Many-Worlds Inverse Rendering

ZIYI ZHANG, École Polytechnique Fédérale de Lausanne (EPFL), Switzerland
NICOLAS ROUSSEL, École Polytechnique Fédérale de Lausanne (EPFL), Switzerland
WENZEL JAKOB, École Polytechnique Fédérale de Lausanne (EPFL), Switzerland

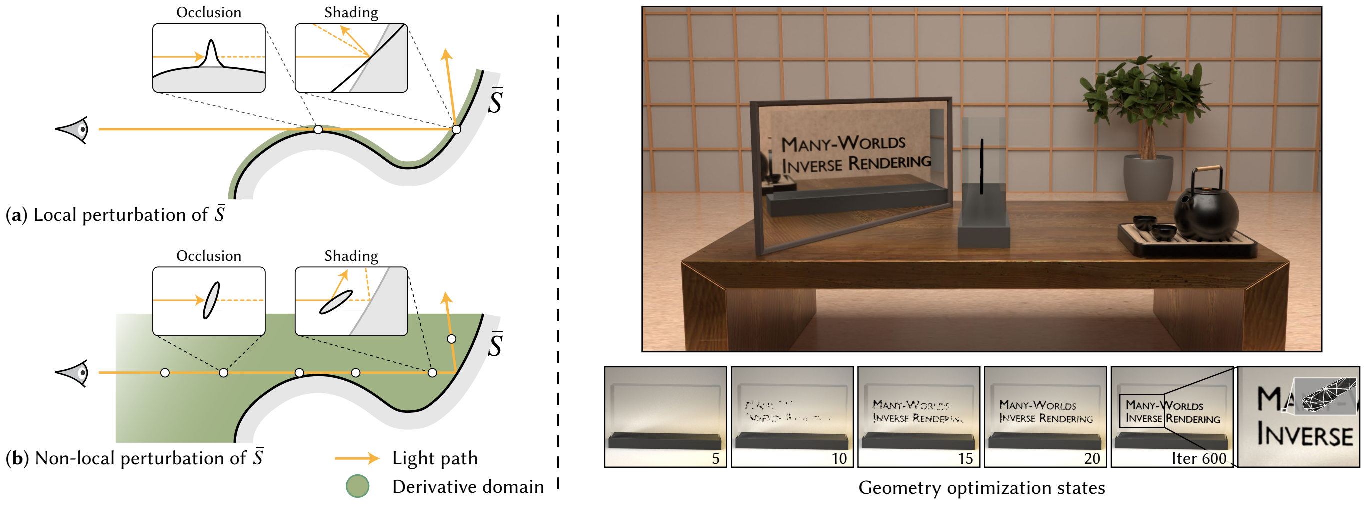


Fig. 1. **Left:** (a) Prior differentiable rendering methods compute how surface deformations impact light transport through occlusion and shading changes. The resulting gradients drive local geometric adjustments. (b) Our method instead considers adding hypothetical surface patches anywhere in 3D space. We simultaneously evaluate many such patches as independent, competing explanations of the input data. This approach, termed *many-worlds derivatives*, extends gradient computation from surfaces into the surrounding space. This combines the robustness of volumes with the efficiency of surface rendering: our method does not require an initial mesh and can be started from an empty scene, while avoiding the expense of transmittance and multiple scattering computations. **Right:** An example reconstruction using many-worlds derivatives: a triangle mesh embedded in glass and observed through a mirror.

Discontinuous visibility changes remain a major bottleneck when optimizing surfaces within a physically based inverse renderer. Many previous works have proposed sophisticated algorithms and data structures to sample visibility silhouettes more efficiently.

Our work presents another solution: instead of evolving a surface locally, we extend differentiation to hypothetical surface patches anywhere in 3D space. We refer to this as a "many-worlds" representation because it models a superposition of independent surface hypotheses that compete to explain the reference images. These hypotheses do not interact through shadowing or scattering, leading to a new transport law that distinguishes our method from prior work based on exponential random media.

The complete elimination of visibility-related discontinuity handling bypasses the most complex and costly component of prior inverse rendering methods, while the extended derivative domain promotes rapid convergence. We demonstrate that the resulting Monte Carlo algorithm solves physically based inverse problems with both reduced per-iteration cost and fewer total iterations.

