightarrow x_1), & n = 0 \\ W_e(x_{N-1} \rightarrow x_N), & n = N \end{cases} \quad (8)$$

where f_s is the **bidirectional scattering distribution function** (BSDF); f_p denotes the **single-scattering phase function**; and σ_s is the **scattering coefficient**. Further, L_e and W_e capture, respectively, the **source emission** and **detector importance** (or response). In this paper, we focus on surface sources (e.g., area lights) and detectors (e.g., virtual camera sensors), which are used almost exclusively in computer graphics and vision, yielding $x_0 \in \mathcal{M}$ and $x_N \in \mathcal{M}$.

In Eq. (7), the per-segment contribution is given by the **generalized geometric term** defined as

$$G(x \leftrightarrow y) := \mathbb{V}(x \leftrightarrow y) \tau(x \leftrightarrow y) \frac{D_x(y) D_y(x)}{\|x - y\|^2}, \quad (9)$$

where \mathbb{V} is the **mutual visibility function** and, for any $x, y \in \mathcal{V}$,

$$D_x(y) := \begin{cases} |\mathbf{n}(x) \cdot \overrightarrow{xy}|, & x \in \mathcal{M} \\ 1, & x \in \mathcal{V}_0 \end{cases} \quad (10)$$

with $\mathbf{n}(x)$ being the (unit-length) surface normal at x , and $\overrightarrow{xy} := (y-x)/\|y-x\|$. Further, $\tau(x \leftrightarrow y)$ indicates the **transmittance** between x and y that equals

$$\tau(x \leftrightarrow y) = \exp \left[- \int_{\overrightarrow{xy}} \sigma_t(x') d\ell(x') \right], \quad (11)$$

where σ_t is the **extinction coefficient**; \overrightarrow{xy} denotes the line segment connecting x and y ; and ℓ is the curve-length measure.

¹We hyperlink keywords to their definitions.

Table 1. List of symbols commonly used in this paper.

Symbol	Definition
θ	abstract scene parameter
$\mathcal{M}(\theta)$	object surfaces
$\mathcal{V}(\theta)$	volume encapsulating the scene
$\mathcal{V}_0(\theta)$	$\mathcal{V}(\theta) \setminus \mathcal{M}(\theta)$
f	measurement contribution
$\Omega(\theta)$	path space
$\partial\Omega(\theta)$	boundary path space
l	path characteristic
χ	motion
\mathcal{B}_M	reference surface
\mathcal{B}_V	reference volume
\mathcal{B}_{V_0}	$\mathcal{B}_V \setminus \mathcal{B}_M$
\hat{f}	material measurement contribution
$\hat{\Omega}$	material path space
$\partial\hat{\Omega}(\theta)$	material boundary path space

4 GENERALIZED DIFFERENTIAL PATH INTEGRAL

We now derive derivatives of the generalized path integral of Eq. (1) with respect to arbitrary scene parameters. To this end, we generalize the formulation of *differential path integral* introduced by Zhang et al. [2020] for differentiable rendering of surfaces.

Specifically, we first introduce a material-form reparameterization of Eq. (1) in §4.1. Then, we introduce the *generalized differential path integral* in §4.2 and discuss how perspective pinhole cameras and point lights can be supported using our formulation in §4.3. Lastly, we discuss the relation between our technique and some closely related prior works in §4.4. Table 1 summarizes the symbols commonly used in this paper.

4.1 Material-Form Parameterization

Assume the (optical and/or geometric) properties of a scene to be controlled by some abstract parameter $\theta \in \mathbb{R}$. In general, the volume \mathcal{V} encapsulating the scene and the union of object surfaces $\mathcal{M} \subset \mathcal{V}$ can both depend on the parameter θ . This causes the corresponding path space Ω given by Eq. (5) to also depend on θ , making the differentiation of Eq. (1) more challenging.

To address this problem, Zhang et al. propose to apply a change of variable to the ordinary path integral so that the new integral domain is independent of the scene parameter θ . They have also shown how this can be achieved for surface-only light transport (using Veach’s formulation [1997]). In what follows, we demonstrate how this idea can be realized for generalized path integrals.

Material path space. Let χ be a differentiable mapping, or a **motion**,² such that $\chi(\cdot, \theta)$ is a smooth bijection that transforms (i) some **reference surface** \mathcal{B}_M to $\mathcal{M}(\theta)$ and (ii) some **reference volume** $\mathcal{B}_V \supset \mathcal{B}_M$ to $\mathcal{V}(\theta) \supset \mathcal{M}(\theta)$. We note that both \mathcal{B}_V and \mathcal{B}_M are independent of the scene parameter θ . Additionally, we call any $\mathbf{x} \in \mathcal{V}(\theta)$ a **spatial point** and any $\mathbf{p} \in \mathcal{B}_V$ a **material point**.

²We follow the terminology used by Zhang et al. [2020] originated in continuum and fluid mechanics.

Let $\mathcal{B}_{V_0} := \mathcal{B}_V \setminus \mathcal{B}_M$. Following Eq. (2), we define the set of **material light paths** $\bar{\mathbf{p}}$ of length N (i.e. with N segments) and characteristic l as

$$\hat{\Omega}_N^l := \left\{ (\mathbf{p}_0, \mathbf{p}_1, \dots, \mathbf{p}_N) : \mathbf{p}_n \in \begin{cases} \mathcal{B}_M, & b_n(l) = 1 \\ \mathcal{B}_{V_0}, & b_n(l) = 0 \end{cases} \right\}, \quad (12)$$

associated with the measure μ_N^l defined in Eq. (3) (where the surface area and volume are with respect to the reference surface \mathcal{B}_M and volume \mathcal{B}_V , respectively). Then, similar to Eq. (5), we define the **material path space** $\hat{\Omega}$ as

$$\hat{\Omega} := \bigcup_{N \geq 1} \bigcup_{l=0}^{2^{N+1}-1} \hat{\Omega}_N^l, \quad (13)$$

with an associated measure identical to the one expressed in Eq. (6).

Choices of reference configurations. In practice, we make the reference configurations coincide with the actual scene geometry with the parameter θ fixed at some θ_0 . Precisely, we set the reference surface \mathcal{B}_M to the object surfaces \mathcal{M} at $\theta = \theta_0$ —that is, $\mathcal{B}_M = \mathcal{M}(\theta_0)$ —and the reference volume to $\mathcal{B}_V = \mathcal{V}(\theta_0)$, as illustrated in Figure 2.

Under this setting, the mapping $\chi(\cdot, \theta_0) : \mathcal{B}_V \mapsto \mathcal{V}(\theta_0)$ reduces to the identity map (that is, $\chi(\mathbf{p}, \theta_0) = \mathbf{p}$) for all \mathbf{p} , causing the material path space $\hat{\Omega}$ and the ordinary path space Ω to coincide: $\hat{\Omega} = \Omega(\theta_0)$. We note that $\chi(\cdot, \theta)$ generally does not equal the identity map for $\theta \neq \theta_0$.

We will discuss how χ can be expressed and stored in §6.1.

Change of variable. The motion χ induces a differentiable mapping $\bar{\chi}$ such that $\bar{\chi}(\cdot, \theta)$ transforms the material path space $\hat{\Omega}$ to the ordinary one $\Omega(\theta)$ for any θ . Precisely, $\bar{\chi}(\cdot, \theta)$ maps a material path $\bar{\mathbf{p}} = (\mathbf{p}_0, \dots, \mathbf{p}_N) \in \hat{\Omega}$ to an ordinary one $\bar{\mathbf{x}} \in \Omega(\theta)$:

$$\bar{\mathbf{x}} = \bar{\chi}(\bar{\mathbf{p}}, \theta) = (\mathbf{x}_0, \dots, \mathbf{x}_N), \quad \text{where } \mathbf{x}_n = \chi(\mathbf{p}_n, \theta) \text{ for all } n. \quad (14)$$

By applying to Eq. (1) this change of variable from $\bar{\mathbf{x}}$ to $\bar{\mathbf{p}}$, we obtain the **material-form generalized path integral**:

$$I = \int_{\hat{\Omega}} \hat{f}(\bar{\mathbf{p}}) d\mu(\bar{\mathbf{p}}), \quad (15)$$

where \hat{f} is the **material measurement contribution function** defined as

$$\hat{f}(\bar{\mathbf{p}}) := f(\bar{\mathbf{x}}) \left\| \frac{d\mu(\bar{\mathbf{x}})}{d\mu(\bar{\mathbf{p}})} \right\| = f(\bar{\mathbf{x}}) \prod_n J(\mathbf{p}_n), \quad (16)$$

where $\bar{\mathbf{x}}$ is the ordinary light path corresponding to $\bar{\mathbf{p}}$ given by $\bar{\mathbf{x}} = \bar{\chi}(\bar{\mathbf{p}}, \theta)$, and

$$J(\mathbf{p}) := \begin{cases} \left\| \frac{dA(\mathbf{x})}{dA(\mathbf{p})} \right\|, & \mathbf{p} \in \mathcal{B}_M \\ \left\| \frac{dV(\mathbf{x})}{dV(\mathbf{p})} \right\|, & \mathbf{p} \in \mathcal{B}_{V_0} \end{cases} \quad (17)$$

Since $\mathbf{x} = \chi(\mathbf{p}, \theta)$, J generally depends on the scene parameter θ . With the aforementioned choices of reference configurations with $\chi(\cdot, \theta_0)$ reducing to the identity map for some fixed θ_0 , $J|_{\theta=\theta_0}$ will always have unit value (with potentially nonzero derivative). We will discuss how this term can be computed in practice in §6.2.

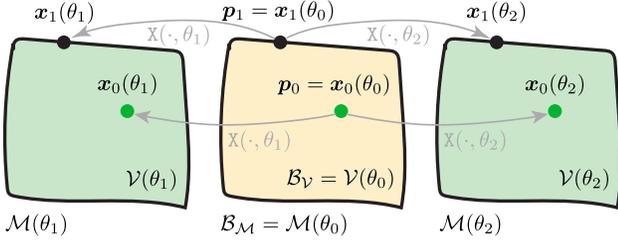


Fig. 2. **Material-form parameterization** of a block whose horizontal location is controlled by a parameter θ . In this example, the reference surface \mathcal{B}_M and volume \mathcal{B}_V are selected as the block at some fixed $\theta = \theta_0$ (illustrated in orange). Then, the motion X captures the motion of the block (hence the name) by mapping each point in the reference volume to the corresponding one in the “moving” block via $X(\cdot, \theta)$ for any θ . We show the images $\mathcal{V}(\theta)$, $\mathbf{x}_0(\theta)$ and $\mathbf{x}_1(\theta)$ of the reference volume \mathcal{B}_V , an interior point $\mathbf{p}_0 \in \mathcal{B}_{V_0}$ and of a boundary point $\mathbf{p}_1 \in \mathcal{B}_M$, respectively, at $\theta = \theta_1$ and $\theta = \theta_2$.

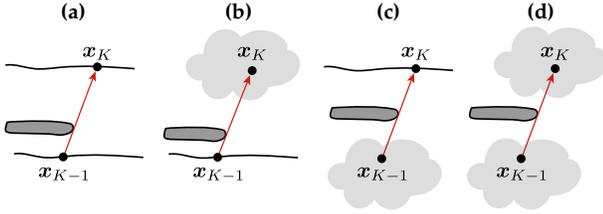


Fig. 3. A **boundary segment** $\overline{\mathbf{x}_{K-1}\mathbf{x}_K}$ has property that its interior intersects with another surface in the scene at exactly one point. This causes the endpoint \mathbf{x}_K to lie on the discontinuity boundary of the visibility function $\mathbb{V}(\mathbf{x}_{K-1} \leftrightarrow \cdot)$ with \mathbf{x}_{K-1} fixed (and vice versa). With the presence of participating media, a boundary segment can connect two surface vertices (a), two volume vertices (d), or one surface and one volume vertices (b, c).

