Extended Path Space Manifolds for Physically Based Differentiable Rendering

Jiankai Xing
xjk21@mails.tsinghua.edu.cn
BNRist, Department of CS&T,
Tsinghua University
Beijing, China

Xuejun Hu
huxj19@mails.tsinghua.edu.cn
BNRist, Department of CS&T,
Tsinghua University
Beijing, China

Fujun Luan
fluan@adobe.com
Adobe Research
USA

Ling-Qi Yan
lingqi@cs.ucsb.edu
University of California, Santa Barbara
USA

Kun Xu*
xukun@tsinghua.edu.cn
BNRist, Department of CS&T,
Tsinghua University
Beijing, China

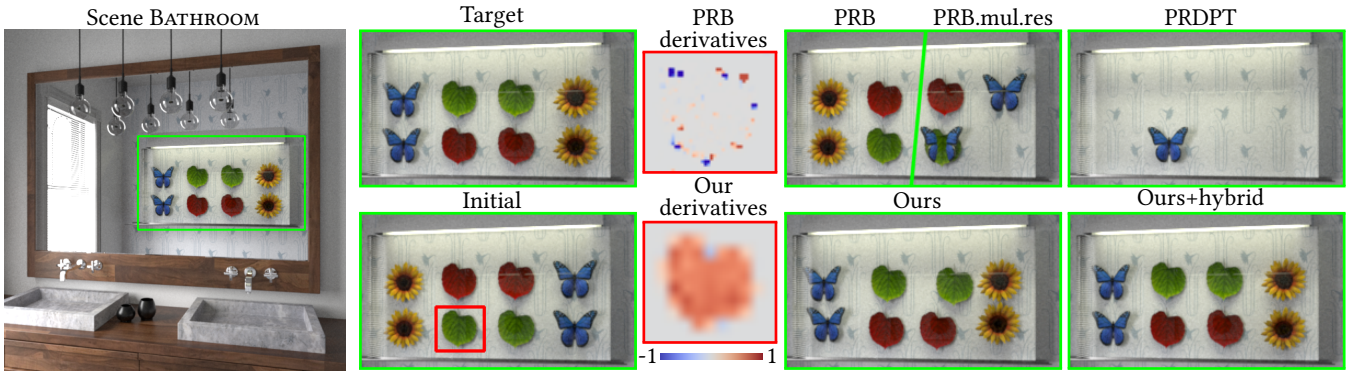


Figure 1: Optimizing the 2D translation vectors of eight specimen objects via physically based differentiable rendering. Note that the specimen objects are placed inside a glass box and are viewed through nested reflection and refraction. Given the target image and initial scene parameters, state-of-the-art methods including Path Replay Backpropagation with reparameterization [Vicini et al. 2021] (short as PRB), PRB with a multi-scale scheme (short as PRB.mul.res), and Plateau-reduced Differentiable Path Tracing [Fischer and Ritschel 2022] (short as PRDPT) fail to correctly recover the scene parameters. In contrast, our method successfully recovers the positions of all specimen objects due to the effectiveness of extended path space manifolds. Our hybrid optimization scheme (ours+hybrid) is able to further refine the results. We also show the close-up views of the color derivatives of PRB and our geometric derivatives with respect to a single scene parameter, respectively. Note that the two derivatives are inherently of different types — they are not computing the same quantities. The optimization is performed using a rendering resolution of 128×128 and 32 spps, while the images displayed in the figure are re-rendered in a higher resolution of 512×512 and 8192 spps. The scene is modified from [Bitterli 2016].

ABSTRACT

Physically based differentiable rendering has become an increasingly important topic in recent years. A common pipeline computes local color derivatives of light paths or pixels with respect to arbitrary scene parameters, and enables optimizing or recovering the scene parameters through iterative gradient descent by minimizing the difference between rendered and target images. However, existing approaches cannot robustly handle complex illumination effects including reflections, refractions, caustics, shadows, and highlights, especially when the initial and target locations of such illumination effects are not close to each other in the image space.

To address this problem, we propose a novel data structure named *extended path space manifolds*. The manifolds are defined in the combined space of path vertices and scene parameters. By enforcing geometric constraints, the path vertices could be implicitly and uniquely determined by perturbed scene parameters. This enables the manifold to track specific illumination effects and the corresponding paths, i.e., specular paths will still be specular paths after scene parameters are perturbed. Besides, the path derivatives

*Kun Xu is the corresponding author.

SA Conference Papers '23, December 12–15, 2023, Sydney, NSW, Australia

ACM ISBN 979-8-4007-0315-7/23/12.
https://doi.org/10.1145/3610548.3618195

with respect to scene parameters could be computed by solving small linear systems.

We further propose a physically based differentiable rendering method built upon the theoretical results of extended path space manifolds. By incorporating the path derivatives computed from the manifolds and an optimal transport based loss function, our method is demonstrated to be more effective and robust than state-of-the-art approaches in inverse rendering applications involving complex illumination effects.

CCS CONCEPTS

• **Computing methodologies → Ray tracing**.

KEYWORDS

differentiable rendering, extended path space manifolds

ACM Reference Format:

Jiankai Xing, Xuejun Hu, Fujun Luan, Ling-Qi Yan, and Kun Xu. 2023. Extended Path Space Manifolds for Physically Based Differentiable Rendering. In *SIGGRAPH Asia 2023 Conference Papers (SA Conference Papers '23), December 12–15, 2023, Sydney, NSW, Australia.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3610548.3618195

1 INTRODUCTION

Physically based *forward rendering* has been a central research topic in computer graphics for decades, with a focus on robust and accurate light transport simulation in virtual scenes with specified shapes, illumination and materials. Recently, significant progress has been made on its inverse problem with advances in theoretical frameworks, sampling algorithms and system pipelines of *differentiable rendering*, which offers the capability of evaluating derivatives of the rendered image with respect to arbitrary scene parameters and being used to facilitate gradient-based optimization problems [Bangaru et al. 2020; Li et al. 2018; Loubet et al. 2019; Nimier-David et al. 2019; Vicini et al. 2021, 2022; Zeltner et al. 2021; Zhang et al. 2020].

Physically based differentiable rendering pipelines generally calculate the local color derivatives of light paths or pixels with respect to arbitrary scene parameters. These computations facilitate the optimization or reconstruction of these arbitrary scene parameters through the iterative process of gradient descent which minimizes the discrepancy between the rendered image and the target image. However, current methods often struggle with robustly handling complex illumination effects including reflections, refractions, caustics, shadows, and highlights, especially when the initial and target locations of such illumination effects are not close to each other in the image space.

On the other hand, significant efforts have been dedicated to exploring methods for efficiently rendering these complex light transport effects within the scope of forward rendering, especially the methods involving the exploration of *path space manifolds* (PSMs) [Jakob and Marschner 2012; Jakob 2013; Kaplanyan et al. 2014; Veach and Guibas 1997; Zeltner et al. 2020]. In this work, drawing inspiration from these approaches, we bridge the gap by introducing the mathematical formulation of *extended path space manifolds* (EPSMs) in the context of physically based differentiable rendering, which are defined in the combined space of path vertices and scene parameters.

Concretely, our contributions include:

- We present extended path space manifolds. By enforcing geometric constraints, the manifolds implicitly define a mapping from scene parameters to path vertices. We further derive path derivatives that measure how path vertices change with respect to scene parameters under constraints.
- Built upon the extended path space manifolds, we present a physically based differentiable rendering method. Experiments show it is more effective and robust than state-of-the-art methods in inverse rendering applications involving complex illumination effects, including specular and glossy reflections, refractions, highlights, caustics, and shadows. Such an example is given in Fig. 1.

2 RELATED WORK

2.1 Path Space Manifolds

Physically based light transport algorithms are built on top of the path integral formulation [Veach 1998] using (Markov chain) Monte Carlo techniques. Veach and Guibas [1997] partitions the path space into submanifolds and designs several light path perturbation strategies for efficient sampling of difficult light paths, such as lens perturbation, caustic perturbation and multi-chain mutations.

Manifold exploration [Jakob and Marschner 2012; Jakob 2013] further improves the efficiency of MLT algorithms with manifold walks on the path space manifolds (PSMs), addressing the challenges posed by specular and near-specular light paths, which often lead to slow convergence rates in traditional light transport simulations. To navigate these complexities, manifold exploration leverages the inherent structure of these paths as manifolds in path space. A simple equation-solving iteration allows for efficient exploration on these path manifolds, resulting in an effective method to perturb specular paths using available geometric constraints in the path tracer. Kaplanyan et al. [2014] proposed to mutate paths by explicitly modeling the ray differentials [Igehy 1999] in half vector space manifold, yielding better rendering performance on glossy scenes. Zeltner et al. [2020] further improved the specular manifold constraints for rendering high-frequency caustics and glints.