Authors' Contact Information: Ziyi Zhang, École Polytechnique Fédérale de Lausanne (EPFL), Lausanne, Switzerland, ziyi.zhang@epfl.ch; Nicolas Roussel, École Polytechnique Fédérale de Lausanne (EPFL), Lausanne, Switzerland, nicolas.roussel@epfl.ch; Wenzel Jakob, École Polytechnique Fédérale de Lausanne (EPFL), Lausanne, Switzerland, wenzel.jakob@epfl.ch.

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

Additional Key Words and Phrases: differentiable rendering

ACM Reference Format:
Ziyi Zhang, Nicolas Roussel, and Wenzel Jakob. 2025. Many-Worlds Inverse Rendering. *ACM Trans. Graph.* 1, 1 (September 2025), 17 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 Introduction

Dramatic progress in the area of inverse rendering has led to methods that can fully reverse the process of image formation to reconstruct 3D scenes from 2D images.

This is usually formulated as an optimization problem: given a loss function $\mathcal{L}$ and a rendering $\mathcal{R}(\pi)$ of tentative parameters π, we seek to minimize their composition

$$\pi^* = \operatorname*{argmin}_{\pi \in \Pi} \mathcal{L}(\mathcal{R}(\pi)). \tag{1}$$

The details greatly vary depending on the application, but usually $\mathcal{L}$ will quantify the difference between the rendering and one or more reference images.

Recent work on this problem is largely based on *emissive* volume reconstruction [Mildenhall et al. 2021; Kerbl et al. 2023]. The term "emissive" highlights that this approach does not rely on simulating light physics or material reflectance; instead, the volume is treated as if it was a natural emitter of light and stores color values representing this emitted radiation. These methods are popular because the direct mapping between stored color and observed appearance results in well-behaved optimization problems.

Physically based methods instead seek a more complicated explanation: they simulate emission and scattering inside a general scene that can contain essentially anything: surfaces, volumes, physically based BRDFs, etc. Interreflection adds dense nonlinear parameter dependencies that make this a significantly harder optimization problem. It goes without saying that these two approaches do not compete: when an emissive volume is an acceptable answer, it should always be preferred. This is because it encapsulates material, lighting, geometry and inter-reflection in a unified field representation that promotes speedy and stable convergence.

That said, many applications rely on reconstructions that account for indirect cues—such as shadows and interreflections—and aim to produce physically meaningful results suitable for downstream tasks like relighting and editing. Unfortunately, algorithms designed for this harder physical problem are often *surprisingly brittle*.

A default strategy is to use gradient-based descent to evolve a scene representation (e.g., a triangle mesh) in a domain Π containing the target π^*. It seems only natural that we try to reach this goal by evolving compatible models $\pi_i \in \Pi$ in this domain, e.g., by deforming a tentative surface starting from an initial guess.

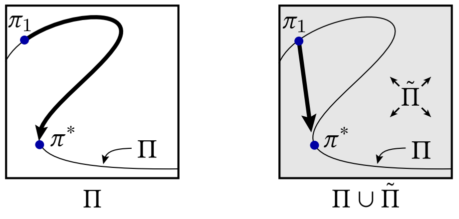
However, a significant complication arises when targeting the physically based flavor of this problem: computing $\nabla_\pi \mathcal{L}(\mathcal{R}(\pi_i))$ with automatic differentiation produces incorrect gradients due to parameter-dependent discontinuities caused by the visibility function. Incorporating specialized techniques to fix this problem simply leads to the next one: the algorithm now outputs sparse gradients on object silhouettes that cause convergence to bad local minima. Preconditioning and regularization can help, but even with all these fixes in place, the optimization often still does not work all that well.

To avoid these problems, we consider optimizations on an extended parameter space:

$$\pi^*, \tilde{\pi}^* = \underset{\pi\in\Pi,\, \tilde{\pi}\in\tilde{\Pi}}{\operatorname{argmin}}\ \mathcal{L}(\tilde{\mathcal{R}}(\pi, \tilde{\pi})), \tag{2}$$

where $\tilde{\mathcal{R}}$ is furthermore parameterized by $\tilde{\pi} \in \tilde{\Pi}$ representing features that we are *unwilling to accept* in the final solution. If the role of these "perpendicular dimensions" diminishes over time, then we can simply discard $\tilde{\pi}^*$ at the end and take π^* to be the solution of the original optimization problem.