4.2 Generalized Differential Path Integral

With the formulation of material-form generalized path integral established, differentiable rendering of participating media boils down to differentiating Eq. (15).

Continuity assumptions. To facilitate the derivation of the derivative, we make a few assumptions:

- A.1** There is no ideal specular surface (e.g., perfect mirror or smooth glass);
- A.2** The source emission L_e , sensor importance W_e , BSDFs f_s , and phase functions f_p are C^0 -continuous spatially and directionally;
- A.3** The extinction coefficient σ_s and scattering coefficient σ_t are continuous in the interior of each medium;
- A.4** Discontinuities of the Jacobian determinants J of Eq. (17), if they exist, are independent of the parameter θ .

We note that Assumption **A.2** can be relaxed and will discuss this aspect in §4.3.

Boundary segment, path, and path space. We define a **boundary light path** $\bar{\mathbf{x}} = (\mathbf{x}_0, \mathbf{x}_1, \dots, \mathbf{x}_N)$ to be a light path containing exactly one **boundary segment** $\overline{\mathbf{x}_{K-1}\mathbf{x}_K}$ (for some $0 < K \leq N$) such that the interior of this segment intersects the object surfaces $\mathcal{M}(\theta)$ at exactly one point (see Figure 3). We further denote the set of all boundary paths (with finite length) as the **boundary path space** $\partial\Omega(\theta)$.

The measure $\dot{\mu}$ associated with the boundary path space satisfies that, for any boundary path $\bar{\mathbf{x}} = (\mathbf{x}_0, \mathbf{x}_1, \dots, \mathbf{x}_N)$ with characteristic l and boundary segment $\overline{\mathbf{x}_{K-1}\mathbf{x}_K}$:

$$d\dot{\mu}(\bar{\mathbf{x}}) = \left(\prod_{n \neq K} d\mu_{N,n}^l(\mathbf{x}_n) \right) \begin{cases} dl(\mathbf{x}_K), & b_K(l) = 1 \\ dA(\mathbf{x}_K), & b_K(l) = 0 \end{cases} \quad (18)$$

where $d\mu_{N,n}^l$ is defined in Eq. (4).

Under the material-form parameterization described in §4.1, for each boundary path $\bar{\mathbf{x}} \in \partial\Omega(\theta)$ with boundary segment $\overline{\mathbf{x}_{K-1}\mathbf{x}_K}$, we call $\bar{\mathbf{p}} = \bar{X}^{-1}(\bar{\mathbf{x}}, \theta)$ a **material boundary path**, and the segment $\overline{\mathbf{p}_{K-1}\mathbf{p}_K}$ on $\bar{\mathbf{p}}$ a **material boundary segment**. Additionally, we define the **material boundary path space** $\partial\hat{\Omega}(\theta)$ as the set of all material boundary paths. We note that, unlike the material path space $\partial\Omega$ that is independent of the scene parameter θ , the material boundary path space $\partial\hat{\Omega}$ does typically depend on θ . This is because, with \mathbf{p}_{K-1} fixed, for $\overline{\mathbf{p}_{K-1}\mathbf{p}_K}$ to be a material boundary segment, \mathbf{p}_K will need to depend on θ . We demonstrate this in Figure 4.

Generalized differential path integral. Based on Assumptions **A.1–A.4**, discontinuities of the material measurement contribution \hat{f} would fully emerge from the mutual visibility function \mathbb{V} in the geometric terms G . It follows that differentiating the material-form generalized path integral of Eq. (15) produces the following (material-form) **generalized differential path integral**:

$$\frac{dI}{d\theta} = \int_{\hat{\Omega}} \frac{d\hat{f}(\bar{\mathbf{p}})}{d\theta} d\mu(\bar{\mathbf{p}}) + \int_{\partial\hat{\Omega}} \Delta\hat{f}(\bar{\mathbf{p}}) v(\mathbf{p}_K) d\dot{\mu}(\bar{\mathbf{p}}), \quad (19)$$

where the definitions of individual terms will be discussed in the following. For more details on the derivation of this result, please see Appendix A.

- In the *interior* integral, $d\hat{f}/d\theta$ indicates the **scene derivative**—a type of material derivative—of the material measurement contribution \hat{f} given by Eq. (16). This derivative is calculated based on the relation of $\bar{\mathbf{x}} = \bar{X}(\bar{\mathbf{p}}, \theta)$.
- In the *boundary* integral, $\partial\hat{\Omega}$ is the material boundary path space, and the measure $\dot{\mu}$ is defined in Eq. (18).
- For each material boundary path $\bar{\mathbf{p}}$ with material boundary segment $\overline{\mathbf{p}_{K-1}\mathbf{p}_K}$, the term $v(\mathbf{p}_K)$ is a scalar that captures how “fast” (with respect to θ) the discontinuity boundary evolves at \mathbf{p}_K along the normal direction (see Figure 4). Precisely,

$$v(\mathbf{p}_K) := \frac{d\mathbf{p}_K}{d\theta} \cdot \mathbf{n}(\mathbf{p}_K), \quad (20)$$

where $\mathbf{p}_K = X^{-1}(\mathbf{x}_K, \theta)$, and “ \cdot ” denotes the dot-product operator. Additionally, when \mathbf{p}_K is a surface vertex, $\mathbf{n}(\mathbf{p}_K)$ is a unit vector perpendicular to the discontinuity curve within the reference

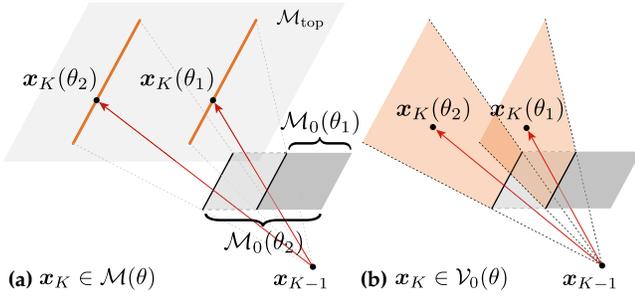


Fig. 4. **Evolution of discontinuity boundaries:** when the scene geometry varies with the parameter θ , so will the visibility boundaries. In this example, θ controls the width of the surface \mathcal{M}_0 . Then, to form a boundary segment $\overline{\mathbf{x}_{K-1} \mathbf{x}_K}$ with one endpoint \mathbf{x}_{K-1} fixed (on a surface or inside a medium), the other endpoint \mathbf{x}_K can lay (a) on a curve within an object surface (\mathcal{M}_{top} in this example); and (b) within a surface determined by \mathbf{x}_{K-1} and the left edge (in solid black lines) of \mathcal{M}_0 . The discontinuity curves and surfaces (illustrated in orange) generally depend on the parameter θ . After transforming these curves and surfaces back to the reference configuration (using the inverse of $X(\cdot, \theta)$ that maps $\mathbf{x}_K(\theta)$ to \mathbf{p}_K for all θ), $v(\mathbf{p}_K)$ in Eq. (19) captures the change rate (with respect to θ) of \mathbf{p}_K along the normal direction of the discontinuity curve or surface (under reference configurations).

surface containing \mathbf{p}_K . When \mathbf{p}_K is a volume vertex, on the other hand, $\mathbf{n}(\mathbf{p}_K)$ is a unit vector perpendicular to the corresponding discontinuity surface in the reference volume. Further, evaluating $d\mathbf{p}_K/d\theta$ in Eq. (20) requires parameterizing locally the discontinuity curve or surface near \mathbf{p}_K . We will discuss how this can be done in practice in §6.3.

- $\Delta \hat{f}(\hat{\mathbf{p}})$ denotes the difference in material measurement contribution \hat{f} across the discontinuity boundary. Based on our continuity assumptions (A.1–A.4), it holds that

$$\Delta \hat{f}(\hat{\mathbf{p}}) = \hat{f}(\hat{\mathbf{p}}) \frac{\Delta G(\mathbf{x}_{K-1} \leftrightarrow \mathbf{x}_K)}{G(\mathbf{x}_{K-1} \leftrightarrow \mathbf{x}_K)}, \quad (21)$$

where $\Delta G(\mathbf{x}_{K-1} \leftrightarrow \mathbf{x}_K)$ equals $-G(\mathbf{x}_{K-1} \leftrightarrow \mathbf{x}_K)$ if the normal (of the discontinuity boundary at \mathbf{x}_K) points toward a region visible to \mathbf{x}_{K-1} , or $G(\mathbf{x}_{K-1} \leftrightarrow \mathbf{x}_K)$ if otherwise.

4.3 Supporting Pinhole Cameras and Point Lights

In the following, we discuss how perspective pinhole cameras and point lights—which are commonly used in computer graphics and vision—can be incorporated in our material path integral framework established in §4.1 and §4.2.

With the detector being a pinhole camera located at $\mathbf{x}_{\text{cam}} \in \mathcal{V}_0$, any light transport path must terminate at \mathbf{x}_{cam} to have a nonzero measurement contribution. For a light path $(\mathbf{x}_0, \dots, \mathbf{x}_N, \mathbf{x}_{\text{cam}})$, the detector importance of the pinhole camera equals

$$W_e^{\text{pinhole}}(\mathbf{x}_N \rightarrow \mathbf{x}_{\text{cam}}) := \frac{\mathcal{P}(\mathbf{x}_N)}{(\mathbf{n}_{\text{cam}} \cdot \overrightarrow{\mathbf{x}_{\text{cam}} \mathbf{x}_N})^3}, \quad (22)$$

where \mathbf{n}_{cam} is the camera's axis of projection, $\overrightarrow{\mathbf{x}_{\text{cam}} \mathbf{x}_N}$ denotes the unit vector pointing from \mathbf{x}_{cam} toward \mathbf{x}_N , and $\mathcal{P}(\mathbf{x}_N)$ indicates the

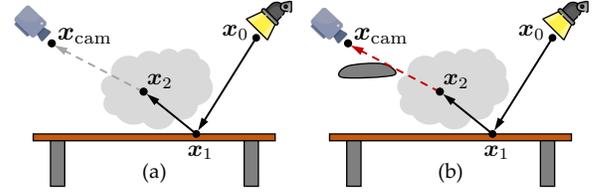


Fig. 5. **Perspective pinhole camera:** (a) To support this camera model, we encode the contribution of the segment $\overline{\mathbf{x}_N \mathbf{x}_{\text{cam}}}$ illustrated as gray dashed arrows in the detector importance function via Eq. (23). (b) We also allow $\overline{\mathbf{x}_N \mathbf{x}_{\text{cam}}}$ to be a boundary segment to capture the additional discontinuities introduced by this segment.

pixel reconstruction filter—which we assume to be C^0 -evaluated at the projection of \mathbf{x}_N on the image plane.

To avoid introducing additional Dirac delta functions in our derivations, we do not treat \mathbf{x}_{cam} as an endpoint of all light paths. Instead, we encode its contributions in the detector importance: For any path $(\mathbf{x}_0, \dots, \mathbf{x}_N)$ with \mathbf{x}_N being a standard surface or volume vertex, we set

$$W_e(\mathbf{x}_{N-1} \rightarrow \mathbf{x}_N) := f_v(\mathbf{x}_{N-1} \rightarrow \mathbf{x}_N \rightarrow \mathbf{x}_{\text{cam}}) G(\mathbf{x}_N \leftrightarrow \mathbf{x}_{\text{cam}}) W_e^{\text{pinhole}}(\mathbf{x}_N \rightarrow \mathbf{x}_{\text{cam}}), \quad (23)$$

where f_v is defined in Eq. (8), and G is the geometric term of Eq. (9). Then, as demonstrated in Figure 5-a, the measurement contribution of a light path $(\mathbf{x}_0, \dots, \mathbf{x}_N)$ with the detector importance of Eq. (23) equals that of $(\mathbf{x}_0, \dots, \mathbf{x}_N, \mathbf{x}_{\text{cam}})$ with Eq. (22).