Unlike aforementioned PSMs that are primarily tailored for forward rendering, we present extended path space manifolds (EPSMs) for physically based differentiable rendering, which differ mainly in two ways — First, PSMs are defined on the space of paths, while our EPSMs are defined on the combined space of paths and optimizable scene parameters. Second, PSMs compute the derivatives of paths with respect to the positions of endpoints, while our EPSMs compute the derivatives of paths with respect to scene parameters. Similar to PSMs where we could implicitly determine the positions of all specular vertices from the positions of diffuse endpoints, in EPSMs, we could also uniquely find the updated positions of all path vertices when scene parameters are slightly changed.

2.2 Differentiable Rendering

Inverse rendering typically requires both the development of an advanced forward parametric model and the computation of its corresponding derivatives. This process is often approached via

analysis-by-synthesis techniques [Gkioulekas et al. 2013; Khungurn et al. 2015; Zhao et al. 2016]. Recently, there has been a surge in interest for fully-differentiable forward rendering techniques, known as *differentiable rendering*. It enables practical inverse rendering applications, such as object capture and material estimation [Cai et al. 2022; Deng et al. 2022; Gao et al. 2019; Guo et al. 2020; Luan et al. 2021; Lyu et al. 2020; Munkberg et al. 2021; Shi et al. 2020].

The pioneer general-purpose differentiable rendering frameworks, including OpenDR [Loper and Black 2014] and Neural 3D Mesh Renderer [Kato et al. 2018], leveraged analytical differentiation with approximate forward models. A variety of differentiable rasterization techniques have been developed to efficiently render primitives within a scene [Laine et al. 2020; Liu et al. 2019; Ravi et al. 2020]. Although capable of handling primary visibility, these frameworks encountered difficulties with complex illumination effects.

Physically based differentiable rendering has been focused on differentiating a path tracer to handle global illumination through light transport simulation [Bangaru et al. 2020; Li et al. 2018; Loubet et al. 2019; Vicini et al. 2021, 2022; Yan et al. 2022; Zeltner et al. 2021; Zhang et al. 2020, 2021]. Generally, physically based differentiable rendering involves estimating two main components: (i) the *interior* integrals derived from differentiating the integrands associated with the forward-rendering models, and (ii) the *boundary* integrals determined over the discontinuities present in those integrands. Previously, the estimation of interior integrals primarily leveraged path sampling methods originally conceived for forward rendering, while reparameterization techniques [Bangaru et al. 2020; Loubet et al. 2019] apply suitable changes of variables to the integrands to avoid computing boundary integrals. Recently, Zeltner et al. [2021] and Vicini et al. [2021] investigated how reparameterization techniques and different sampling strategies such as "attached sampling" and "detached sampling" influence the performance of Monte Carlo estimations. Some of the directional and positional derivatives computed in these attached sampling methods are indeed somewhat similar to our proposed manifold derivatives, while our method focuses on computing geometric derivatives (i.e., the change of path geometries w.r.t. scene parameters) instead of per-pixel color derivatives (i.e., how per-pixel color contributions change w.r.t. scene parameters). More precisely, contemporary differentiable rendering methods that compute per-pixel image derivatives often encounter limitations in inverse rendering tasks. In particular, they are ineffective in aiding global and long-range optimization (when initial and target objects/shadows/caustics are not close to each other, i.e., are not overlapping in the image space) during inverse rendering.

Recently, Xing et al. [2022] linked screen-space pixels to their corresponding visible 3D points and derived 5D RGBXY derivatives, assessing how color and screen position change with scene parameters. However, it is based on differentiable rasterization and cannot handle inverse rendering applications involving complex global illumination effects such as reflections and caustics. Fischer and Ritschel [2022] convolved the rendering function with a kernel that blurs the scene parameter space. The method could capture long-range relationships to some extent, however, its gradients have relatively high variance and will become less effective when the number of scene parameters grows larger.

3 BACKGROUND: PATH SPACE MANIFOLDS

Path space manifolds (short as PSMs), or specular manifolds, are first proposed by Jakob and Marschner [2012] to address the problem of rendering scenes with difficult specular light transport. A general length-n light path could be represented as $\overline{\mathbf{x}} = \mathbf{x}_0\mathbf{x}_1\cdots\mathbf{x}_n$ where the two endpoints are the positions of the eye $\mathbf{x}_0$ and a point $\mathbf{x}_n$ on a light source, and the middle bouncing vertices $\mathbf{x}_i$ $(1 \le i \le n-1)$ could be either diffuse or specular.

Without loss of generality, let's consider a simpler case of a path $\overline{\mathbf{x}} = \mathbf{x}_1\cdots\mathbf{x}_k$, and we assume that the two endpoints $\mathbf{x}_1$ and $\mathbf{x}_k$ are diffuse and all other vertices are specular. While the dimension of the path space is relatively large (i.e., $2k$), the path in fact lies in a lower dimensional subspace (i.e., 4). Since all middle specular vertices need to satisfy the law of reflection or Snell's law, to formulate it, Jakob and Marschner [2012] introduced a half-vector constraint function to each specular vertex $\mathbf{x}_i$, i.e., constraining the half-vector to be parallel to surface normal:

$$\mathbf{c}_i(\mathbf{x}_{i-1}, \mathbf{x}_i, \mathbf{x}_{i+1}) = T(\mathbf{x}_i)^T h(\overrightarrow{\mathbf{x}_i\mathbf{x}_{i-1}}, \overrightarrow{\mathbf{x}_i\mathbf{x}_{i+1}}) = \mathbf{0}, \tag{1}$$

where $T(\mathbf{x}_i)$ is a 2×3 matrix that represents the local tangent vectors, and the half vector function $h(\cdot)$ is defined as [Walter et al. 2007]:

$$h(\mathbf{i}, \mathbf{o}) = \frac{\mathbf{h}}{\|\mathbf{h}\|}, \text{ where } \mathbf{h} = \begin{cases} \mathbf{i}+\mathbf{o} & \text{if reflection,} \\ \eta_i\mathbf{i}+\eta_o\mathbf{o} & \text{if refraction.} \end{cases} \tag{2}$$

η_i and η_o denote the index of refraction of the two sides, respectively.

By putting together the half-vector constraints on all specular vertices, we could get a stacked constraint function $\mathbf{C} : \mathbb{R}^{2k} \to \mathbb{R}^{2(k-2)}$, expressed as:

$$\mathbf{C}(\overline{\mathbf{x}}) = [\mathbf{c}_2(\overline{\mathbf{x}}), \cdots, \mathbf{c}_{k-1}(\overline{\mathbf{x}})] = \mathbf{0}, \tag{3}$$

and the PSMs are defined as the set of paths that satisfy the constraints:

$$\{\overline{\mathbf{x}} \mid \mathbf{C}(\overline{\mathbf{x}}) = \mathbf{0}\} \tag{4}$$

The *Implicit Function Theorem* [Spivak 1965] tells that the whole path $\overline{\mathbf{x}}$ is a function of two endpoints $\mathbf{x}_1$ and $\mathbf{x}_k$ in a neighborhood of a current path. In other words, all middle specular vertices $\mathbf{x}_i$ $(2 \le i \le k-1)$ could be implicitly determined by the positions of the two diffuse endpoints. Besides, the partial derivatives of the path $\overline{\mathbf{x}}$ with respect to the two endpoints could be computed by solving a linear system derived from the Jacobian matrix of $\mathbf{C}$.

4 EXTENDED PATH SPACE MANIFOLDS

Inspired by existing works of PSMs, we present *extended path space manifolds* (EPSMs) in order to handle differentiable rendering problems involving difficult light paths. Currently, we focus on global illumination with surface interactions, while volumetric effects are left for future works.