The parameter space extension serves two purposes: first, it turns a circuitous trajectory through a non-convex energy landscape into a more direct route by using the extra degrees of freedom:



Second, we will choose the extension so that visibility changes in the original problem cease to be discontinuous in the extended space, which greatly reduces algorithmic complexity.

Seen from a high level, this isn't a new idea: numerous works in emissive surface reconstruction [Yariv et al. 2021; Wang et al. 2021; Miller et al. 2024] have shown that the problem becomes tractable when starting from an emissive volume like NeRF. However, taking this idea to the world of physically based rendering (e.g., by optimizing a scattering microflake volume) leads to several serious problems:

(1) **Speed**. Introducing random volumes into a physically based renderer requires solving the *radiative transfer equation* (RTE), which involves free-flight sampling and transmittance estimation along every ray segment. In heterogeneous volumes, both are computationally expensive iterative processes, making this approach significantly slower than surface-based methods.
(2) **Extraction**. A separate optimization is required to convert converged volumes into optically similar surfaces with BRDFs, and the resulting surface is often only an approximation of the volume appearance. Prior work related the bulk properties of microfacets and microflakes [Dupuy et al. 2016; Miller et al. 2024], but this insight cannot be used to convert a general microflake volume into a surface, nor does it generalize to the richness of reflectance models in modern rendering systems.

Instead of introducing an exponential volume that blends all possible surfaces multiplicatively, we retain a surface (denoted $\bar{S}$) and augment it with a spatial representation S to model *non-local perturbations* of that surface.

Consider placing a new surface patch above the original surface. Standard differentiation of a path tracer does not account for this type of change, yet it is a valid way to evolve the scene representation—one that evolves the surface in a non-local manner.

Importantly, these non-local perturbations can occur simultaneously along a light path: the optimization of a potential surface patch at one location is independent of that at another location. This is because these hypothetical surfaces *aren't real* (or at least, not yet). They constitute different surface possibilities that may or may not become part of the final reconstruction. It makes little sense for them to affect each other.

This leads to a new transport law that we refer to as *many-worlds derivative transport*. The term "many-worlds" not only suggests a distribution of surfaces but also emphasizes the *non-interacting* nature of different perturbations ("worlds").

Differentiation of this model produces dense surface derivatives in an extended domain. Since we avoid diffusing values into an exponential volume, the optimization remains as efficient as surface evolution methods. Finally, mesh extraction is a simple projection that discards S, and all of this is easily incorporated into a physically based path tracer.

Although this paper sometimes refers to the spatial representation S as a "volume", our method should *not* be interpreted as volume reconstruction. Specifically, we never optimize this "volume" S to match input images; instead, we use it to refine the surface $\bar{S}$. Moreover, our method interacts with BRDFs rather than phase functions, and has no concept of transmittance in its radiative transfer.

Our main contribution is a solution to the classic discontinuity problem that is algorithmically simpler and computationally more efficient than prior methods. We present two derivations that alternatively formulate the central algorithm as either

- a non-local perturbation of an evolving surface (Section 3), or
- an extension of the surface derivative domain that removes the need for silhouette sampling (Section 4 and Appendix A).

2 Related work and background

This section reviews related prior work on inverse rendering of solid geometry. The simultaneous disentanglement of geometry, material and lighting is orthogonal to the topic of this work.

2.1 Physically based inverse rendering

The physically based approach tries to model all available information in input images by accounting for lighting, materials, and interreflection. This leads to several challenges.

Discontinuities. Analytic differentiation of the rendering equation produces a boundary derivative term that is not sampled by primal rendering strategies. Consequently, applying automatic differentiation to a rendering algorithm produces incorrect gradients.

Estimating the missing derivative requires sampling rays along shape boundaries (i.e., silhouettes). Prior works [Li et al. 2018; Zhang et al. 2020; Yan et al. 2022; Zhang et al. 2023a; Xu et al. 2023, 2024] have extensively studied the theory and practice of estimating this boundary term. Another approach employs reparameterizations to convert the boundary contribution to neighboring regions, enabling simpler area-based formulations. These methods [Loubet et al. 2019; Bangaru et al. 2020] often require tracing auxiliary rays to detect the presence of neighboring silhouettes. For geometry representations based on *signed distance fields* (SDFs), specialized algorithms can alleviate some of this cost [Vicini et al. 2022; Bangaru et al. 2022; Wang et al. 2024].