Similarly, with a (uniform) point light located at \mathbf{x}_{src} , we encode the contributions related to \mathbf{x}_{src} in the source emission L_e by setting

$$L_e(\mathbf{x}_0 \rightarrow \mathbf{x}_1) := f_v(\mathbf{x}_{\text{src}} \rightarrow \mathbf{x}_0 \rightarrow \mathbf{x}_1) G(\mathbf{x}_{\text{src}} \leftrightarrow \mathbf{x}_0) \mathcal{I}, \quad (24)$$

where \mathcal{I} denotes the intensity of the point light.

We note that our formulations of Eqs. (23) and (24) are mostly theoretical: They allow pinhole cameras and point lights to be handled using the same derivations in §4.1 and §4.2.

Material-form formulation. In general, the position \mathbf{x}_{cam} and orientation \mathbf{n}_{cam} of a pinhole camera can be functions of the scene parameter θ and can be parameterized using the material-form formulation as follows. Let $\mathbf{p}_{\text{cam}}^{(0)}$ and $\mathbf{p}_{\text{cam}}^{(1)}$ be two fixed points in the reference volume $\mathcal{B}_{\mathcal{V}_0}$ that represent, respectively, the camera's center of projection and a point along the center axis. Then, for all θ , we have

$$\mathbf{x}_{\text{cam}}(\theta) := X(\mathbf{p}_{\text{cam}}^{(0)}, \theta), \quad (25)$$

$$\mathbf{n}_{\text{cam}}(\theta) := \frac{X(\mathbf{p}_{\text{cam}}^{(1)}, \theta) - X(\mathbf{p}_{\text{cam}}^{(0)}, \theta)}{\|X(\mathbf{p}_{\text{cam}}^{(1)}, \theta) - X(\mathbf{p}_{\text{cam}}^{(0)}, \theta)\|}. \quad (26)$$

The position \mathbf{x}_{src} of a point light can be parameterized in a similar fashion as Eq. (25).

Handling discontinuities. The inclusion of the generalized geometric term $G(\mathbf{x}_N \leftrightarrow \mathbf{x}_{\text{cam}})$ in Eq. (23) and $G(\mathbf{x}_{\text{src}} \rightarrow \mathbf{x}_0)$ in Eq. (24) can violate the assumption (A.2) of W_e and L_e being continuous. Fortunately, this can be handled easily by including a new set of material boundary paths in the *boundary* term of Eq. (19).

Specifically, for pinhole cameras, we consider $\bar{\mathbf{p}} = (\mathbf{p}_0, \dots, \mathbf{p}_N)$ such that \mathbf{p}_N is a discontinuity point of $G(\mathbf{x}_N \leftrightarrow \mathbf{x}_{\text{cam}})$. In other words, we allow $\overline{\mathbf{x}_N \mathbf{x}_{\text{cam}}}$ to be a boundary segment (see Figure 5-b). Similarly, when handling point lights, we track discontinuities of \mathbf{x}_0 such that $\overline{\mathbf{x}_{\text{src}} \mathbf{x}_0}$ is effectively a boundary segment.

Other zero-measure detectors and sources. Using the formulations outlined in Eqs. (23) and (24), other zero-measure detectors (e.g., orthographic cameras) and sources (e.g., directional lights) can be handled in a similar manner. In case of a directional light with incident direction ω_{src} , we can encode its contributions in the source emission by letting

$$L_e(\mathbf{x}_0 \rightarrow \mathbf{x}_1) := f_V(\mathbf{x}_0, \omega_{\text{src}}, \overline{\mathbf{x}_0 \mathbf{x}_1}) \mathbb{V}(\mathbf{x}_0, \omega_{\text{src}}) \mathcal{I}, \quad (27)$$

where $\mathbb{V}(\mathbf{x}_0, \omega_{\text{src}})$, which can be discontinuous with respect to \mathbf{x}_0 , indicates whether a ray with origin \mathbf{x}_0 and direction ω_{src} can reach infinity without being occluded.

4.4 Relation to Prior Works

Relation to differential radiative transfer. Theoretically, our generalized differential path integral of Eq. (19) offers the same level of generality as the differential theory of radiative transfer (DTRT) [Zhang et al. 2019], since both formulations allow differentiating volumetric light transport with respect to arbitrary scene parameters.

On the other hand, our mathematical framework enjoys several significant advantages in practice.

Firstly, thanks to the material-form parameterization (§4.1), our formulation requires tracking fewer types of discontinuities. For example, DTRT involves *boundary* terms emerging from differentiating the line integral in the (integral-form) radiative transfer equation. Our formulation, on the other hand, only requires handling discontinuities resulting from the mutual visibility \mathbb{V} .

Secondly, being a path-space formulation, our technique allows the design of sophisticated Monte Carlo estimators (which we will discuss in §5) that are capable of handling complex light transport effects efficiently without the need of explicit silhouette detection.

Relation to path-space differentiable rendering. The differential path integral formulation introduced by Zhang et al. [2020] is limited to surface-only light transport and essentially a special case of Eq. (19). Further, Zhang et al. have assumed the absence of zero-measure sources, while we show how this can be relaxed in §4.3.

5 MONTE CARLO ESTIMATORS

We now derive new unbiased and consistent Monte Carlo estimators for our generalized differential path integral of Eq. (19). We focus on the problem of estimating $\frac{dI}{d\theta}(\theta_0)$ for some user-specified θ_0 . Further, we set the reference configurations (that is, the reference volume and surface) as the scene geometry at $\theta = \theta_0$ (as discussed in §4.1).

Thanks to the full separation between the *interior* and *boundary* terms in the generalized differential path integral, we estimate these terms in a separated fashion. In the rest of this section, we discuss the estimation of the *interior* integral in §5.1 and that of the *boundary* integral in §5.2. We keep discussions in this section high-level and provide some implementation details in §6.

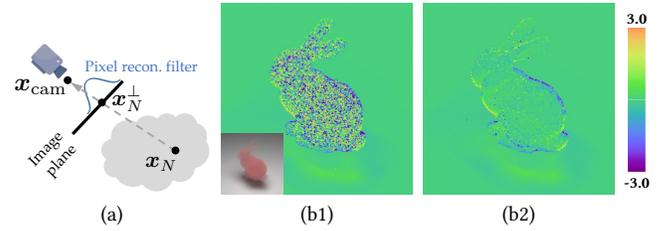


Fig. 6. **Pixel-level antithetic sampling:** As a function of \mathbf{x}_N^\perp , pixel reconstruction filters usually exhibit point symmetry (a), causing its spatial derivative to be an odd function that can produce high variance when estimating the *interior* integral (b1). With our pixel-level antithetic sampling, significant variance reduction can be achieved (b2). In this example, the derivatives in (b1) and (b2) are computed with respect to the bunny’s vertical position, and the ordinary image is shown as an inset in (b1).

5.1 Estimating the Interior Integral

With our choice of reference configurations, the material path space $\hat{\Omega}$ coincides with the ordinary one $\Omega(\theta_0)$. This allows estimating the *interior* integral in Eq. (19) using path sampling methods developed for forward rendering.

Specifically, we sample a material path $\bar{\mathbf{p}}$ with probability density $\text{prob}(\bar{\mathbf{p}})$ using standard techniques (such as volumetric path tracing or bidirectional path tracing). This path sampling process does not need to be differentiated since the material path space $\hat{\Omega}$ and all material paths $\bar{\mathbf{p}} \in \hat{\Omega}$ are independent of the scene parameter θ (and thus have zero derivatives).

With the material path $\bar{\mathbf{p}}$ constructed, we compute the corresponding ordinary path $\bar{\mathbf{x}} \in \Omega(\theta_0)$ by setting $\mathbf{x}_n = \mathcal{X}(\mathbf{p}_n, \theta_0)$ for each vertex \mathbf{p}_n of $\bar{\mathbf{p}}$. We note that, although \mathbf{x}_n takes the same value as \mathbf{p}_n for all n (as $\mathcal{X}(\cdot, \theta_0)$ reduces to the identity map with our choice of references), the derivative $\frac{d\mathbf{x}_n}{d\theta}(\theta_0)$ —which can be obtained by differentiating the motion \mathcal{X} —is typically nonzero.

Lastly, we compute the material measurement contribution \hat{f} of Eq. (16) in a differentiable fashion to obtain $\frac{d\hat{f}(\bar{\mathbf{p}})}{d\theta}(\theta_0)$. Returning this derivative divided by the path sampling probability $\text{prob}(\bar{\mathbf{p}})$ completes our Monte Carlo estimation of the *interior* term in Eq. (19).

Pixel-level antithetic sampling. Antithetic sampling is a classic variance-reduction framework for Monte Carlo estimation [Hammerly and Mauldon 1956]. When estimating integrals of the form $\int h(x) dx$ where h is an approximately odd function, it is desired to use correlated pairs samples x and $-x$ so that $h(x) + h(-x) \approx 0$. Recently, Zhang et al. [2021] have introduced this idea to differentiable rendering of glossy and near-specular materials.

We apply antithetic sampling at the pixel level. Specifically, when using perspective pinhole cameras given by Eqs. (22) and (23), the *interior* component of the generalized differential path integral involves derivative of the pixel reconstruction filter \mathcal{P} satisfying:

$$\frac{d\mathcal{P}}{d\theta} = \frac{\partial \mathcal{P}}{\partial \mathbf{x}_N^\perp} \frac{d\mathbf{x}_N^\perp}{d\theta}, \quad (28)$$

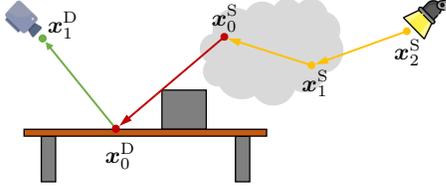


Fig. 7. We **rename the vertices** of a boundary path such that $\overline{x_0^S x_0^D}$ is the boundary segment (illustrated in red). The source and detector subpaths are shown in yellow and green, respectively. The arrows in this figure illustrate the physical flow of light (that is from the source to the detector) and do not indicate how the subpaths are constructed.

where x_N^\perp denotes the projection of x_N on the image plane (see Figure 6-a), and the derivative $dx_N^\perp/d\theta$ can be calculated based on $dx_N/d\theta$ given by our material-form parameterization $x_N = X(\mathbf{p}_N, \theta)$.

In practice, the pixel reconstruction filter \mathcal{P} , as a function of x_N^\perp , usually exhibit point symmetry with respect to the pixel center. This causes the (vector-valued) spatial derivative $\partial\mathcal{P}/\partial x_N^\perp$ to be an odd function. To reduce the variance introduced by this derivative, we generate pairs of samples that are point-symmetric with respect to the center of the pixel. Figure 6-b demonstrates the effectiveness of our pixel-level antithetic sampling.