4.1 Definition of EPSMs

Given a length-n light path $\overline{\mathbf{x}} = \mathbf{x}_0\mathbf{x}_1\cdots\mathbf{x}_n$ which starts from the position of the eye $\mathbf{x}_0$, follows with multiple diffuse or specular bounces inside the scene, and ends at a point $\mathbf{x}_n$ on a light source, we define an *extended path* by associating it with scene parameters of interest $\theta = [\theta_1, \cdots, \theta_m]$, simply as a combined vector of path

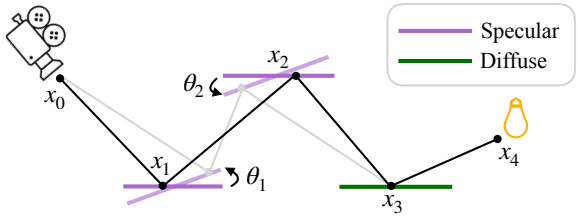


Figure 2: A motivating example of extended path space manifolds.

vertices and scene parameters: $(\overline{\mathbf{x}}, \theta)$. So that the *extended path space* is an Euclidean space (i.e., $\mathbb{R}^{2(n+1)+m}$) defined as:

$$\{(\overline{\mathbf{x}}, \theta) \mid \overline{\mathbf{x}} \in \mathbb{R}^{2(n+1)}, \theta \in \mathbb{R}^m\}. \tag{5}$$

Analogous to PSMs, we also introduce constraints to the extended path space so that its actual dimension could be reduced. Specifically, we always introduce $n+1$ 2D vector valued constraint functions, i.e., the same number as path vertices. The constraint functions could be stacked as $\mathbf{C} : \mathbb{R}^{2(n+1)+m} \rightarrow \mathbb{R}^{2(n+1)}$, in the form:

$$\mathbf{C}(\overline{\mathbf{x}}, \theta) = [\mathbf{c}_1(\overline{\mathbf{x}}, \theta), \cdots, \mathbf{c}_{n+1}(\overline{\mathbf{x}}, \theta)] = \mathbf{0}, \tag{6}$$

where $\mathbf{c}_i$ denotes the i-th constraint function ($1 \leq i \leq n+1$). Therefore, the *extended path space manifolds* (EPSMs) could be defined as the set of extended paths satisfying the above constraints:

$$\{(\overline{\mathbf{x}}, \theta) \mid \mathbf{C}(\overline{\mathbf{x}}, \theta) = \mathbf{0}\} \tag{7}$$

The EPSMs essentially give a mapping from scene parameters $\theta \in \mathbb{R}^m$ to the path $\overline{\mathbf{x}} \in \mathbb{R}^{2(n+1)}$, i.e., the positions of all path vertices could be implicitly determined by scene parameters θ. The *Implicit Function Theorem* ensures that the mapping exists in the neighborhood of a current extended path.

A motivating example of EPSMs is given in Fig 2. It shows an extended path $(\overline{\mathbf{x}}, \theta)$, where $\overline{\mathbf{x}} = \mathbf{x}_0\mathbf{x}_1\mathbf{x}_2\mathbf{x}_3\mathbf{x}_4$ starts from the eye, then sequentially bounces at two mirrors and a diffuse surface, and ends at a light source. The associated scene parameters $\theta = [\theta_1, \theta_2]$ indicate the rotating angles of the two mirrors respectively. *Half-vector constraints* are enforced on the specular vertices and *fixed position constraints* are enforced on other vertices (will be explained in Sec. 4.2). As shown in the figure, if we slightly rotate the mirrors, the path will be perturbed and uniquely determined, i.e., $\mathbf{x}_1$ and $\mathbf{x}_2$ will be moved accordingly to satisfy the law of reflection, and $\mathbf{x}_3$ will keep fixed since it is a diffuse vertex.

4.2 Constraint Functions

As mentioned in Sec. 4.1, in order to define the EPSMs, we need to enforce $n+1$ constraint functions on the extended paths. Below, we introduce the 4 types of constraint functions we have used in EPSMs. Note that all constraint functions are 2D vector valued.

4.2.1 Half-vector constraint. It constrains the half-vector of incoming and outgoing ray directions in the local frame of a specific vertex $\mathbf{x}_i$ to be unchanged when scene parameter changes:

$$T(\mathbf{x}_i, \theta)^T h(\overrightarrow{\mathbf{x}_i\mathbf{x}_{i-1}}, \overrightarrow{\mathbf{x}_i\mathbf{x}_{i+1}}, \theta) = \mathbf{const}, \tag{8}$$

where $T(\cdot)$ is a 2×3 matrix representing the local tangent vectors, and $h(\cdot)$ is the half vector. Different from the one used in Jakob and Marschner [2012] which strictly constrains the half-vector to be parallel to surface normal, we constraint the half-vector in the local frame to be fixed during scene parameter perturbation. This makes the constraint function applicable to both specular and glossy surfaces.

4.2.2 Fixed position constraint. This constraint enforces the position of a path vertex $\mathbf{x}_i$ to be locally unchanged. Specifically, we constrain the barycentric weights of a vertex $\mathbf{x}_i$ with respect to its belonging surface triangle to be unchanged:

$$w(\mathbf{x}_i, \mathbf{v}_1(\theta), \mathbf{v}_2(\theta), \mathbf{v}_3(\theta)) = \mathbf{const}, \tag{9}$$

where $\mathbf{v}_\mathbf{k}(\theta)$ ($1 \leq k \leq 3$) denotes the three vertices of the surface triangle, and $w(\mathbf{x}, \mathbf{a}, \mathbf{b}, \mathbf{c})$ computes the barycentric coordinates of point $\mathbf{x}$ with respect to a triangle $\triangle\mathbf{abc}$. The fixed position constraint is usually applied to diffuse vertices and the two endpoints, i.e., the eye point and the point on the light source.

4.2.3 Fixed direction constraint. This constraint enforces a ray direction $\overrightarrow{\mathbf{x}_i\mathbf{x}_{i-1}}$ to be locally unchanged in the local frame of its neighboring vertex $\mathbf{x}_i$:

$$T(\mathbf{x}_i, \theta)^T \cdot \overrightarrow{\mathbf{x}_i\mathbf{x}_{i-1}} = \mathbf{const}. \tag{10}$$

It is usually used to constrain the outgoing ray from the light source (i.e., $\overrightarrow{\mathbf{x}_n\mathbf{x}_{n-1}}$) when dealing with caustic effects.

4.2.4 Colinear constraint. The colinear direction constraint is used to constrain two neighboring ray directions of a path vertex $\mathbf{x}_i$ to be always colinear:

$$\overrightarrow{\mathbf{x}_{i-1}\mathbf{x}_i} \times \overrightarrow{\mathbf{x}_i\mathbf{x}_{i+1}} = \mathbf{0}, \tag{11}$$

where $\times$ denotes the cross product operator. The colinear constraint is only used for shadow rays (will be explained in Sec. 4.3.3).

4.3 Construction of EPSMs

EPSMs are built by enforcing constraints to the space of extended paths $(\overline{\mathbf{x}}, \theta)$. To effectively handle different rendering effects, we have designed 3 different types of EPSMs, as illustrated in Fig. 4. Different EPSMs use different combinations of constraint functions, but the number of applied constraint functions is always $n+1$, the same as the number of path vertices. This ensures that the positions of all path vertices $\overline{\mathbf{x}} = \mathbf{x}_0\mathbf{x}_1 \cdots \mathbf{x}_n \in \mathbb{R}^{2(n+1)}$ could be implicitly determined from scene parameters $\theta = [\theta_1, \cdots, \theta_m] \in \mathbb{R}^m$. Below, we explain how each type of EPSM is constructed.

4.3.1 General EPSMs. We apply the following constraints to an extended path $(\overline{\mathbf{x}} = \mathbf{x}_0\mathbf{x}_1 \cdots \mathbf{x}_n, \theta)$ to construct a general EPSM:

(1) enforcing the *half-vector constraint* (Sec. 4.2.1) on specular vertices;
(2) enforcing the *fixed position constraint* (Sec. 4.2.2) on all diffuse vertices and the two endpoints $\mathbf{x}_0$ and $\mathbf{x}_n$.

The general EPSM uses a combination of constraint functions similar to Jakob and Marschner [2012]. It is the most general type of EPSM and could be used in handling a wide range of rendering effects including reflections, refractions, and highlights. Note that it could also handle glossy or semi-glossy surfaces, diffuse indirect illumination and direct lighting as well.

4.3.2 Caustic EPSMs. Caustic EPSMs are specifically designed for handling caustic effects. We construct caustic EPSMs only for caustic light paths in the form of $\mathbf{ES^*DS^+L}$ [Heckbert 1990], by applying the following constraints (as illustrated in Fig. 4 (b)):

(1) enforcing the *fixed position constraint* (Sec. 4.2.2) on the two endpoints $\mathbf{x}_0$ and $\mathbf{x}_n$;
(2) enforcing the *fixed direction constraint* (Sec. 4.2.3) on the outgoing ray from the light source $\overrightarrow{\mathbf{x}_n\mathbf{x}_{n-1}}$;
(3) enforcing the *half-vector constraint* (Sec. 4.2.1) on all $n-2$ specular vertices except the diffuse vertex.