While the theory of the boundary term continues to evolve, practical efficiency and robustness remain challenging. Current methods still have limitations concerning bias (e.g., specular surfaces) and variance (e.g., when multiple silhouettes are in close proximity). These methods are difficult to implement, and the cost of boundary handling is usually the main bottleneck of the entire optimization.

Sparse gradients. The derivative of the rendering process can be decomposed into two types: the continuous part and the discontinuous part. We refer readers to Figure 4 in Zhang et al.'s work [2023a] for an illustration. Since our paper focuses on geometry optimization, we refer to the continuous part as the *shading derivative* and the discontinuous part as the *boundary derivative* in the following.

The shading derivative is defined everywhere on the surface and primarily affects the surface through its normal and heavily depends on the lighting. Adjoint rendering methods [Nimier-David et al. 2020; Vicini et al. 2021b] exploit symmetries to compute the shading derivative with linear time complexity. Zeltner et al. [2021] analyze different strategies for variance reduction in this process. The boundary derivative, on the other hand, is only defined on the visibility silhouette of the surface. These surface derivatives are sparse in two senses:

(1) A single view only observes a limited subset of all possible silhouettes, and a derivative step is likely to introduce a kink at this subset. Nicolet et al. [2021] reparameterize meshes on a space that promotes smoothness to mitigate this issue.
(2) The derivatives are only defined on the surface, not throughout the entire 3D space. Although light transport is simulated in this larger space, only the neighborhood of a surface can use the computed gradients. Regions far from the initial guess, must wait until the surface extends to them to receive gradients. Mehta et al. [2023] propose an elegant solution for vector graphics in 2D; however, they do not extend their approach to define surface derivatives in the entire 3D space.

The sparsity has two implications: (1) the optimization is slow because it needs many iterations to deform the surface to the target shape, and (2) a good initial guess is needed, especially when the target shape has a complicated topology or self-occlusions.

Approximations. There is a substantial body of literature on approximating the PBR process to make optimization easier. These methods often bypass the discontinuity problem by assuming knowledge of a shape mask (implicitly assuming the shape is directly observable), approximating boundary derivatives [Laine et al. 2020], or diffusing the rendering process into a higher-dimensional space [Fischer and Ritschel 2023]. Additionally, many methods approximate the light transport for efficiency [Zhang et al. 2021a; Jin et al. 2023; Zhang et al. 2021b; Boss et al. 2021]. These methods are physically inspired and can handle complex scenes, while our theory focuses on differentiating a fully physically based rendering process.

Nimier-David et al. [2022] demonstrate that a physically based non-emissive volume can be optimized to fit the input images. These methods are more physically accurate than the emissive volume methods but do not yield a surface representation in the end.

2.2 Emissive volume inverse rendering

A large body of work based on variants of the NeRF [Mildenhall et al. 2021] technique replaces the expensive rendering process with an emissive volume, which leads to a fog-like volume that does not accurately represent a solid surface.

Later work [Yariv et al. 2021; Wang et al. 2021; Li et al. 2023; Miller et al. 2024] added a surface prior to the volume model by introducing an SDF and deriving the volume density from it. These methods map SDF values to volume densities using a distribution function, assuming that a ray intersects a stochastic object following a Markov process. The variance of the distribution is learnable during optimization: it begins with a high-variance distribution, mimicking the NeRF model, and gradually reduces to ensure that a surface can confidently be extracted.

These SDF-parameterized volume reconstructions can be seen as a *blurry* version of surface-based methods without indirect effects. When the variance is zero, these methods simplify to surface-based methods[1]. With moderate variance, the effect of a solid object, which we can interpret as "terminating a ray at a position with probability 1", is diffused to the neighborhood, interpretable as "terminating a ray in this neighborhood according to a probability density".

[1]This equivalence is theoretical. In practice, algorithms designed for volume optimization cannot handle such a Dirac-delta density distribution due to discontinuities.

2.3 Surfaces versus volumes

Several works studied the relationship of surface and volume rendering: Heitz et al. [2016] and Dupuy et al. [2016] describe microfacet surfaces within the framework of random volumes. Vicini et al. [2021a] incorporated non-exponential transmittance to account for correlations arising from opaque surfaces. Miller et al. [2024] formalize microflake volumes as the relaxation of a stochastic surface model and solve inverse problems using this representation. Seyb et al. [2024] recently incorporated a richer set of spatial correlations, producing a forward rendering framework that spans the full continuum ranging from pure volume to surface-like interactions.