5.2 Estimating the Boundary Integral

To estimate the *boundary* integral, Zhang et al. [2020] introduced a multi-directional sampling technique that starts with drawing the material boundary segment $\overline{\mathbf{p}_{K-1} \mathbf{p}_K}$ (such that \mathbf{p}_K is a jump discontinuity point of $\mathbb{V}(\mathbf{p}_{K-1} \leftrightarrow \mathbf{p}_K)$ when \mathbf{p}_K is fixed). This avoids explicit silhouette detections that can be expensive for complex scenes. Their technique, unfortunately, is derived for the surface-only case. In what follows, we generalize this technique to also support participating media.

Multi-directional boundary integral. To facilitate the development of our Monte Carlo estimator of the *boundary* integral, we first rewrite this integral in a *multi-directional form*. The derivations of Eq. (29)–(35) are mostly identical to those from the work by Zhang et al. [2020], and we show them nonetheless for completeness.

We start with renaming the vertices of a material boundary path $\bar{\mathbf{p}} \in \partial\Omega$:

$$\bar{\mathbf{p}} = (\mathbf{p}_s^S, \mathbf{p}_{s-1}^S, \dots, \mathbf{p}_0^S, \mathbf{p}_0^D, \mathbf{p}_1^D, \dots, \mathbf{p}_t^D), \quad (29)$$

such that \mathbf{p}_s^S and \mathbf{p}_t^D are located, respectively, on the source and the detector;³ and $\overline{\mathbf{p}_0^S \mathbf{p}_0^D}$ is the material boundary segment. Similarly, we rename vertices of the corresponding boundary path as $\bar{\mathbf{x}} = (x_s^S, x_{s-1}^S, \dots, x_0^S, x_0^D, x_1^D, \dots, x_t^D)$, as illustrated in Figure 7. Similar to the estimation of the *interior* integral (discussed in §5.1), under our choice of reference configurations, a material boundary path and its ordinary counterpart coincide.

This allows us to factorize the integrand of the *boundary* component of Eq. (19) into contributions of the segment $\overline{\mathbf{p}_0^S \mathbf{p}_0^D}$, the

³When the detector is a pinhole camera, as discussed in §4.3, \mathbf{p}_t^D is further connected to the camera's center of projection $\mathbf{p}_{\text{cam}}^{(0)}$ (instead of being on the sensor).

source subpath $\bar{\mathbf{p}}^S = (\mathbf{p}_s^S, \dots, \mathbf{p}_0^S)$, and the detector subpath $\bar{\mathbf{p}}^D := (\mathbf{p}_0^D, \dots, \mathbf{p}_t^D)$, respectively:

$$\Delta f(\bar{\mathbf{p}}) v(\mathbf{p}_K) = \underbrace{f^B(\mathbf{p}_0^S, \mathbf{p}_0^D)}_{\text{boundary seg.}} \underbrace{f^S(\bar{\mathbf{p}}^S; \mathbf{p}_0^D)}_{\text{src. subpath}} \underbrace{f^D(\bar{\mathbf{p}}^D; \mathbf{p}_0^S)}_{\text{det. subpath}}, \quad (30)$$

where

$$f^B(\mathbf{p}_0^S, \mathbf{p}_0^D) := \Delta G(x_0^S \leftrightarrow x_0^D) v(\mathbf{p}_0^D), \quad (31)$$

$$f^S(\bar{\mathbf{p}}^S; \mathbf{p}_0^D) := \hat{f}_v(\mathbf{p}_1^S \rightarrow \mathbf{p}_0^S \rightarrow \mathbf{p}_0^D) \prod_{n=1}^s \hat{f}_v(x_{n+1}^S \rightarrow x_n^S \rightarrow x_{n-1}^S) G(x_n^S \leftrightarrow x_{n-1}^S), \quad (32)$$

$$f^D(\bar{\mathbf{p}}^D; \mathbf{p}_0^S) := \hat{f}_v(\mathbf{p}_0^S \rightarrow \mathbf{p}_0^D \rightarrow \mathbf{p}_1^D) \prod_{n=1}^t \hat{f}_v(x_{n-1}^D \rightarrow x_n^D \rightarrow x_{n+1}^D) G(x_{n-1}^D \leftrightarrow x_n^D). \quad (33)$$

In Eqs. (32) and (33), G is the geometric term, and \hat{f}_v captures both per-vertex contribution f_v of Eq. (8) and the Jacobian determinant J of Eq. (17). That is, for any $\mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_3 \in \mathcal{B}_V$ and $x_n = X(\mathbf{p}_n, \theta)$ for $n = 1, 2, 3$:

$$\hat{f}_v(\mathbf{p}_1 \rightarrow \mathbf{p}_2 \rightarrow \mathbf{p}_3) := f_v(x_1 \rightarrow x_2 \rightarrow x_3) J(\mathbf{p}_2). \quad (34)$$

With its integrand expressed using Eq. (35), we can rewrite the *boundary* integral in its *multi-directional form* as:⁴

$$\iint \hat{f}^B(\mathbf{p}_0^S, \mathbf{p}_0^D) \left[\int \hat{f}^S(\bar{\mathbf{p}}^S; \mathbf{p}_0^D) d\bar{\mathbf{p}}^S \right] \left[\int \hat{f}^D(\bar{\mathbf{p}}^D; \mathbf{p}_0^S) d\bar{\mathbf{p}}^D \right] d\mathbf{p}_0^D d\mathbf{p}_0^S, \quad (35)$$

where the outer integral is over the material boundary segment $\overline{\mathbf{p}_0^S \mathbf{p}_0^D}$. Additionally, $\bar{\mathbf{p}}_0^S$ and $\bar{\mathbf{p}}_0^D$ denote the source and detector subpaths with endpoints \mathbf{p}_0^S and \mathbf{p}_0^D of the boundary segment excluded, respectively. That is, $\bar{\mathbf{p}}_0^S := (\mathbf{p}_s^S, \dots, \mathbf{p}_1^S)$ and $\bar{\mathbf{p}}_0^D := (\mathbf{p}_1^D, \dots, \mathbf{p}_t^D)$.

In the case of surface-only rendering, both \mathbf{p}_0^S and \mathbf{p}_0^D would always be surface vertices (Figure 3-a). With the presence of participating media, on the other hand, they both can be either a surface or a volume vertex, leading to three extra combinations (Figure 3-bcd).

Change of variable. To facilitate efficient sampling of the material boundary segment $\overline{\mathbf{p}_0^S \mathbf{p}_0^D}$, we apply a series of changes of variables to Eq. (35) as follows. First, we use the predetermined differentiable mapping $X(\cdot, \theta)$ to make the outer integral to be over the corresponding boundary segment $\overline{x_0^S x_0^D}$. In principle, this requires computing the Jacobian determinant $\|(dx_0^S dx_0^D)/(d\mathbf{p}_0^S d\mathbf{p}_0^D)\|$ based on the mapping $X(\cdot, \theta)$. In practice, because of our choice of reference configurations, the Jacobian determinant reduces to one (i.e., $\|(d\mathbf{p}_0^S d\mathbf{p}_0^D)/(dx_0^S dx_0^D)\| \equiv 1$).

Then, let \mathbf{x}^B be the intersection point between $\overline{x_0^S x_0^D}$ and the union of all object surfaces and ω^B be the direction of this boundary segment (i.e., a unit vector pointing from x_0^S to x_0^D). We apply another change of variable to make the outer integral to be on \mathbf{x}^B and ω^B . We note that the point \mathbf{x}^B is not a vertex of the resulting boundary path—we use it only for sampling purposes.

⁴We omit the integral domains and measures in Eq. (35) for notational simplicity.

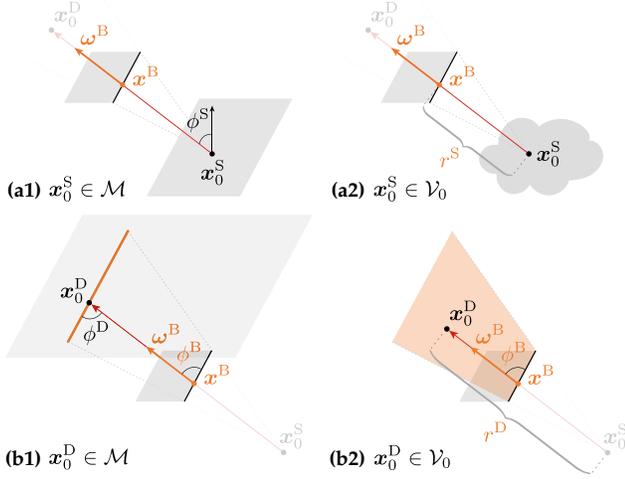


Fig. 8. Illustrations for deriving the **change-of-variable** Jacobian determinants expressed in Eqs. (40)–(43).

In what follows, we derive the Jacobian determinants corresponding to this change of variable. We base our derivations on the assumption that all surfaces in the scene are represented using polygonal meshes. In this case, \mathbf{x}^B will always belong to an edge of some polygonal face.

To facilitate the derivation of the Jacobian determinants, we first relate $d\mathbf{x}_0^S$ to $d\sigma(\omega^B)$ and $d\mathbf{x}_0^D$ to $d\ell(\mathbf{x}^B)$ as follows.

- When \mathbf{x}_0^S is a surface vertex, as illustrated in Figure 8-a1, we have

$$\frac{dA(\mathbf{x}_0^S) |\cos \phi^S|}{(r^S)^2} = d\sigma(\omega^B), \quad (36)$$

where $r^S := \|\mathbf{x}^B - \mathbf{x}_0^S\|$ is the distance between \mathbf{x}^B and \mathbf{x}_0^S ; ϕ^S is the angle between the segment $\overline{\mathbf{x}_0^S \mathbf{x}^B}$ and the surface normal at \mathbf{x}_0^S ; and σ is the solid-angle measure.

- When \mathbf{x}_0^S is a volume vertex, as shown in Figure 8-a2, we have

$$\frac{dV(\mathbf{x}_0^S)}{(r^S)^2} = d\sigma(\omega^B) dr^S. \quad (37)$$

- When \mathbf{x}_0^D is a surface vertex, for $\mathbf{x}_0^S \mathbf{x}_0^D$ to be a boundary segment with \mathbf{x}_0^S fixed, \mathbf{x}_0^D belongs to a curve (see Figure 8-b1). In this case, we have

$$\frac{d\ell(\mathbf{x}_0^D) \sin \phi^D}{r^D} = \frac{d\ell(\mathbf{x}^B) \sin \phi^B}{r^S}, \quad (38)$$

where $r^D := \|\mathbf{x}_0^D - \mathbf{x}_0^S\|$ is the distance between \mathbf{x}_0^D and \mathbf{x}_0^S , and ϕ^D is the angle between ω^B and the curve's tangent at \mathbf{x}_0^D .