Compared to general EPSMs, caustic EPSMs additionally enforce the fixed direction constraint on the light ray while removing the fixed position constraint on the diffuse vertex $\mathbf{x}_1$. This enables a caustic EPSM to effectively track the caustic pattern cast on diffuse surfaces, which will move accordingly when light sources or specular objects on the path move.

4.3.3 Shadow EPSMs. Shadow EPSMs are specifically designed for handling shadows in direct illumination. In forward rendering, we will trace shadow rays from shading points towards light sources. If the shadow ray hits an occluder before arriving at the light source, we consider that the shading point is inside shadow. As shown in Fig. 4 (c), we construct a shadow EPSM for each shadow ray that hits an occluder. Since it only considers direct illumination, the path is short and consists of only 4 vertices (i.e., the eye point $\mathbf{x}_0$, shading vertex $\mathbf{x}_1$, the occluder vertex $\mathbf{x}_2$, and the light source point $\mathbf{x}_3$). Note that we only record the first hit point on the shadow ray as the occluder vertex. We apply the following constraints to the shadow EPSMs:

(1) enforcing the *fixed position constraint* (Sec. 4.2.2) on the two endpoints $\mathbf{x}_0$ and $\mathbf{x}_3$ and the occluder vertex $\mathbf{x}_2$;
(2) enforcing the *colinear constraint* (Sec. 4.2.4) on rays $\overrightarrow{\mathbf{x}_1\mathbf{x}_2}$ and $\overrightarrow{\mathbf{x}_2\mathbf{x}_3}$ since the shadow ray ($\mathbf{x}_1 \rightarrow \mathbf{x}_2 \rightarrow \mathbf{x}_3$) needs to be a straight line by definition.

Note that no constraints are applied to the shading vertex $\mathbf{x}_1$ so that the shadow EPSMs could effectively capture moving shadows.

4.4 Computation of Path Derivatives

In this subsection, we show how to compute the path derivatives, i.e., how path geometries $\overline{\mathbf{x}}$ change with respect to scene parameters θ under the constraints of an EPSM:

$$\frac{\partial \overline{\mathbf{x}}}{\partial \theta} = \left[\frac{\partial \mathbf{x}_0}{\partial \theta}, \frac{\partial \mathbf{x}_1}{\partial \theta}, \cdots, \frac{\partial \mathbf{x}_n}{\partial \theta} \right]. \tag{12}$$

The *Implicit Function Theorem* [Spivak 1965] guarantees the existence of the derivatives in the neighborhood of a current extended path $(\overline{\mathbf{x}}, \theta)$. The derivatives could be computed through *implicit partial differentiation* of the stacked constraint function $\mathbf{C}(\overline{\mathbf{x}}, \theta)$ in Eq. 6:

$$\frac{\mathrm{d}\mathbf{C}(\overline{\mathbf{x}}, \theta)}{\mathrm{d}\theta} = \frac{\partial \mathbf{C}}{\partial \theta} + \frac{\partial \mathbf{C}}{\partial \overline{\mathbf{x}}} \cdot \frac{\partial \overline{\mathbf{x}}}{\partial \theta} = \mathbf{0}. \tag{13}$$

Hence, the path derivatives $\partial\overline{\mathbf{x}}/\partial\theta$ could be obtained through solving the following linear system:

$$\frac{\partial \overline{\mathbf{x}}}{\partial \theta} = -\left(\frac{\partial \mathbf{C}}{\partial \overline{\mathbf{x}}} \right)^{-1} \cdot \frac{\partial \mathbf{C}}{\partial \theta}, \tag{14}$$

where $\partial\mathbf{C}/\partial\overline{\mathbf{x}}$ is a $(2n+2)\times(2n+2)$ matrix, and $\partial\mathbf{C}/\partial\theta$ is a $(2n+2)\times m$ matrix. The two matrices are actually two sub-matrices of the sparse Jacobian matrix $\nabla\mathbf{C}$.

5 OUR DIFFERENTIABLE RENDERING METHOD

In this section, we introduce our physically based differentiable rendering method based on the theoretical results of the EPSMs.

5.1 Method Overview

5.1.1 Motivation. Our goal is to address the difficult problem of effectively and robustly handling complex illumination effects in the context of physically based differentiable rendering, such as recovering the position and orientation of a mirror from its reflected image, optimizing the location of a light source through optimizing the shape of the caustics, or recovering the positions of occluder objects by its shadows. While existing physically based differentiable rendering methods (such as Mitsuba 2 [Nimier-David et al. 2019]) support such effects, they usually require that the initial and target images are already well-aligned. In particular, they are ineffective in handling global and long-range optimization, i.e., when initial and target objects/shadows/caustics are not close to each other in the image space, since the per-pixel color derivatives are intrinsically local and sparse.

In contrast, our method could handle such long-range optimization in a more robust way. First, our proposed EPSMs are rather suitable for tracking those illumination effects. For example, considering a specular path (i.e. $\mathbf{ES^+DL}$) causing a reflection, the updated path implicitly determined by perturbed scene parameters will still be a specular path. Second, different from previous works which consider color derivatives at fixed pixel locations, our manifold derivatives are essentially *geometric derivatives*, i.e., computing how path geometries change with respect to scene parameters, which are denser and potentially lead to more stable optimization (see visualization in Fig. 1). Finally, our employed optimal transport could help find long-range matching and could be directly connected and combined with our manifold derivatives.

5.1.2 Method pipeline. Let's consider a typical setting of inverse rendering: given initial scene parameters and target image(s) from one or multiple viewpoints, we aim to recover desired scene parameters through iterative gradient descent by minimizing a predefined loss function between the rendered and target images. At each iteration, we perform the following steps:

(1) Forward rendering. We use Monte Carlo path tracing for rendering images and we record all sampled light paths.
(2) Construction of EPSMs. For each sampled light path, we associate it with scene parameters of interest to obtain an extended path and build an EPSM for it.
(3) Path-pixel matching. We utilize optimal transport to obtain a pixel-to-pixel mapping between the rendered images and the target reference images. For each pair of matched pixels, we record all sampled paths in the rendered pixel as corresponding to the target pixel.
(4) Optimization of scene parameters. We define a loss function according to the path-pixel correspondences, then perform

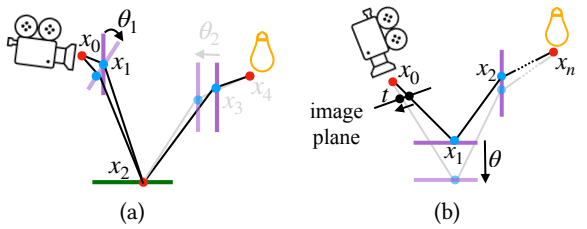


Figure 3: Illustrations. (a) simplification of a general EPSM; (b) loss function.

backpropagation to minimize the loss function using the path derivatives computed from EPSMs.

We will further describe the details in Sec. 5.2.

5.2 Method Details

5.2.1 Forward rendering. We use Monte Carlo path tracing for forward rendering to produce rendered images. We record part of sampled light paths and also record shadow paths in direct illumination. The recorded paths will be used to build EPSMs and the rendered images will be used to find path-pixel matching.

5.2.2 Construction of EPSMs. For each recorded path, we associate it with scene parameters of interest to obtain an extended path and then construct an EPSM of a specific type for the extended path. Specifically, we would like to construct *shadow EPSMs* and *caustic EPSMs* for shadow paths and for caustic paths, respectively, and construct *general EPSMs* for other types of paths. While it is easy to automatically determine shadow paths, it is non-trivial to automatically distinguish caustic paths (i.e., caustic paths must be in the form of $\mathbf{ES^*DS^+L}$, but $\mathbf{ES^*DS^+L}$-paths are not necessarily caustic paths). Hence, we manually specify whether or not the rendered scene is a caustic scene. For non-caustic scenes, we construct general EPSMs for all paths except shadow paths. For caustic scenes, we construct caustic EPSMs for $\mathbf{ES^*DS^+L}$-paths and general EPSMs for other paths.

Note that the number of associated scene parameters could be different for different paths. Let's imagine a scene with two mirrors where the optimizable scene parameters are their rotating angles. Paths that only hit one mirror just need to be associated with a single scene parameter (i.e., the rotating angle of the hit mirror). Paths that hit neither of two mirrors could be directly discarded and will not be used in constructing EPSMs.