3 Method

3.1 Motivation

Extended parameter space. Our algorithm optimizes a surface $\bar{S}$ in an indirect manner via a distribution S of potential surfaces that defines an associated density field in 3D space. The surface $\bar{S}$ is a product of this field, for example by extracting a level set. By modifying the distribution S rather than $\bar{S}$ directly, we enable gradient propagation throughout the entire space, not just on the surface itself. The details of how these spaces are defined are orthogonal to our method, and we describe them in Appendix C.

Figure 1b demonstrates the effect of adding a *hypothetical surface patch* (drawn from S) into a scene containing the surface $\bar{S}$. This patch modifies how light propagates in the scene, thereby changing the surface rendering of $\bar{S}$. By adjusting the existence and properties (e.g., normals, BRDFs) of such patches, we iteratively improve $\bar{S}$ to better match the target image.

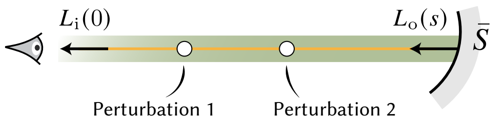
We refer to this a *non-local perturbation* of $\bar{S}$: optimizing such hypothetical patches propagates derivatives across the entire domain of S, rather than confining updates locally on the surface.

There is a noteworthy connection to prior work: in the limiting case where the perturbation position coincides with $\bar{S}$ itself, our derivatives exactly match standard surface derivatives in physical light simulation [Zhang et al. 2020, Zhang et al. 2023a]. Appendix A provides details on this equivalence, showing that our method is a generalization of local surface evolution, extending its domain while preserving its geometric meaning.

The optimization converges when no further perturbation improves the match between $\bar{S}$ and the target image. At this point, we discard S and keep $\bar{S}$.

Within each iteration of the optimization, $\bar{S}$ serves as a static background, providing base colors for perturbations without being optimized itself. Thus, we also refer to $\bar{S}$ as the *background surface.*

Conflicting possibilities. Consider a simple case of two non-local perturbations along a ray. We think of them as competing candidates for improving the agreement between the radiance arriving from $\bar{S}$ and a reference. For example, suppose that $\frac{\partial \text{loss}}{\partial L_\text{i}}$ indicates that the current pixel's radiance is too high—in this case, the same information should be propagated to both positions without weighting.



Here, it might seem natural to distribute the target update between the two positions — say, by scaling it by $\frac{1}{2}$. However, this weighting implicitly assumes the two perturbations *compound* to refine the same surface $\bar{S}$. In reality, they are *mutually exclusive* possibilities — once optimization converges, only one will contribute to the final radiance, without any kind of blending.

This principle extends to more than two perturbations: all candidate positions along the ray should receive the same target update, as if existing in *many worlds* that do not interact. Ultimately, the ray will intersect only one of these possibilities, which becomes part of the final surface.

The above discussion explains the motivation behind our method at an intuitive level. Our next goals are therefore to quantify non-local perturbations and derive the derivative transport law.

3.2 Many-worlds derivative transport

Single perturbation case. Consider a scene containing only the background surface $\bar{S}$, where the radiance propagating along ray $(\mathbf{y}, \mathbf{x})$ with direction $-\boldsymbol{\omega}$ remains constant:

$$L_\text{i}^{\bar{\text{S}}}(0) = L_\text{o}^{\bar{\text{S}}}(s),$$

where $L_\text{i}^{\bar{\text{S}}}(0) \coloneqq L_\text{i}^{\bar{\text{S}}}(\mathbf{x}, \boldsymbol{\omega})$ (incident radiance at $\mathbf{x}$) and $L_\text{o}^{\bar{\text{S}}}(s) \coloneqq L_\text{o}^{\bar{\text{S}}}(\mathbf{y}, -\boldsymbol{\omega})$ (outgoing radiance at $\mathbf{y}$) are parameterized by distance.