- When \mathbf{x}_0^D is a volume vertex, it resides on a surface determined by the point \mathbf{x}^B and direction ω^B (see Figure 8-b2). It follows that

$$\frac{dA(\mathbf{x}_0^D)}{r^D} = \frac{d\ell(\mathbf{x}^B) \sin \phi^B}{r^S} dr^D. \quad (39)$$

Based on the relations given by Eqs. (36)–(39), we derive the Jacobian determinants for changes of variables from \mathbf{x}_0^S and \mathbf{x}_0^D to \mathbf{x}^B and ω^B (as well as r^S, r^D when needed). Specifically:

- When both \mathbf{x}_0^S and \mathbf{x}_0^D are surface vertices, according to Eqs. (36) and (38), we have

$$\left\| \frac{dA(\mathbf{x}_0^S) d\ell(\mathbf{x}_0^D)}{d\ell(\mathbf{x}^B) d\sigma(\omega^B)} \right\| = r^S r^D \frac{\sin \phi^B}{\sin \phi^D |\cos \phi^S|}. \quad (40)$$

- With \mathbf{x}_0^S being a surface vertex and \mathbf{x}_0^D a volume vertex, multiplying Eqs. (36) and (39) yields

$$\left\| \frac{dA(\mathbf{x}_0^S) dA(\mathbf{x}_0^D)}{d\ell(\mathbf{x}^B) d\sigma(\omega^B) dr^D} \right\| = r^S r^D \frac{\sin \phi^B}{|\cos \phi^S|}. \quad (41)$$

- With \mathbf{x}_0^S being a volume vertex and \mathbf{x}_0^D a surface vertex, multiplying Eqs. (37) and (38) gives

$$\left\| \frac{dV(\mathbf{x}_0^S) d\ell(\mathbf{x}_0^D)}{d\ell(\mathbf{x}^B) d\sigma(\omega^B) dr^S} \right\| = r^S r^D \frac{\sin \phi^B}{\sin \phi^D}. \quad (42)$$

- Lastly, when both \mathbf{x}_0^S and \mathbf{x}_0^D are volume vertices, according to Eqs. (37) and (39), we have

$$\left\| \frac{dV(\mathbf{x}_0^S) dA(\mathbf{x}_0^D)}{d\ell(\mathbf{x}^B) d\sigma(\omega^B) dr^S dr^D} \right\| = r^S r^D \sin \phi^B. \quad (43)$$

We note that, among the four cases discussed above, Zhang et al. [2020] only derived the first one—namely Eq. (40)—as Eq. (48) of their paper.

Using these relations, we can rewrite the multi-directional *boundary* integral of Eq. (35). When \mathbf{x}_0^S and \mathbf{x}_0^D are both volume vertices, for instance, we have⁵

$$\int_{\mathcal{E}} \int_{\mathbb{S}^2} \int_0^\infty \int_{r^S}^\infty \left[\hat{f}^B(\mathbf{p}_0^S, \mathbf{p}_0^D) r^S r^D \sin \phi^B \right] \left[\int_{\hat{\Omega}} \hat{f}^S d\mu \right] \left[\int_{\hat{\Omega}} \hat{f}^D d\mu \right] dr^D dr^S d\sigma(\omega^B) d\ell(\mathbf{x}^B), \quad (44)$$

where \mathcal{E} denotes the union of all face edges.

Boundary integrals of the other three cases (when at least one of \mathbf{x}_0^S and \mathbf{x}_0^D is a surface vertex) can be expressed in a similar fashion.

Sampling boundary segments. Based on the reparameterized *boundary* integrals like Eq. (44), we develop a generalized multi-directional sampling algorithm (Algorithm 1) to estimate the *boundary* component of the generalized differential path integral.

Our algorithm starts with sampling an interior point \mathbf{x}^B and direction ω^B of the boundary segment from some predetermined probability density \mathbb{P} (Line 3).

Then, we obtain the two endpoints \mathbf{x}_0^S and \mathbf{x}_0^D of the boundary segment as well as the corresponding probabilities prob^S and prob^D using the `sampleInteraction(x, ω)` function (Lines 4, 5). For any given \mathbf{x} and ω , this function returns a randomly sampled volume or surface vertex along the ray (\mathbf{x}, ω) , accompanied

⁵Strictly, the integral of ω^B in Eq. (44) should be over a subset of \mathbb{S}^2 so that the resulting boundary segment does not penetrate any surface. Please refer to the work by Zhang et al. [2020] for more details.

ALGORITHM 1: Monte Carlo estimator of the *boundary integral* (35)

```

1 EstimateBoundaryIntegral()
2 begin
   /* Sample boundary segment */
3   Draw  $(\mathbf{x}^B, \omega^B) \sim \mathbb{P}$ ;
4    $(\mathbf{x}_0^S, \text{prob}^S) \leftarrow \text{sampleInteraction}(\mathbf{x}^B, -\omega^B)$ ;
5    $(\mathbf{x}_0^D, \text{prob}^D) \leftarrow \text{sampleInteraction}(\mathbf{x}^B, \omega^B)$ ;
   /* Compute Jacobian determinant */
6    $r^S \leftarrow \|\mathbf{x}^B - \mathbf{x}_0^S\|$ ;  $r^D \leftarrow \|\mathbf{x}^B - \mathbf{x}_0^D\|$ ;
7    $J^B \leftarrow r^S r^D \sin \phi^B$ ;
8   if  $\mathbf{x}_0^S$  is a surface vertex then
9      $J^B \leftarrow J^B / |\cos \phi^S|$ ;
10  end
11  if  $\mathbf{x}_0^D$  is a surface vertex then
12     $J^B \leftarrow J^B / \sin \phi^D$ ;
13  end
   /* Evaluate boundary segment */
14   $\mathbf{p}_0^S \leftarrow \mathbf{x}_0^S$ ;  $\mathbf{p}_0^D \leftarrow \mathbf{x}_0^D$ ;
15   $T^B \leftarrow \frac{f^B(\mathbf{p}_0^S, \mathbf{p}_0^D) J^B}{\mathbb{P}(\mathbf{x}^B, \omega^B) \text{prob}^S \text{prob}^D}$ ;
   /* Sample and evaluate source & detector subpaths */
16   $T^S \leftarrow \text{EstimateSourcePath}(\mathbf{p}_0^S; \mathbf{p}_0^D)$ ;
17   $T^D \leftarrow \text{EstimateDetectorPath}(\mathbf{p}_0^D; \mathbf{p}_0^S)$ ;
18  return  $T^B T^S T^D$ ;
19 end

```

with the corresponding probability. Specifically, we follow the standard procedure in volumetric path tracing by first drawing a free-flight distance $t \geq 0$ from an exponential distribution with the pdf $p(t) = \sigma_t(\mathbf{x} + t\omega) \exp(-\int_0^t \sigma_t(\mathbf{x} + s\omega) ds)$ with σ_t being the extinction coefficient. For heterogeneous media, this can be achieved using techniques like delta tracking [Woodcock et al. 1965]. If the line segment connecting \mathbf{x} and $(\mathbf{x} + t\omega)$ does not intersect any object surface, the function returns a volume vertex of $(\mathbf{x} + t\omega)$. Otherwise, let $(\mathbf{x} + t_0\omega)$ be the first intersection along the ray (\mathbf{x}, ω) , `sampleInteraction` returns this point as a surface vertex.

Sampling subpaths. With the boundary segment $\overline{\mathbf{x}_0^S \mathbf{x}_0^D}$ drawn, we compute based on Eqs. (40)–(43) the corresponding Jacobian determinant J^B (Lines 6–12). This allows the contribution T^B of the boundary segment $\overline{\mathbf{x}_0^S \mathbf{x}_0^D}$ to be computed (Line 15).

Lastly, we estimate the contributions of the source and detector subpaths $\overline{\mathbf{p}^S}$ and $\overline{\mathbf{p}^D}$, respectively, using standard techniques such as volumetric path tracing (Lines 16 and 17), completing our estimation of the *boundary integral*.

6 IMPLEMENTATION DETAILS

We now discuss some important aspects for implementing the Monte Carlo estimators presented in §5.

6.1 Representing the Motion

A key ingredient of our material-form parameterization presented in §4.1 is the motion X that, for each θ , gives a differentiable bijection

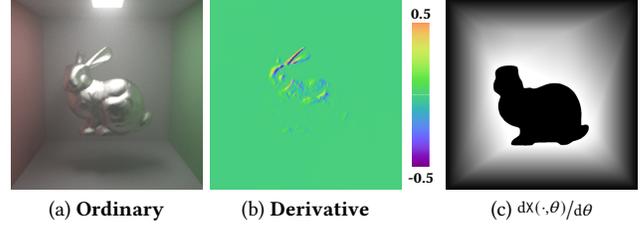


Fig. 9. Representing the motion: This example consists of a bunny inside a Cornell box filled with a homogeneous medium (a). To estimate the derivative with respect to the position of the bunny along the x -axis (b), we set the derivative $dX(\mathbf{p}, \theta)/d\theta$ of the motion to $[1, 0, 0]$ for all \mathbf{p} on the surface of the bunny, and to zero for \mathbf{p} on the box. To smoothly interpolate these derivatives in the interior of the medium, we tetrahedralize its volume (with boundaries given by the bunny and the box) and apply trilinear interpolation in each tetrahedron. We visualize a 2D slice of the resulting $dX(\cdot, \theta)/d\theta$ in (c), where the intensity of each pixel indicates the value of the derivative's x -component.

$X(\cdot, \theta)$ that maps the reference surface \mathcal{B}_M and volume \mathcal{B}_V to the object surfaces $\mathcal{M}(\theta)$ and volume $\mathcal{V}(\theta)$, respectively.

As aforementioned, we focus on the problem of estimating $dI/d\theta$ at some user-specified $\theta = \theta_0$ based on the generalized differential path integral of Eq. (19). When using the reference configurations discussed in §4.1, $X(\cdot, \theta_0)$ reduces to the identity map and does not have to be explicitly stored. Thus, it suffices to specify the derivative $[dX(\mathbf{p}, \theta)/d\theta]_{\theta=\theta_0}$ for each $\mathbf{p} \in \mathcal{B}_V$ as a vector field.

Affine deformation. If the deformation of an object is affine, the corresponding motion can be expressed as $X(\mathbf{p}, \theta) = \mathbf{R}(\theta)\mathbf{p} + \mathbf{t}(\theta)$, where $\mathbf{R}(\theta)$ is an invertible matrix and $\mathbf{t}(\theta)$ is a vector. Assuming that $\dot{\mathbf{R}} := d\mathbf{R}/d\theta$ and $\dot{\mathbf{t}} := dt/d\theta$ can be calculated analytically, we have $[dX(\mathbf{p}, \theta)/d\theta]_{\theta=\theta_0} = \dot{\mathbf{R}}(\theta_0)\mathbf{p} + \dot{\mathbf{t}}(\theta_0)$, which can be computed easily on the fly.

Non-affine deformation. To express general non-affine deformations, we use a tetrahedral mesh with the derivative values defined at each vertex. For the i -th vertex of the tetrahedral mesh, we store its position \mathbf{p}_i and the derivative $\dot{\mathbf{x}}_i := [dX(\mathbf{p}_i, \theta)/d\theta]_{\theta=\theta_0}$. In practice, this can be implemented by representing vertex positions as automatic-differentiation-enabled vectors \mathbf{x}_i . In this way, it holds that $\mathbf{p}_i = \text{detach}(\mathbf{x}_i)$, and $\dot{\mathbf{x}}_i$ can be obtained via automatic differentiation.

In the interior of each tetrahedron, we perform a trilinear interpolation to obtain the derivative. Although higher-order interpolations is possible, we find using a linear one to suffice for our purpose.

In practice, non-affine deformations such as stretching of a medium (as a continuum) are typically pre-specified using tetrahedral meshes. In this case, we reuse these meshes to represent our motion X .

When no tetrahedral mesh is provided as input, we tetrahedralize the input polygonal mesh using the TetGen library [Si 2015] so that the constructed tetrahedral mesh shares the same set of vertices as the input boundary mesh (see Figure 9 for an example). In our experiments, the computational overhead of this step is negligible.