Furthermore, the general EPSMs could be potentially simplified. Since our loss function (Eq. 16, will be explained later) only depends on the first two vertices ($\mathbf{x}_0$ and $\mathbf{x}_1$) of a path and the fixed position constraint applied to diffuse vertices cuts the relationship between vertices before and after a diffuse vertex, we could simplify a general EPSM by removing the path vertices after the first diffuse vertex. Fig. 3 (a) shows such an example: the general EPSM $(\mathbf{x}_0\mathbf{x}_1\mathbf{x}_2\mathbf{x}_3\mathbf{x}_4, [\theta_1, \theta_2])$ could be simplified to $(\mathbf{x}_0\mathbf{x}_1\mathbf{x}_2, [\theta_1])$ since the vertex $\mathbf{x}_2$ is diffuse.

5.2.3 Path-pixel matching. In this step, first, we follow the work of Xing et al. [2022] to use optimal transport to obtain a one-to-one pixel matching between rendered and target images. We choose optimal transport since it could capture long-range relationships. We define the unit transportation loss between a pixel $\mathbf{p_i}$ in the rendered image and a pixel $\mathbf{t_j}$ in the target image as follows:

$$\lambda_1 \left(\mathbf{I_r}(\mathbf{p_i}) - \mathbf{I_t}(\mathbf{t_j})\right)^2 + \lambda_2 \left(\mathbf{p_i} - \mathbf{t_j}\right)^2, \tag{15}$$

where $\mathbf{I_r}(\cdot), \mathbf{I_t}(\cdot)$ denote pixel color values of the rendered image and the target image, respectively, and the balancing weights are set as $\lambda_1 = \lambda_2 = 0.5$. Note that the unit transportation loss considers both color and positional differences between pixels.

The optimal transport algorithm will find the optimal one-to-one pixel matching with the minimal sum of unit transportation losses between matched pixels. To achieve a trade-off between accuracy and efficiency, we also follow Xing et al. [2022] to use the Sinkhorn divergences [Cuturi 2013; Feydy et al. 2019] with parameter $\epsilon = 0.01$ to compute an approximated optimal transport.

After that, we build correspondences between sampled paths and pixels in the target image. For each pair of matched pixels, i.e. a pixel $\mathbf{p_i}$ in the rendered image and a pixel $\mathbf{t_j}$ in the target image, we simply set all sampled paths at pixel $\mathbf{p_i}$ as corresponding to pixel $\mathbf{t_j}$.

5.2.4 Loss function and optimization. As shown in Fig. 3 (b), given path-pixel correspondences, in order to drive a path to move towards the corresponding pixel location, we define the loss function L of each path $\overline{\mathbf{x}}$ as follows:

$$L(\theta) = (P(\overline{\mathbf{x}}, \theta) - \mathbf{t})^2 = (P(\mathbf{x_0}, \mathbf{x_1}, \theta) - \mathbf{t})^2, \tag{16}$$

where $P(\mathbf{x_0}, \mathbf{x_1}, \theta)$ denotes the ray-plane intersection point of ray $\overrightarrow{\mathbf{x_0}\mathbf{x_1}}$ and the image plane, $\mathbf{t}$ denotes the corresponding pixel position in the target image. The total loss is simply the sum of the losses in Eq. 16 over all paths. The formulation of our loss is similar to the refraction loss in [Lyu et al. 2020].

The derivative of the loss function L with respect to scene parameters θ could be simply obtained through the chain rule:

$$\frac{\mathrm{d}L}{\mathrm{d}\theta} = \left(\frac{\partial L}{\partial \overline{\mathbf{x}}}\right) \cdot \left(\frac{\partial \overline{\mathbf{x}}}{\partial \theta}\right)^T + \frac{\partial L}{\partial \theta}, \tag{17}$$

where $\partial\overline{\mathbf{x}}/\partial\theta$ is the path derivative with respect to scene parameters under the constraints of an EPSM, which is computed by solving the small linear system in Eq. 14.

Furthermore, since the loss function (Eq. 16) only depends on the first two vertices ($\mathbf{x}_0$ and $\mathbf{x}_1$) of the path, we only need to compute the derivatives involving the first two vertices:

$$\frac{\mathrm{d}L}{\mathrm{d}\theta} = \left[\frac{\partial L}{\partial \mathbf{x}_0}, \frac{\partial L}{\partial \mathbf{x}_1}\right] \cdot \left[\frac{\partial \mathbf{x}_0}{\partial \theta}, \frac{\partial \mathbf{x}_1}{\partial \theta}\right]^T + \frac{\partial L}{\partial \theta}. \tag{18}$$

By using the above loss derivatives, the scene parameters θ could be optimized through iterative backpropagation.

5.2.5 Implementation details. We implement our method entirely on GPU using Mitsuba 3 [Jakob et al. 2022b,a] and PyTorch [Paszke et al. 2019]. Thanks to the interoperability of the two frameworks, mixed computations, data transmission, and data synchronization between them can be easily achieved. Specifically, forward rendering and backpropagation are implemented using Mitsuba 3. Optimal transport based matching, solving linear systems for path derivatives (Eq. 14) are implemented using PyTorch. The Jacobian matrix of the constraint functions $\nabla\mathbf{C}$ are computed through auto-differentiation with a mixed use of PyTorch and Dr.Jit [Jakob et al.

2022a]. Taking the fixed position constraint (Eq. 9) as an example, the derivatives of the barycentric weights with respect to vertex positions $\partial w/\partial \mathbf{v}_k$ $(1 \leq k \leq 3)$ are computed in PyTorch, while the derivatives of vertex positions with respect to scene parameters $\partial \mathbf{v}_k/\partial \theta$ $(1 \leq k \leq 3)$ are computed in Dr.Jit. They are combined to obtain $\partial w/\partial \theta$ through the chain rule.

6 EXPERIMENTS

All experiments are performed on a PC with an NVIDIA RTX 3090 GPU (24G memory). In our experiments, the paths are recorded using a rendering resolution of 128×128 and 32 samples per pixel (spps). For better quality of matching, we render images with a resolution 512×512 and 64 spps and downsample the rendered images to 128×128 to perform optimal transport. For all examples, we run our EPSM based method for 500 iterations. Typically, one iteration takes about 4.3 - 7.2 seconds. The main bottleneck lies in the computation of the path derivatives which costs about 70% time budget, since it requires solving a small linear system for each path. We use the Adam optimizer [Kingma and Ba 2014] for backpropagation.

6.1 Scene Configurations

In order to demonstrate the robustness and effectiveness of our method, we test our method on inverse rendering applications over four representative scenes. The tested scenes cover various types of complex illumination effects including reflections, refractions, caustics, shadows, and glossy highlights.

The Bathroom scene in Fig. 1 contains eight specimen objects inside a glass box, which are viewed through reflection paths via the mirror followed by refraction paths via the glass. We aim at recovering the 2D translation vectors of the specimen objects through tracking the reflected (and refracted) images.

The Shadow scene in Fig. 5 (a) contains an area light, a floor, and 400 spheres. The light is put above the spheres and casts shadows onto the floor. The camera is put below the spheres and towards the floor so that the camera can see shadows. The goal is to recover the 2D translation vectors of all occluder spheres by their shadows.

The CornellBox scene in Fig. 5 (b) presents a challenging mix of intricate and colorful caustics originating from a glass ball, testing the robustness of caustics path derivatives estimation with respect to the rotation angles of six area lights.

The Highlight scene in Fig. 5 (c) serves to assess the ability to differentiate through glossy and near-specular light paths. Starting with five parallel glossy planes reflecting vibrant, out-of-view emitters, we would like to simultaneously optimize the plane rotations and horizontal translation offsets of the emitters.

6.2 Results and Analysis

6.2.1 Visual comparisons. We compare the results of our method with three baselines, including *Path Replay Backpropagation* [Vicini et al. 2021] with reparameterization [Bangaru et al. 2020] (short as PRB), PRB with a multi-scale scheme (short as PRB.mul.res) and *Plateau-reduced Differentiable Path Tracing* [Fischer and Ritschel 2022] (short as PRDPT). In the multi-scale scheme (PRB.mul.res), we always render images with a resolution of 512×512, while using progressively downsampled rendered images with a resolution from 8×8 to 512×512 for computing derivatives. For each scene, we run each method for 500 iterations. The results are given in Fig. 1 and Fig. 5.

However, neither PRB nor PRDPT converges to the correct results on all these scenes. While PRB is effective in optimizing various types of scene parameters under complex illumination, it usually requires that the initial rendered image is sufficiently aligned with the target image since it relies on local color derivatives, which limits PRB in handling rendering effects with long-range relationships. The multi-scale scheme could alleviate this problem (see Fig. 5 (a), the 4th column), but still cannot solve it robustly. While PRDPT is able to capture long-range relationships to some extent, however, due to the increase of sampling variance in a higher dimensional space, it will quickly become less effective when the number of optimizable parameters grows larger.