For a non-local perturbation candidate at distance t, we model its impact on radiative transport as:

$$L_\text{i}(0) = \alpha(t) \underbrace{L_\text{o}^{\text{S}}(t)}_{\text{perturbed}} + [1 - \alpha(t)] \underbrace{L_\text{o}^{\bar{\text{S}}}(s)}_{\text{original}}, \tag{3}$$

where $\alpha(t) \in [0, 1]$ is the probability of a hypothetical surface patch existing at t. The perturbed radiance $L_\text{o}^{\text{S}}(t)$ computes reflected radiance *as if* the patch were inserted at t, while the rest of the scene remains $\bar{S}$.

While this formulation is of little use for physically based rendering, its derivative (with respect to any parameter π) provides the means to optimize on the extended parameter space:

$$\partial_\pi L_\text{i}(0) = \underbrace{\partial_\pi \alpha(t) \, [L_\text{o}^{\text{S}}(t) - L_\text{o}^{\bar{\text{S}}}(s)]}_{\text{(i) Occlusion}} + \underbrace{\alpha(t) \, \partial_\pi L_\text{o}^{\text{S}}(t)}_{\text{(ii) Shading}}. \tag{4}$$

If $\partial_\pi L_\text{i}(0)$ is positive, we can increase the incident radiance by

(1) Raising the occupancy at $\alpha(t)$ if this perturbation is favorable, i.e., $L_\text{o}^{\text{S}}(t) > L_\text{o}^{\bar{\text{S}}}(s)$, or lower it otherwise.
(2) Increasing the reflected radiance $L_\text{o}^{\text{S}}(t)$, e.g., by altering the normal or BRDF of the hypothetical surface.

Both operations locally update the distribution S at t.

Multiple perturbation case. We now extend to the situation where every point along the ray can be considered a potential perturbation. As discussed in Section 3.1, we aim to distribute the target update uniformly across all possibilities along one segment.

Similar to the reverse-mode derivative of an addition $c = a + b$ that simply forwards the derivative towards its operands a and b without weighting ($\frac{\partial c}{\partial a} = 1$, not $\frac{1}{2}$), we sum radiative contributions from distinct perturbations using a continous integral to achieve an

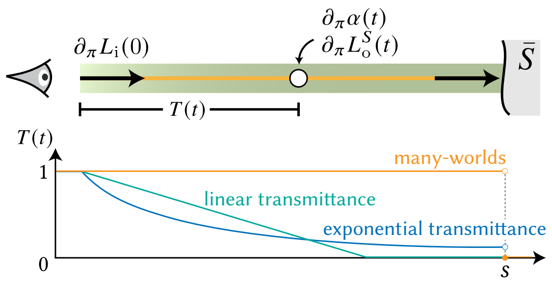


Fig. 2. **Many-worlds derivative transport**. The propagated derivative at distance t is only weighted by the local radiance difference and the local occupancy respectively (Equation 6). In contrast to exponential or non-exponential (e.g., linear [Vicini et al. 2021a]) volumes, the notion of *transmittance* disappears as it models nonsensical inter-world shadowing.

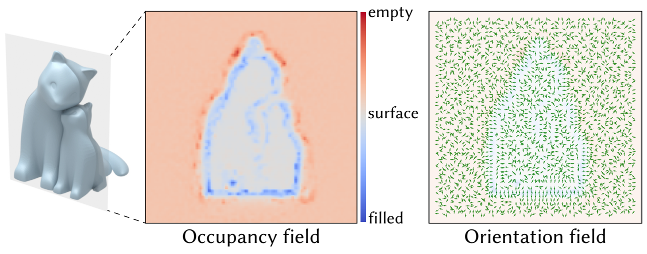


Fig. 3. **Occupancy and orientation fields.** The above images visualize the contents of an occupancy (α) and orientation ($\boldsymbol{\beta}$) field following an optimization. The former models the probability of a surface existing at a position, while the latter assigns normal directions.

analogous propagation behavior:

$$L_i(0) = \int_0^s \Big(\alpha(t)\ \underbrace{L_o^S(t)}_{\text{candidate at } t} + [1-\alpha(t)]\ \underbrace{L_o^{\bar{S}}(s)}_{\text{background}}\Big)\,dt. \tag{5}$$

Differentiating it yields the many-worlds derivative transport law:

$$\partial_\pi L_i(0) = \int_0^s \Big(\partial_\pi \alpha(t)\,[L_o^S(t) - L_o^{\bar{S}}(s)] + \alpha(t)\,\partial_\pi L_o^S(t)\Big)\,dt. \tag{6}$$

The derivative in Equation 6 lacks transmittance terms that would ordinarily model attenuation along the ray (Figure 2). This stems from the core principle that distinct worlds must not interact.