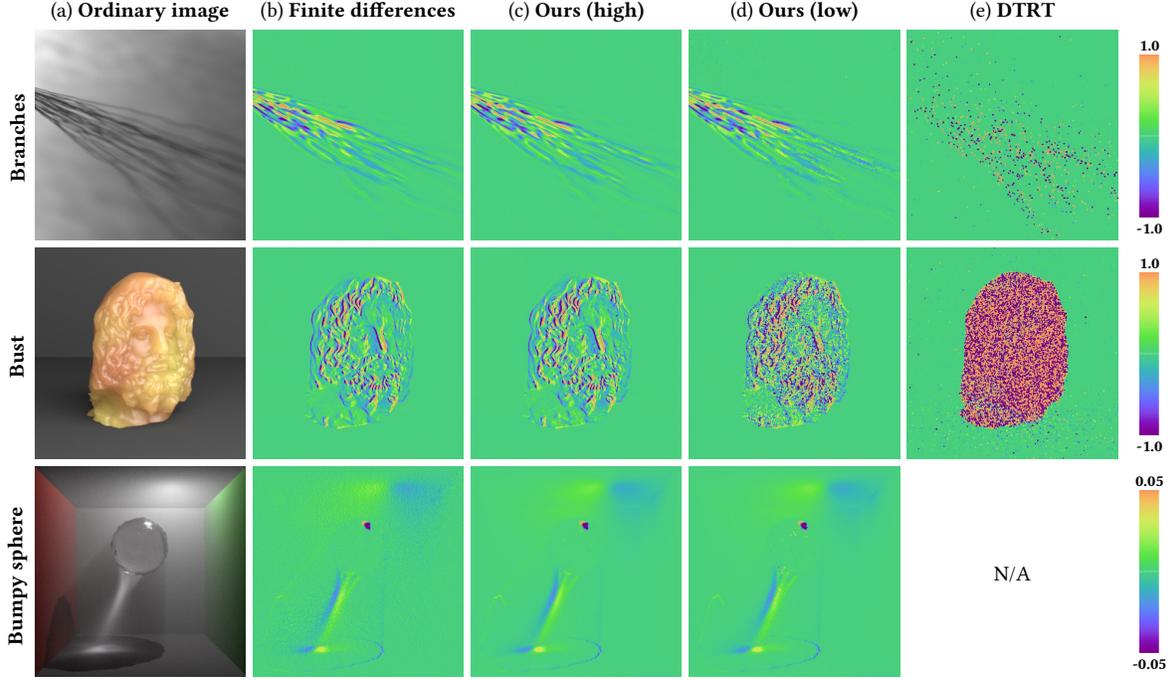


Fig. 10. **Differentiable-rendering comparisons:** (a) Ordinary images. (b) Derivatives obtained using finite differences (in a very long time). (c, d) Derivatives estimated using our technique with high and low sample counts, respectively. (e) Derivative estimates generated using the differentiable volumetric path tracing method introduced by Zhang et al. [2019]. Results in columns (d) and (e) are computed in equal time.

Alternatively, one can interpolate derivatives $[d\chi(\mathbf{p}, \theta)/d\theta]_{\theta=\theta_0}$ specified on a medium’s boundary to its interior using techniques like mean value coordinates [Thiery et al. 2018].

6.2 Computing Jacobian Determinants

Another important term to our material-form parameterization is the Jacobian determinant expressed in Eq. (17).

Under our choice of references, when \mathbf{p} is a surface vertex located on a triangle (that is a face of some tetrahedron) with vertices \mathbf{p}_A , \mathbf{p}_B and \mathbf{p}_C , the Jacobian determinant $\|dA(\mathbf{x})/dA(\mathbf{p})\|$ captures the change rate of this triangle’s area with respect to θ and is constant in the interior of the triangle. Specifically, we have

$$\left\| \frac{dA(\mathbf{x})}{dA(\mathbf{p})} \right\| = \frac{\|(\mathbf{x}_A - \mathbf{x}_C) \times (\mathbf{x}_B - \mathbf{x}_C)\|}{\|(\mathbf{p}_A - \mathbf{p}_C) \times (\mathbf{p}_B - \mathbf{p}_C)\|}, \quad (45)$$

where “ \times ” denotes the cross product of two vectors.

When \mathbf{p} is a volume vertex inside a tetrahedron with vertices \mathbf{p}_A , \mathbf{p}_B , \mathbf{p}_C and \mathbf{p}_D , the Jacobian determinant $\|dV(\mathbf{x})/dV(\mathbf{p})\|$ indicates the change rate of this tetrahedron’s volume:

$$\left\| \frac{dV(\mathbf{x})}{dV(\mathbf{p})} \right\| = \frac{|(\mathbf{x}_A - \mathbf{x}_D) \cdot ((\mathbf{x}_B - \mathbf{x}_D) \times (\mathbf{x}_C - \mathbf{x}_D))|}{|(\mathbf{p}_A - \mathbf{p}_D) \cdot ((\mathbf{p}_B - \mathbf{p}_D) \times (\mathbf{p}_C - \mathbf{p}_D))|}. \quad (46)$$

In practice, as described in §6.1, by expressing vertex positions as automatic-differentiation-enabled vectors, the derivatives of Eqs. (45) and (46) can be obtained easily.

Continuity property. Given Eqs. (45) and (46), it holds that the Jacobian determinant J defined in Eq. (17) is piecewise constant

with discontinuity boundaries being the faces of the tetrahedra. Since the tetrahedron mesh is defined in the reference volume and independent of the scene parameter θ , so are the discontinuity boundaries of J , satisfying our continuity assumption of A.4.

6.3 Computing Change Rates of Discontinuity Boundaries

A key term in the boundary component of our generalized differential path integral of Eq. (19) is $v(\mathbf{p}_K)$ that captures the change rates of discontinuities boundaries with respect to the scene parameter θ .

In practice, evaluating this term using Eq. (20) largely amounts to computing $d\mathbf{p}_K/d\theta$ that, in turn, requires parameterizing the corresponding discontinuity curve or surface near \mathbf{p}_K . Under our multi-directional formulation described in §5.2, \mathbf{p}_K is renamed as \mathbf{p}_0^D . In what follows, we discuss the computation of $v(\mathbf{p}_0^D)$ at some $\theta = \theta_0$ with the motion χ represented as discussed in §6.1.

To obtain \mathbf{p}_0^D , we first compute $\mathbf{x}_0^D \in \mathcal{V}(\theta_0)$ in a differentiable fashion and then transform it back to the reference configuration. Without loss of generality, assume that

$$\mathbf{x}^B = \xi_1 \mathbf{x}_P + (1 - \xi_1) \mathbf{x}_Q, \quad \omega^B = \frac{\mathbf{x}^B - \mathbf{x}_0^S}{\|\mathbf{x}^B - \mathbf{x}_0^S\|}, \quad (47)$$

where $\mathbf{x}_P, \mathbf{x}_Q \in \mathcal{M}(\theta_0)$ are the positions of two mesh vertices (such that the face edge $\overline{\mathbf{x}_P \mathbf{x}_Q}$ contains \mathbf{x}^B), and $\xi_1 \in [0, 1]$ is some real number independent of θ . In Eq. (47), $\mathbf{x}_P, \mathbf{x}_Q$, and \mathbf{x}_0^S can all be expressed as automatic-differentiation-enabled vectors (as discussed in §6.1). In what follows, we discuss how \mathbf{x}_0^D and \mathbf{p}_0^D —which depend

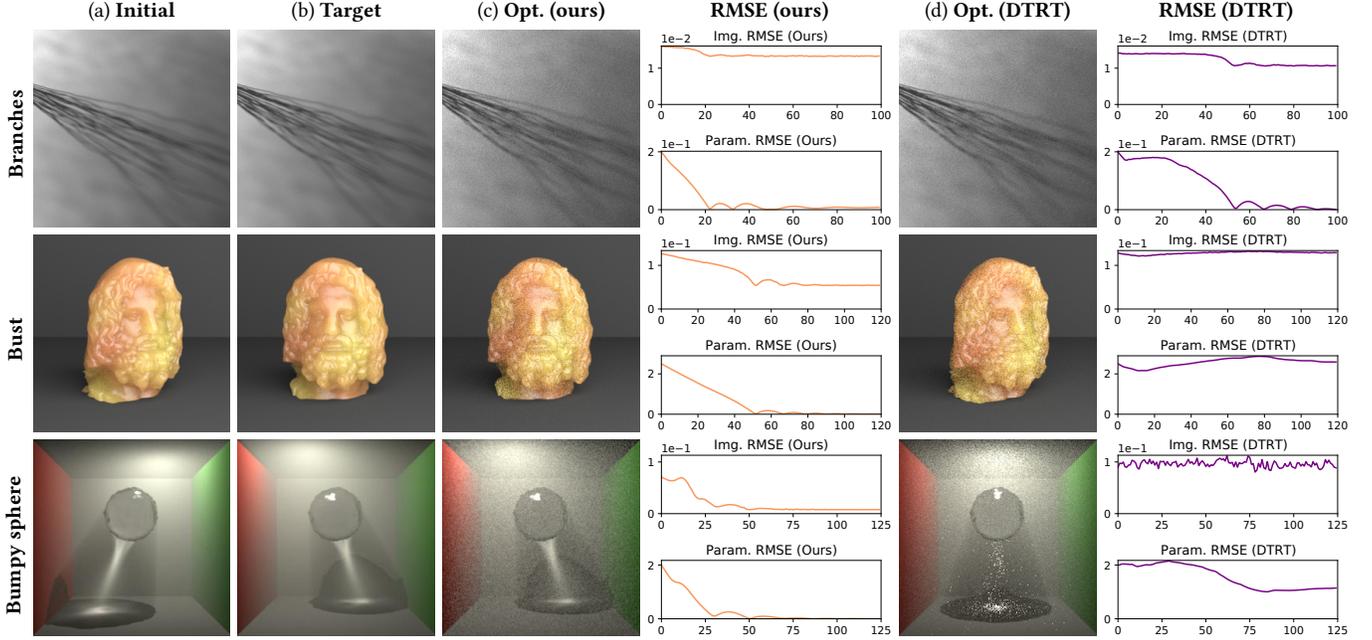


Fig. 11. **Inverse-rendering comparisons** using gradients estimated with our technique and DTRT [Zhang et al. 2019]. For each example, we use identical optimization algorithms (i.e., Adam [Kingma and Ba 2014]), objective functions (i.e., image RMSE), learning rates, and initial configurations. We also adjust the sample count so that both methods take approximately equal time per iteration. The parameter RMSE is used only for evaluation (and not for optimization). Images in columns (a, c, d) contain Monte Carlo noise as we use noisy (but unbiased) estimates for the inverse-rendering optimizations.

on the scene parameter θ in general—can be computed in a differentiable fashion given \mathbf{x}^B and ω^B . After obtaining the derivative $[d\mathbf{p}_0^D/d\theta]_{\theta=\theta_0}$, we can compute the change rate $v(\mathbf{p}_0^D)$ using Eq. (20).

Surface case. When \mathbf{x}_0^D is a surface vertex, as illustrated in Figure 8-b1, \mathbf{x}_0^D and its derivative $[d\mathbf{x}_0^D/d\theta]_{\theta=\theta_0}$ can be computed using differentiable ray tracing:

$$\mathbf{x}_0^D = \text{rayTrace}(\mathbf{x}^B, \omega^B). \quad (48)$$

Then, we obtain \mathbf{p}_0^D by transforming \mathbf{x}_0^D back to the reference surface. Assume that

$$\mathbf{x}_0^D = (1 - u_1 - u_2) \mathbf{x}_A + u_1 \mathbf{x}_B + u_2 \mathbf{x}_C, \quad (49)$$

where: $\mathbf{x}_A, \mathbf{x}_B, \mathbf{x}_C \in \mathcal{M}(\theta_0)$ are vertices of the mesh face containing \mathbf{x}_0^D ; (u_1, u_2) are barycentric coordinates of \mathbf{x}_0^D within the triangular face. Further, $\mathbf{x}_A, \mathbf{x}_B, \mathbf{x}_C, u_1$, and u_2 are determined by the differentiable ray tracing process and can all depend on the scene parameter θ . Given Eq. (49), it follows that

$$\mathbf{p}_0^D = (1 - u_1 - u_2) \mathbf{p}_A + u_1 \mathbf{p}_B + u_2 \mathbf{p}_C, \quad (50)$$

where $\mathbf{p}_* = \mathbf{X}^{-1}(\mathbf{x}_*, \theta_0) = \text{detach}(\mathbf{x}_*)$ for each $* \in \{A, B, C\}$.