In contrast, our method successfully produces nice convergence to the target images on all the scenes. The results demonstrate the robustness of our method in handling a range of complex illumination effects and the superior effectiveness of EPSM-based derivatives over baseline approaches.

6.2.2 Hybrid optimization scheme. Noticing that in scenes Bathroom (Fig. 1) and Shadow (Fig. 5 (a)), our optimized results are close to the target images but still have some subtle differences. This is because the approximated optimal transport using Sinkhorn divergences may produce inaccurate pixel matching when the rendered and target images are already aligned well [Xing et al. 2022], which may lead to less accurate loss derivatives and lower convergence rate in the last iterations of optimization. To address this issue, similar to [Xing et al. 2022], we could optionally employ a hybrid optimization scheme to refine the results. This is done by simply running the optimization for 100 iterations using PRB after running our method for 500 iterations. As shown in Fig. 1 and Fig. 5 (a), the hybrid scheme generates better aligned results.

6.2.3 Error curves. Fig. 6 offers the error curves from the four inverse rendering experiments discussed above. For the Bathroom, CornellBox, and Highlight scenes, we opt for parameter loss, as the target parameters are available. For the Shadow scene, featuring 400 spheres, we compute the image RMSE loss between the rendered image and the target image. Our method converges quickly on all the scenes.

6.2.4 Visualization of derivatives. In Fig. 1 (the middle column), We visualize the derivatives of PRB [Vicini et al. 2021] and our derivatives with respect to a single scene parameter (i.e., the vertical movement of one specimen object), respectively. The derivatives of PRB compute how pixel colors change with scene parameters, which are sparse and only have non-zero values near object boundary. In contrast, we compute a different type of derivatives — how path geometries change with scene parameters. Our derivatives are dense inside the object and could lead to more stable optimization.

6.2.5 Additional results. Fig. 7 showcases more inverse rendering results. The tasks incorporate multiple types of optimizing goals — light translation vector through caustics, object translation through nested reflections, camera pose estimation, translation vectors of three light sources via glossy highlights, normal map of a refractive glass slab via the refracted image, and a 72-parameter human model

(Skinned Multi-Person Linear, SMPL) [Loper et al. 2015] from its curved shadow. These tasks further demonstrate the robustness of our method in handling various scenes involving complex illumination effects and different types of optimizable scene parameters.

We will show the progressive optimization process for all examples in the supplemental video.

7 CONCLUSION

In this paper, we introduced a novel approach to physically based differentiable rendering. Our key contribution lies in the formulation of extended path space manifolds (EPSMs), designed to navigate the complex non-local and long-range relationships that often characterize intricate illumination effects in a scene. By enforcing geometric constraints of EPSMs that implicitly enable a mapping from scene parameters to path vertices, our approach significantly enhances the robustness and efficacy of the optimization processes involved in differentiable rendering. Through various experiments, our method demonstrated marked improvements over existing techniques, particularly in handling complex illumination effects.

While our method shows promise in inverse rendering applications, it still faces limitations in computation time and memory consumption. The timing bottleneck lies in the computation of path derivatives which requires solving small linear systems. Currently we use PyTorch routine *torch.linalg.inv*. Since the Jacobian matrix $\nabla \mathbf{C}$ is sparse, a possible way for acceleration would be using iterative methods instead of direct methods to solve linear systems. As for the usage of GPU memory, currently we store all sampled light paths for the simplicity of implementation. In the future, we could switch to a batch computation mode and only store light paths in a batch to reduce memory consumption.

Furthermore, our method cannot handle "shadow in mirror" effect since it does not fit well with any EPSMs we have defined. To handle it, we need to extend the definition of shadow EPSMs, where the shadow path could contain zero or more specular vertices between the eye point and the shading point. Finally, the effectiveness of our method depends heavily on the quality of matching. More sophisticated matching algorithms beyond optimal transport are also worth investigating.

ACKNOWLEDGMENTS

We would like to thank all reviewers for their helpful comments. This work is supported by the National Natural Science Foundation of China (Project Number: 61932003). Ling-Qi Yan is supported by gift funds from Adobe, Dimension 5 and XVerse.

REFERENCES

Sai Praveen Bangaru, Tzu-Mao Li, and Frédo Durand. 2020. Unbiased warped-area sampling for differentiable rendering. *ACM Transactions on Graphics (TOG)* 39, 6 (2020), 1–18.

Benedikt Bitterli. 2016. Rendering resources. https://benedikt-bitterli.me/resources/.

Guangyan Cai, Kai Yan, Zhao Dong, Ioannis Gkioulekas, and Shuang Zhao. 2022. Physics-Based Inverse Rendering using Combined Implicit and Explicit Geometries. *arXiv preprint arXiv:2205.01242* (2022).

Marco Cuturi. 2013. Sinkhorn distances: Lightspeed computation of optimal transport. *Advances in neural information processing systems* 26 (2013).

Xi Deng, Fujun Luan, Bruce Walter, Kavita Bala, and Steve Marschner. 2022. Reconstructing Translucent Objects using Differentiable Rendering. In *Special Interest Group on Computer Graphics and Interactive Techniques Conference Proceedings*. 1–10.

Jean Feydy, Thibault Séjourné, François-Xavier Vialard, Shun-ichi Amari, Alain Trouve, and Gabriel Peyré. 2019. Interpolating between Optimal Transport and MMD using Sinkhorn Divergences. In *The 22nd International Conference on Artificial Intelligence and Statistics*. 2681–2690.

Michael Fischer and Tobias Ritschel. 2022. Plateau-reduced Differentiable Path Tracing. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2022).

Duan Gao, Xiao Li, Yue Dong, Pieter Peers, Kun Xu, and Xin Tong. 2019. Deep inverse rendering for high-resolution SVBRDF estimation from an arbitrary number of images. *ACM Trans. Graph.* 38, 4 (2019), 134–1.

Ioannis Gkioulekas, Shuang Zhao, Kavita Bala, Todd Zickler, and Anat Levin. 2013. Inverse volume rendering with material dictionaries. *ACM Transactions on Graphics (TOG)* 32, 6 (2013), 1–13.

Yu Guo, Cameron Smith, Miloš Hašan, Kalyan Sunkavalli, and Shuang Zhao. 2020. MaterialGAN: reflectance capture using a generative SVBRDF model. *arXiv preprint arXiv:2010.00114* (2020).

Paul S. Heckbert. 1990. Adaptive Radiosity Textures for Bidirectional Ray Tracing. In *Proceedings of the 17th Annual Conference on Computer Graphics and Interactive Techniques* (Dallas, TX, USA) *(SIGGRAPH '90)*. Association for Computing Machinery, New York, NY, USA, 145–154. https://doi.org/10.1145/97879.97895

Homan Igehy. 1999. Tracing ray differentials. In *Proceedings of the 26th annual conference on Computer graphics and interactive techniques*. 179–186.

Wenzel Jakob and Steve Marschner. 2012. Manifold exploration: A markov chain monte carlo technique for rendering scenes with difficult specular transport. *ACM Transactions on Graphics (TOG)* 31, 4 (2012), 1–13.

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

Wenzel Jakob, Sébastien Speierer, Nicolas Roussel, and Delio Vicini. 2022a. 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

Wenzel Alban Jakob. 2013. *Light transport on path-space manifolds*. Cornell University.

Anton S Kaplanyan, Johannes Hanika, and Carsten Dachsbacher. 2014. The natural-constraint representation of the path space for efficient light transport simulation. *ACM Transactions on Graphics (TOG)* 33, 4 (2014), 1–13.

Hiroharu Kato, Yoshitaka Ushiku, and Tatsuya Harada. 2018. Neural 3d mesh renderer. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 3907–3916.

Pramook Khungurn, Daniel Schroeder, Shuang Zhao, Kavita Bala, and Steve Marschner. 2015. Matching Real Fabrics with Micro-Appearance Models. *ACM Trans. Graph.* 35, 1 (2015), 1–1.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

Samuli Laine, Janne Hellsten, Tero Karras, Yeongho Seol, Jaakko Lehtinen, and Timo Aila. 2020. Modular primitives for high-performance differentiable rendering. *ACM Transactions on Graphics (TOG)* 39, 6 (2020), 1–14.

Tzu-Mao Li, Miika Aittala, Frédo Durand, and Jaakko Lehtinen. 2018. Differentiable monte carlo ray tracing through edge sampling. *ACM Transactions on Graphics (TOG)* 37, 6 (2018), 1–11.

Shichen Liu, Tianye Li, Weikai Chen, and Hao Li. 2019. Soft rasterizer: A differentiable renderer for image-based 3d reasoning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 7708–7717.