We note that, given Eqs. (47)–(50), we essentially parameterize the discontinuity curve near \mathbf{p}_0^D using ξ_1 .

Volume case. When \mathbf{x}_0^D is a volume vertex, as illustrated in Figure 8-b2, it must lie on the discontinuity plane determined by \mathbf{x}_0^S and the face edge $\overline{\mathbf{x}_P \mathbf{x}_Q}$ containing \mathbf{x}^B . Assume that

$$\mathbf{x}_0^D = \mathbf{x}_0^S + \xi_2 (\mathbf{x}^B - \mathbf{x}_0^S), \quad (51)$$

Table 2. Performance statistics for the inverse-rendering results in Figures 11 and 12. The “time” numbers indicate average computation time per iteration for both our technique and DTRT [Zhang et al. 2019] (when applicable). The experiments are conducted on a workstation equipped with 8-core intel i7-7820X CPU.

Scene	Branches (Fig. 11)	Bust (Fig. 11)	Bumpy sph. (Fig. 11)	Spheres (Fig. 12)	Bunny (Fig. 12)	Pool (Fig. 12)
# param.	1	1	2	2	17	100
# iter.	100	120	125	150	125	300
time	39s	3m 24s	23s	3m 15s	9m 35s	5m 52s

for some $\xi_2 \geq 1$. Then, the discontinuity plane containing \mathbf{x}_0^D is effectively parameterized with ξ_1 and ξ_2 via Eqs. (47) and (51).

When the motion \mathbf{X} is affine, as discussed in §6.1, we have

$$\mathbf{p}_0^D = \mathbf{R}^{-1}(\mathbf{x}_0^D - \mathbf{t}). \quad (52)$$

When a tetrahedral mesh is used to express \mathbf{X} , assume that \mathbf{x}_0^D is located inside a tetrahedron with vertices $\mathbf{x}_A, \mathbf{x}_B, \mathbf{x}_C, \mathbf{x}_D \in \mathcal{V}(\theta_0)$ and has barycentric coordinates (u_1, u_2, u_3) . Similar to the surface case, $\mathbf{x}_A, \mathbf{x}_B, \mathbf{x}_C, \mathbf{x}_D, u_1, u_2$, and u_3 all depend on θ in general. Then, it holds that

$$\mathbf{p}_0^D = (1 - u_1 - u_2 - u_3) \mathbf{p}_A + u_1 \mathbf{p}_B + u_2 \mathbf{p}_C + u_3 \mathbf{p}_D, \quad (53)$$

where $\mathbf{p}_* = \text{detach}(\mathbf{x}_*)$ for all $* \in \{A, B, C, D\}$.

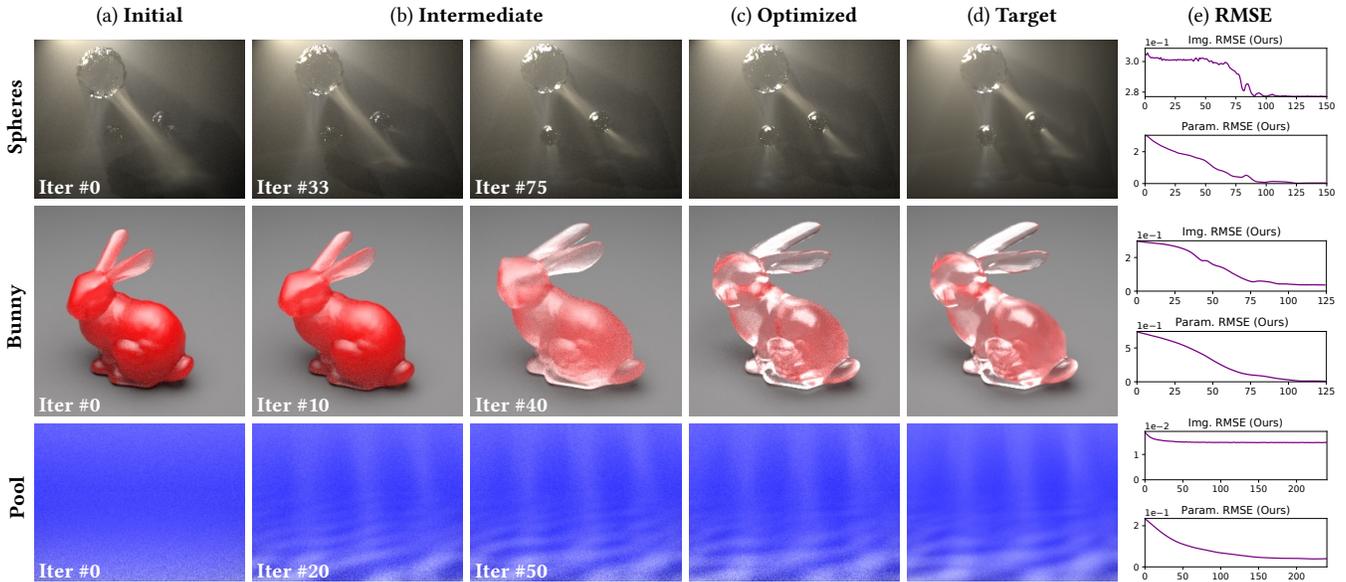


Fig. 12. **Inverse-rendering results** using gradients estimated with our technique. Images in columns (a–c) contain Monte Carlo noise since we use noisy (but unbiased) estimates for the optimizations. Please refer to the supplement for animated versions of these results.

7 RESULTS

We implement our techniques presented in §4–§6 in C++ on the CPU. Specifically, we develop a Monte Carlo estimator that samples material light paths \bar{p} using unidirectional path tracing. Additionally, we build another estimator that utilizes a bidirectional path sampling scheme to handle challenging light transport effects such as caustics.

In the rest of this section, we validate our implementations and demonstrate their effectiveness in §7.1 and show more inverse-rendering examples in §7.2. Please refer to the supplemental material for more animated versions of the inverse-rendering results.

7.1 Validation and Evaluation

We use the following virtual scenes to evaluate the correctness and effectiveness of our Monte Carlo estimators:

- The **branches** scene is modeled after that with the same name from Zhang et al.’s work [2020]. This scene contains a tree-like object with fine structures (that is outside the field of view) casting a shadow on the ground. This object is further embedded within an optically thin heterogeneous medium.
- The **bust** scene contains a translucent bust with complex geometry and spatially varying scattering properties. The bust is optically thick and, thus, exhibits strong multiple scattering.
- The **bumpy sphere** scene consists of a sphere made of rough glass inside a box filled with a homogeneous participating medium. The sphere is lit by a point light from above, yielding strong volumetric shadow and caustic effects.

We use perspective pinhole cameras for all these scenes.

Differentiable rendering comparisons. In Figure 10, we show derivative images of these scenes generated using finite differences, our

technique, and the differential theory of radiative transfer (DTRT) [Zhang et al. 2019].

For the *branches* scene, we compute derivatives with respect to the rotation angle of the object around the vertical axis. Derivative image generated using our unidirectional estimator closely matches the reference obtained using finite differences. Because of the complex visibility, DTRT—which relies on explicit detection of object silhouettes—produces highly noisy derivative estimates. Our method, on the other hand, does not require silhouette detection and can generate much cleaner results in equal time.

For the *bust* scene, we differentiate the ordinary image with respect to the rotation angle of the translucent bust. Our results, which match the finite-differences reference, are generated using our unidirectional estimator. Due to the presence of complex geometry and high-order scatterings, DTRT produces extremely noisy results. In equal time, our technique is capable of generating derivative estimates with significantly lower noise.

For the *bumpy sphere* scene, we estimate derivatives with respect to the horizontal translation of the light source. Due to the complexity of light transport effects in this example, the finite-differences result contains some Monte Carlo noise even after being rendered for many hours. Our results obtained using our bidirectional algorithm accurately captures the gradients around specular highlights on the sphere as well as shadow and caustic boundaries in the volume and on the ground. DTRT does not work for this example due to the lack of support for point light sources.

Inverse rendering comparisons. We further demonstrate in Figure 11 the usefulness of our low-variance derivative estimates by comparing inverse-rendering performance of our method and DTRT. For all examples, we minimize the image RMSE as the choice of

loss function is orthogonal to our technique. We use the Adam algorithm [Kingma and Ba 2014] with identical settings including initial configurations and learning rates for both methods. We also adjust sample counts so that each optimization iteration takes approximately equal time. Additionally, for the *bumpy sphere* scene, we replace the point light with a small area light since DTRT only supports the latter. Please see Table 2 for performance statistics and the supplement for animated versions of these results.

For the *branches* scene, gradient-based optimizations driven by both methods converge correctly. Our method offers faster convergences thanks to its cleaner derivative estimates. For the *bust* and *bumpy sphere* scenes, optimizations using gradients estimated by DTRT fail to converge due to very high variance in the estimated gradients. Our method, on the other hand, allows smooth convergence to the correct results.

7.2 Inverse-Rendering Results

We show additional inverse-rendering results using derivatives estimated with our method in Figure 12:

- The **spheres** scene contains three glass spheres within a homogeneous medium lit by a small area light, yielding complex volumetric caustics. We search for the position of the larger sphere (expressed using 2 variables) to match the target caustics pattern.
- The **bunny** scene contains a homogeneous translucent bunny that exhibits rough refraction on the surface and multiple scattering in the interior. Further, we use a control cage with 16 vertices to deform the bunny in a non-rigid fashion (with the vertex weights computed using mean value coordinates [Thiery et al. 2018]). We jointly optimize: (i) the positions of five vertices on the control cage (expressed using 15 variables); (ii) the bunny's surface roughness and the medium's optical density.
- The **pool** scene consists of a curved refractive interface producing caustics in the medium and the surface below. We optimize the shape of the refractive interface (parameterized with 100 variables).

For all three examples, using gradient estimates generated with our technique, inverse-rendering optimizations successfully recover the target parameters. Please refer to the supplemental materials to see animated version of these results demonstrating the full optimization processes.

8 DISCUSSION AND CONCLUSION

Limitations and future work. Our technique works mainly in the (generalized) path space. Due to the success of primary-sample-space (PSS) methods in forward rendering, adopting our technique to work in that space—potentially by using different choices of reference surfaces and volumes—can be an interesting future topic. Also, extending our technique to handle implicit scene geometries (such as interfaces defined using signed distance functions) will enable inverse-rendering optimizations robust to changes of topology. Lastly, implementing our estimators on the GPU with state-of-the-art computational differentiation techniques will allow solving challenging inverse-rendering problems with large numbers (i.e.,

millions) of unknowns, benefiting a large array of real-world applications.

Conclusion. In this paper, we introduced the mathematical formulation of generalized differential integrals that offers the generality of differentiating both interfacial and volumetric light transport with respect to arbitrary scene parameters include global object geometries. Utilizing this formulation, we developed new Monte Carlo estimators that handle the *interior* and *boundary* components of generalized differential integrals separately. Our estimators are unbiased, consistent, and capable of efficiently handling complex scene geometries and light transport effects such as multiple scattering and volumetric caustics. We demonstrated the effectiveness of our technique via a few differentiable-rendering and inverse-rendering examples.