Matthew Loper, Naureen Mahmood, Javier Romero, Gerard Pons-Moll, and Michael J. Black. 2015. SMPL: A Skinned Multi-Person Linear Model. *ACM Trans. Graphics (Proc. SIGGRAPH Asia)* 34, 6 (Oct. 2015), 248:1–248:16.

Matthew M Loper and Michael J Black. 2014. OpenDR: An approximate differentiable renderer. In *European Conference on Computer Vision*. Springer, 154–169.

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

Fujun Luan, Shuang Zhao, Kavita Bala, and Zhao Dong. 2021. Unified shape and svbrdf recovery using differentiable monte carlo rendering. In *Computer Graphics Forum*, Vol. 40. Wiley Online Library, 101–113.

Jiahui Lyu, Bojian Wu, Dani Lischinski, Daniel Cohen-Or, and Hui Huang. 2020. Differentiable refraction-tracing for mesh reconstruction of transparent objects. *ACM Transactions on Graphics (TOG)* 39, 6 (2020), 1–13.

Jacob Munkberg, Jon Hasselgren, Tianchang Shen, Jun Gao, Wenzheng Chen, Alex Evans, Thomas Müller, and Sanja Fidler. 2021. Extracting Triangular 3D Models, Materials, and Lighting From Images. *arXiv preprint arXiv:2111.12503* (2021).

Merlin Nimier-David, Delio Vicini, Tizian Zeltner, and Wenzel Jakob. 2019. Mitsuba 2: A retargetable forward and inverse renderer. *ACM Transactions on Graphics (TOG)* 38, 6 (2019), 1–17.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. PyTorch: An Imperative Style, High-Performance Deep Learning

Library. In *Advances in Neural Information Processing Systems 32*. Curran Associates, Inc., 8024–8035. http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf

Nikhila Ravi, Jeremy Reizenstein, David Novotny, Taylor Gordon, Wan-Yen Lo, Justin Johnson, and Georgia Gkioxari. 2020. Accelerating 3d deep learning with pytorch3d. *arXiv preprint arXiv:2007.08501* (2020).

Liang Shi, Beichen Li, Miloš Hašan, Kalyan Sunkavalli, Tamy Boubekeur, Radomir Mech, and Wojciech Matusik. 2020. MATch: Differentiable material graphs for procedural material capture. (2020).

Michael Spivak. 1965. *Calculus On Manifolds: A Modern Approach To Classical Theorems Of Advanced Calculus (1st ed.)*. CRC Press. https://doi.org/10.1201/9780429501906

Eric Veach. 1998. *Robust Monte Carlo methods for light transport simulation*. Stanford University.

Eric Veach and Leonidas J Guibas. 1997. Metropolis light transport. In *Proceedings of the 24th annual conference on Computer graphics and interactive techniques*. 65–76.

Delio Vicini, Sébastien Speierer, and Wenzel Jakob. 2021. Path Replay Backpropagation: Differentiating Light Paths using Constant Memory and Linear Time. *Transactions on Graphics (Proceedings of SIGGRAPH)* 40, 4 (Aug. 2021), 108:1–108:14. https://doi.org/10.1145/3450626.3459804

Delio Vicini, Sébastien Speierer, and Wenzel Jakob. 2022. Differentiable signed distance function rendering. *ACM Transactions on Graphics (TOG)* 41, 4 (2022), 1–18.

Bruce Walter, Stephen R Marschner, Hongsong Li, and Kenneth E Torrance. 2007. Microfacet models for refraction through rough surfaces. In *Proceedings of the 18th Eurographics conference on Rendering Techniques*. 195–206.

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

Kai Yan, Christoph Lassner, Brian Budge, Zhao Dong, and Shuang Zhao. 2022. Efficient Estimation of Boundary Integrals for Path-Space Differentiable Rendering. *ACM Trans. Graph.* 41, 4 (2022), 123:1–123:13.

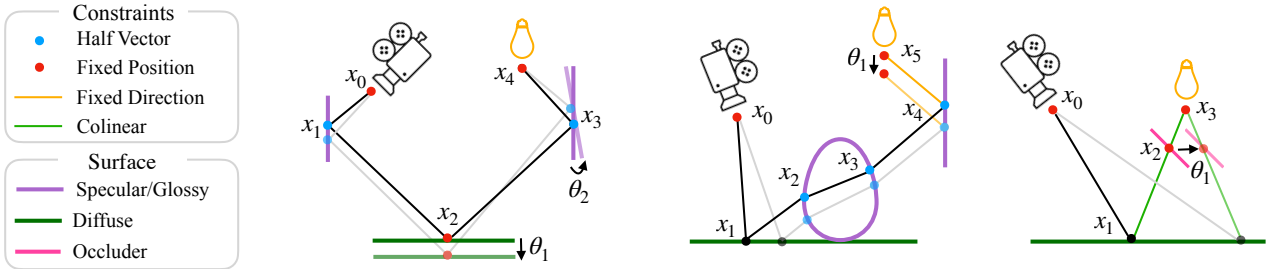
Tizian Zeltner, Iliyan Georgiev, and Wenzel Jakob. 2020. Specular manifold sampling for rendering high-frequency caustics and glints. *ACM Transactions on Graphics (TOG)* 39, 4 (2020), 149–1.

Tizian Zeltner, Sébastien Speierer, Iliyan Georgiev, and Wenzel Jakob. 2021. Monte Carlo estimators for differential light transport. *ACM Transactions on Graphics (TOG)* 40, 4 (2021), 1–16.

Cheng Zhang, Bailey Miller, Kan Yan, Ioannis Gkioulekas, and Shuang Zhao. 2020. Path-space differentiable rendering. *ACM transactions on graphics* 39, 4 (2020).

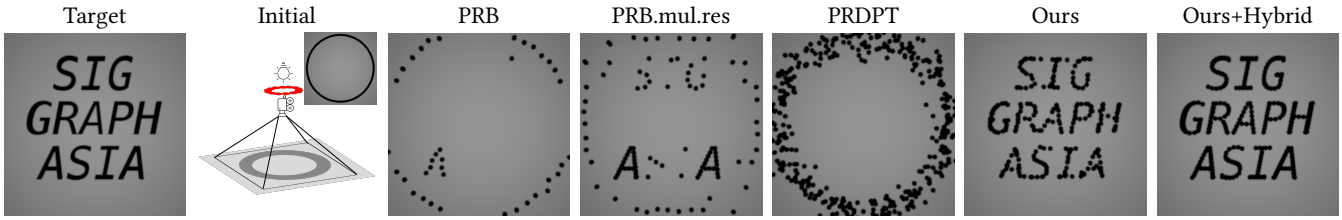
Cheng Zhang, Zihan Yu, and Shuang Zhao. 2021. Path-space differentiable rendering of participating media. *ACM Transactions on Graphics (TOG)* 40, 4 (2021), 1–15.

Shuang Zhao, Lifan Wu, Frédo Durand, and Ravi Ramamoorthi. 2016. Downsampling scattering parameters for rendering anisotropic media. *ACM Transactions on Graphics (TOG)* 35, 6 (2016), 1–11.

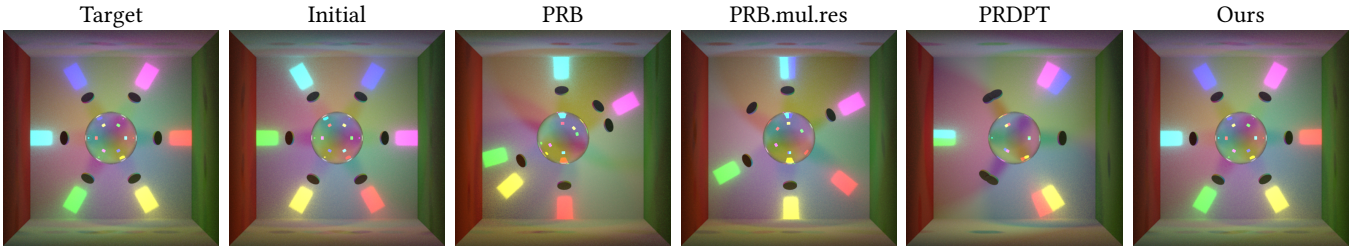


(a) general EPSM (b) caustics EPSM (c) shadow EPSM

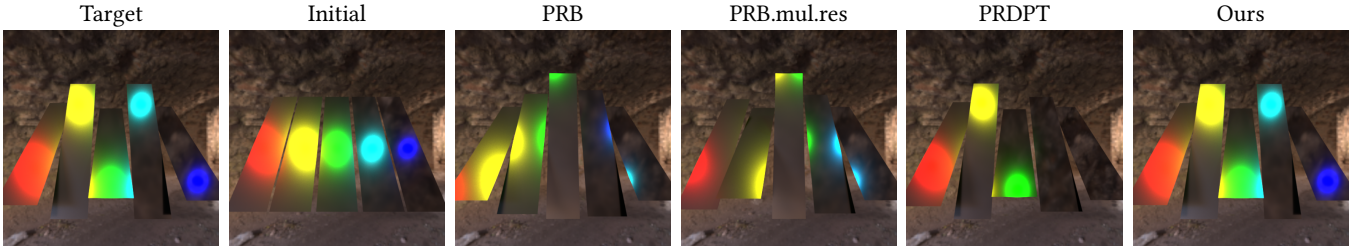
Figure 4: Three types of extended path space manifolds (EPSMs).



(a) Scene Shadow



(b) Scene CornellBox



(c) Scene Highlight

Figure 5: Visual comparisons between our method, Path Replay Backpropagation with reparameterization [Bangaru et al. 2020; Vicini et al. 2021] (PRB), PRB with a multi-scale scheme (PRB.mul.res), and Plateau-reduced Differentiable Path Tracing [Fischer and Ritschel 2022] (PRDPT). The Shadow scene also shows a result of our hybrid optimization scheme (ours+hybrid). The optimization of all scenes is performed using a rendering resolution of 128×128 and 32 spps, while the images displayed in the figure are re-rendered in a higher resolution of 512×512 and 8192 spps. (a) The Shadow scene contains an area light, a floor, and 400 spheres. The area light is put above the spheres and casts shadows onto the floor. The camera is put below the spheres and towards the floor so that the camera can see shadows on the floor. The optimizable scene parameters are the 2D translation vectors of all occluder spheres; Initially the spheres are placed to make a circle and the goal is to cast desired shadows like text 'SIGGRAPH ASIA'. (b) The CornellBox scene contains a refractive glass ball and six area light sources with different colors around the ball. Each area light generates a caustic pattern on the wall. The optimizable scene parameters are the rotating angles of the six light sources. (c) The Highlight scene contains five parallel glossy planes with microfacet GGX [Walter et al. 2007] of different roughness values, and five out-of-view area lights with different colors. The optimizable parameters are the rotation angles of the planes and the 1D horizontal translation offsets of the lights.

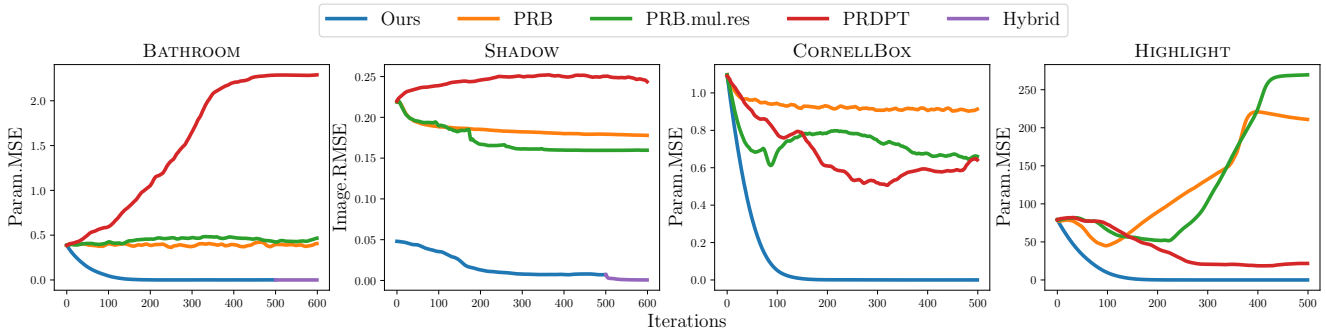
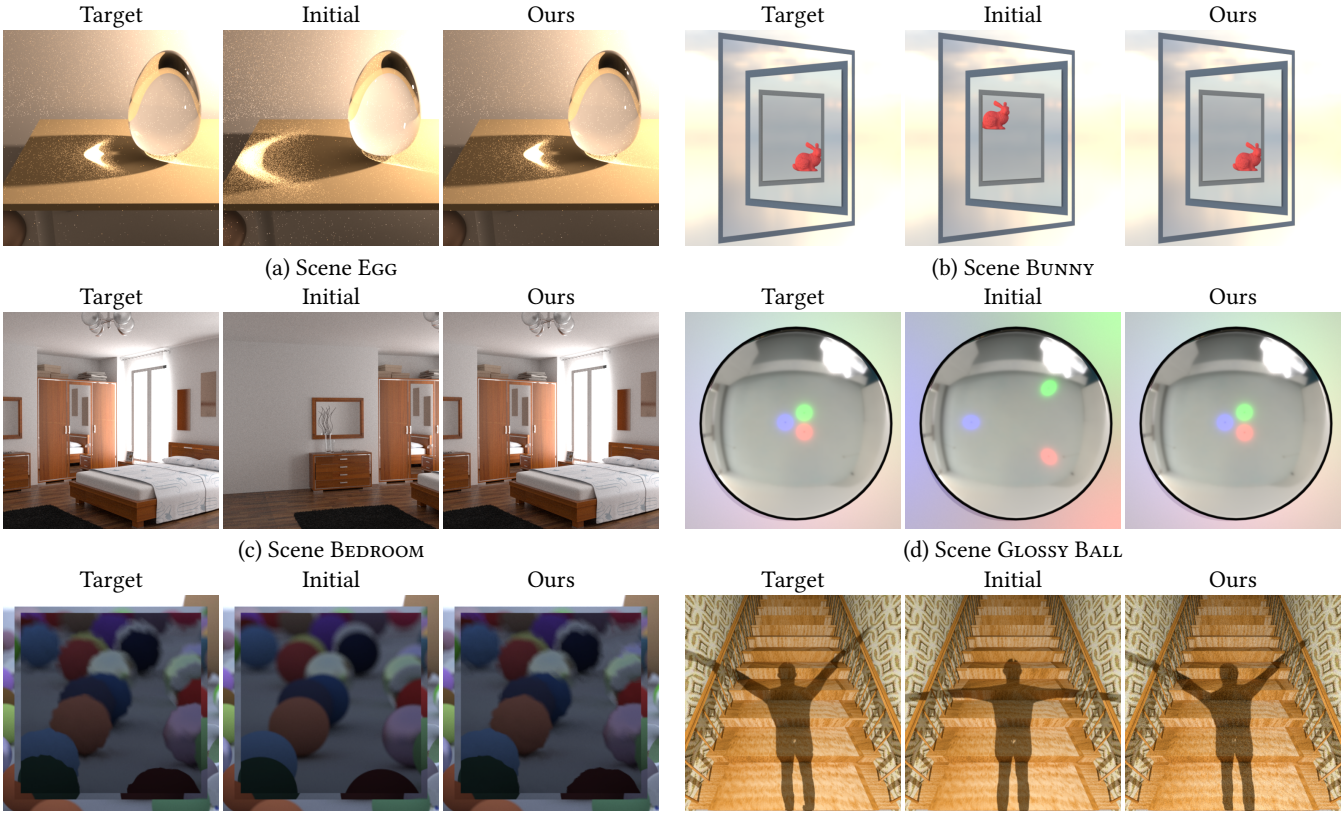


Figure 6: Error curves. We show how the errors change during the optimization of the 4 scenes shown in Fig. 1 and Fig 5. For the Shadow scene featuring 400 spheres, since the target image is hand-drawn and no ground truth scene parameters are available, we provide RMSE between the rendered and target images. For other scenes, we provide the MSE of optimized and target scene parameters.



(a) Scene Egg (b) Scene Bunny

(c) Scene Bedroom (d) Scene Glossy Ball

(e) Scene Glass Slab (f) Scene Human

Figure 7: Additional results of our method (without hybrid scheme). The optimization of all scenes is performed using a rendering resolution of 128×128 and 32 spps, while the images displayed in the figure are re-rendered in a higher resolution of 512×512 and 8192 spps. Scenes Egg and Bedroom are modified from [Bitterli 2016]. (a) Scene Egg. Optimizing the 1D translation offset of the light source through caustics. (b) Scene Bunny. Optimizing the 2D translation vector of the bunny object through nested reflections. (c) Scene Bedroom. Optimizing the camera pose. (d) Scene Glossy Ball. Optimizing the 2D translation vectors of three light sources via glossy highlights. (e) Scene Glass Slab. Optimizing the normal map with a 32×32 resolution of a refractive glass slab via refraction. (f) Scene Human. Optimizing 72 parameters of an SMPL human model [Loper et al. 2015] from its curved shadows on a staircase.