ACKNOWLEDGMENTS

We thank the anonymous reviewers for their constructive comments and suggestions. This work was partially supported by NSF grant 19090927.

REFERENCES

- Sai 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.
- Eva Cerezo, Frederic Pérez, Xavier Pueyo, Francisco J Seron, and François X Sillion. 2005. A survey on participating media rendering techniques. *The Visual Computer* 21, 5 (2005), 303–328.
- Subrahmanyam Chandrasekhar. 1960. *Radiative transfer*. Courier Corporation.
- Chengqian Che, Fujun Luan, Shuang Zhao, Kavita Bala, and Ioannis Gkioulekas. 2020. Towards Learning-based Inverse Subsurface Scattering. In *2020 IEEE International Conference on Computational Photography (ICCP)*. IEEE, 1–12.
- Ioannis Gkioulekas, Shuang Zhao, Kavita Bala, Todd Zickler, and Anat Levin. 2013. Inverse volume rendering with material dictionaries. *ACM Trans. Graph.* 32, 6 (2013), 162:1–162:13.
- John Michael Hammersley and JG Mauldon. 1956. General principles of antithetic variates. In *Mathematical proceedings of the Cambridge philosophical society*, Vol. 52. Cambridge University Press, 476–481.
- Wenzel Jakob and Steve Marschner. 2012a. Manifold Exploration: A Markov Chain Monte Carlo technique for rendering scenes with difficult specular transport. *ACM Trans. Graph.* 31, 4 (2012), 58:1–58:13 pages.
- Wenzel Jakob and Steve Marschner. 2012b. Manifold exploration: A Markov chain Monte Carlo technique for rendering scenes with difficult specular transport. *ACM Trans. Graph.* 31, 4 (2012), 58:1–58:13.
- James T. Kajiya. 1986. The Rendering Equation. In *SIGGRAPH '86*. 143–150.
- James T. Kajiya and Brian P Von Herzen. 1984. Ray tracing volume densities. *SIGGRAPH Comput. Graph.* 18, 3 (1984), 165–174.
- Csaba Kelemen, László Szirmay-Kalos, György Antal, and Ferenc Csonka. 2002. A simple and robust mutation strategy for the metropolis light transport algorithm. In *Computer Graphics Forum*, Vol. 21. Wiley Online Library, 531–540.
- Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).
- Eric P LaFortune and Yves D Willems. 1996. Rendering participating media with bidirectional path tracing. In *Rendering techniques '96*. Springer, 91–100.
- 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.
- Merlin Nimier-David, Delio Vicini, Tizian Zeltner, and Wenzel Jakob. 2019. Mitsuba 2: A retargetable forward and inverse renderer. *ACM Trans. Graph.* 38, 6 (2019), 203:1–203:17.
- Jan Novák, Iliyan Georgiev, Johannes Hanika, and Wojciech Jarosz. 2018. Monte Carlo methods for volumetric light transport simulation. In *Computer Graphics Forum*, Vol. 37. Wiley Online Library, 551–576.
- Mark Pauly, Thomas Kollig, and Alexander Keller. 2000. Metropolis light transport for participating media. In *Rendering Techniques 2000*. Springer, 11–22.
- Osborne Reynolds. 1903. *Papers on mechanical and physical subjects: the sub-mechanics of the universe*. Vol. 3. The University Press.

- Hang Si. 2015. TetGen, a Delaunay-based quality tetrahedral mesh generator. *ACM Transactions on Mathematical Software (TOMS)* 41, 2 (2015), 1–36.
- Jerome Spanier and Ely M Gelbard. 1969. *Monte Carlo principles and neutron transport problems*. The Addison-Wesley Publishing Company.
- Denis Sumin, Tobias Rittig, Vahid Babaei, Thomas Nindel, Alexander Wilkie, Piotr Didyk, Bernd Bickel, Jaroslav Krivánek, Karol Myszkowski, and Tim Weyrich. 2019. Geometry-aware scattering compensation for 3D printing. *ACM Trans. Graph.* 38, 4 (2019), 111:1–111:14.
- Jean-Marc Thiery, Pooran Memari, and Tamy Boubekeur. 2018. Mean value coordinates for quad cages in 3D. *ACM Trans. Graph.* 37, 6 (2018), 229:1–229:14.
- Eric Veach. 1997. *Robust Monte Carlo methods for light transport simulation*. Vol. 1610. Stanford University PhD thesis.
- Eric Veach and Leonidas Guibas. 1995. Bidirectional estimators for light transport. In *Photorealistic Rendering Techniques*. Springer, 145–167.
- Eric Veach and Leonidas J. Guibas. 1997. Metropolis light transport. In *Proceedings of the 24th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '97)*. ACM Press/Addison-Wesley Publishing Co., 65–76.
- 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.
- 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.
- Shuang Zhao, Lifan Wu, Frédo Durand, and Ravi Ramamoorthi. 2016. Downsampling scattering parameters for rendering anisotropic media. *ACM Trans. Graph.* 35, 6 (2016), 166:1–166:11.

A DERIVING THE GENERALIZED DIFFERENTIAL PATH INTEGRAL

To derive derivatives of Eq. (15), we first rewrite this equation as

$$I = \sum_{N \geq 1} \sum_{l=0}^{2^{N+1}-1} \underbrace{\int_{\hat{\Omega}_N^l} \hat{f}(\bar{\mathbf{p}}) d\mu_{N,0}^l(\bar{\mathbf{p}})}_{=: I_N^l}, \quad (54)$$

where $\hat{\Omega}_N^l$ follows the definition in Eq. (12). Then, deriving $dI/d\theta$ amounts to differentiating I_N^l for any fixed $N > 0$ and path characteristic $0 \leq l < 2^{N+1}$. To this end, we facilitate the proof introduced by Zhang et al. [2020] for the surface-only case as follows.

First, we rewrite I_N^l in a recursive fashion by defining

$$h_N(\mathbf{p}_N; \mathbf{p}_{N-1}) := J(\mathbf{p}_N) W_e(\mathbf{x}_{N-1} \rightarrow \mathbf{x}_N), \quad (55)$$

and, for $0 \leq n < N$,

$$h_n(\mathbf{p}_n; \mathbf{p}_{n-1}) := \int_{\mathcal{B}_{n+1}} h_{n+1}(\mathbf{p}_{n+1}; \mathbf{p}_n) g(\mathbf{x}_{n+1}; \mathbf{x}_{n-1}, \mathbf{x}_n) d\mu_{N,n+1}^l(\mathbf{p}_{n+1}), \quad (56)$$

where $d\mu_{N,n+1}^l$ is defined in Eq. (4) and

$$g(\mathbf{x}_{n+1}; \mathbf{x}_{n-1}, \mathbf{x}_n) := J(\mathbf{p}_n) f_v(\mathbf{x}_{n-1} \rightarrow \mathbf{x}_n \rightarrow \mathbf{x}_{n+1}) G(\mathbf{x}_n \leftrightarrow \mathbf{x}_{n+1}). \quad (57)$$

In Eqs. (55) and (57), the function J follows the definition in Eq. (17); in Eq. (56), the domain of integration \mathcal{B}_{n+1} depends on $b_{n+1}(l)$, the $(n+1)$ -th bit of the given path characteristic l :

$$\mathcal{B}_{n+1} = \begin{cases} \mathcal{B}_{\mathcal{M}}, & b_{n+1}(l) = 1 \\ \mathcal{B}_{\mathcal{V}_0}. & b_{n+1}(l) = 0 \end{cases} \quad (58)$$

Given the definitions of Eqs. (55)–(58), it is easy to verify that

$$I_N^l = \int_{\mathcal{B}_0} h_0(\mathbf{p}_0) d\mu_{N,0}^l(\mathbf{p}_0). \quad (59)$$

Then, deriving $dI_N^l/d\theta$ amounts to differentiating Eq. (56). Under Assumptions A.1–A.4, discontinuities of the integrand of Eq. (56) emerge solely from the mutual visibility function that is a factor of the g term. It follows that applying Reynolds transport theorem [Reynolds 1903] yields:

$$\frac{dh_n}{d\theta} = \int_{\mathcal{B}_{n+1}} \frac{d}{d\theta} (h_{n+1} g) d\mu_{N,n+1}^l + \int_{\partial\mathcal{B}_{n+1}} h_{n+1} \Delta g V_{n+1} d\mu_{N,n+1}^l. \quad (60)$$

In this equation:

- $\partial\mathcal{B}_{n+1} \subset \mathcal{B}_{n+1}$ denotes the discontinuity points of g with respect to \mathbf{p}_{n+1} when \mathbf{p}_{n-1} and \mathbf{p}_n are fixed. Specifically, if $b_{n+1}(l) = 1$, the integral in Eq. (56) is over all object surfaces $\mathcal{B}_{\mathcal{M}}$, and $\partial\mathcal{B}_{n+1}$ is a set of curves. If $b_{n+1}(l) = 0$, we have a volumetric integral in Eq. (56), and $\partial\mathcal{B}_{n+1}$ involves a collection of surfaces.
- V_{n+1} is the scalar normal velocity of a discontinuity point $\mathbf{p}_{n+1} \in \partial\mathcal{B}_{n+1}$.
- Δg indicates the difference in g across the discontinuity boundaries.
- The measure $\mu_{N,n+1}^l$ is defined as

$$d\mu_{N,n+1}^l := \begin{cases} d\ell, & b_{n+1}(l) = 1 \\ dA, & b_{n+1}(l) = 0 \end{cases} \quad (61)$$

where ℓ and A are, respectively, curve-length and surface-area measures.

With Eqs. (56) and (60), we can now differentiate I_N^l defined in Eq. (59) by repeatedly expanding h_n and $dh_n/d\theta$ for $n = 0, 1, \dots, N-1$, resulting in:

$$\frac{dI_N^l}{d\theta} = \underbrace{\int_{\hat{\Omega}_N^l} \frac{d\hat{f}(\bar{\mathbf{p}})}{d\theta} d\mu_{N,0}^l(\bar{\mathbf{p}})}_{\text{interior}} + \underbrace{\sum_{K=0}^N \left[\int_{\partial\hat{\Omega}_{N,K}^l} \Delta \hat{f}_K(\bar{\mathbf{p}}) V_K(\mathbf{p}_K) d\mu_{N,K}^l(\bar{\mathbf{p}}) \right]}_{\text{boundary}}, \quad (62)$$

where

$$\partial\hat{\Omega}_{N,K}^l := \left(\prod_{n=0}^{K-1} \mathcal{B}_n \right) \times \partial\mathcal{B}_K \times \left(\prod_{n=K+1}^N \mathcal{B}_n \right), \quad (63)$$

$$d\mu_{N,K}^l(\bar{\mathbf{p}}) := d\mu_{N,K}^l(\mathbf{p}_K) \prod_{\substack{0 \leq n \leq N \\ n \neq K}} d\mu_{N,n}^l(\mathbf{p}_n), \quad (64)$$

$$\Delta \hat{f}_K(\bar{\mathbf{p}}) := \hat{f}(\bar{\mathbf{p}}) \frac{\Delta g(\mathbf{x}_{K+1}; \mathbf{x}_{K-1}, \mathbf{x}_K)}{g(\mathbf{x}_{K+1}; \mathbf{x}_{K-1}, \mathbf{x}_K)}. \quad (65)$$

Lastly, we can sum up Eq. (62) for all $N \geq 1$ and $0 \leq l \leq 2^{N+1} - 1$ to obtain our generalized differential path integral of Eq. (19).