The only way in which the background surface $\bar{S}$ manifests in this equation is to provide a single baseline radiance value $L_o^{\bar{S}}(s)$ needed to compute a difference of radiance values. As a result, $\bar{S}$ does not directly receive gradients; yet, it still evolves during optimization as a result of changes in the distribution S that generates $\bar{S}$.

Parameterization. Equation 6 requires derivative propagation towards two quantities in the extended parameter space: the probability of encountering a surface within the distribution, and the outgoing radiance $L_o^S(\mathbf{x},\boldsymbol{\omega})$ determined by its properties.

Any S with differentiable realizations of these quantities is in principle suitable—we use an occupancy field $\alpha(\mathbf{x}) : \mathbb{R}^3 \to [0,1]$ and an orientation field $\boldsymbol{\beta}(\mathbf{x}) : \mathbb{R}^3 \to \mathcal{S}^2$ (Figure 3).

The orientation field $\boldsymbol{\beta}(\mathbf{x})$ assigns a normal direction to the surface patch at $\mathbf{x}$. The occupancy field $\alpha(\mathbf{x})$ [Mescheder et al. 2019; Niemeyer et al. 2020] models the probability

$$\alpha(\mathbf{x}) \coloneqq \Pr\{\mathbf{x} \text{ is inside of } S\}, \tag{7}$$

and is zero when encountering the back side ($\boldsymbol{\omega}\cdot\boldsymbol{\beta}(\mathbf{x}) < 0$).

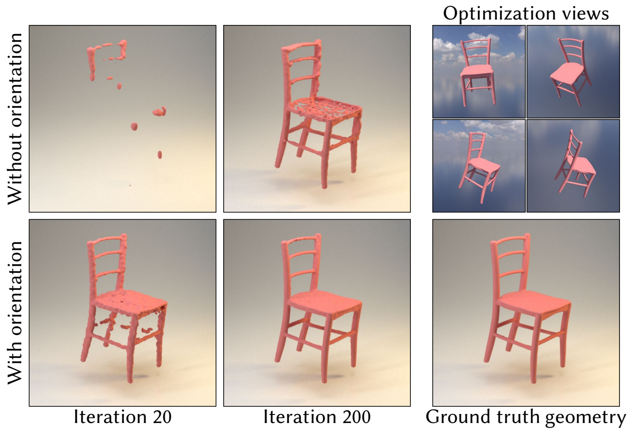


Fig. 4. **Importance of the orientation field.** We compare reconstructions done *with* and *without* an orientation field $\boldsymbol{\beta}$. The top row without $\boldsymbol{\beta}$ uses an isotropic normal distribution. A lack of orientation information dramatically slows down convergence and produces incorrect meshes.

Anisotropy is crucial for physically based inverse rendering. Figure 4 demonstrates this: assuming a uniform normal distribution for surface patches produces incorrect results. This happens because the reference scene is surface-based, where light reflection is highly anisotropic—a property that isotropic distributions fail to capture.

This concludes our derivation via non-local surface perturbations. To reinforce these results and gain a deeper understanding:

(1) Appendix A re-derives Equation 6 by analyzing standard surface derivatives and extending them into space.
(2) Appendix B frames our approach using the mathematical language of random volumes.

3.3 Primal rendering

Differentiable rendering pipelines normally repeatedly render a primal image, differentiate a loss, and then backpropagate derivatives. The previous discussion was only concerned with derivative propagation, and there is thus a question of how to generate a primal image of our extended parameter space[2].

A natural choice is to only use the background surface $\bar{S}$ for primal rendering. (Note that primal images serve solely to compute the adjoint radiance $\frac{\partial \text{loss}}{\partial L_i}$—we still employ Equation 6 for derivative propagation.) Unfortunately, this approach fails catastrophically in our framework: regions beyond the surface $\bar{S}$ are *excluded* from the loss computation yet still receive gradient updates, causing optimization to become unstable, divergent, and effectively random.

We thus use a primal rendering that incorporates both the surface $\bar{S}$ and the distribution S. Recall that the many-worlds principle

[2]In many applications, derivative propagation arises naturally from automatic differentiation of the primal computation. However, this coupling introduces bias in physically based rendering [Nimier-David et al. 2020, Section 3.2], which necessitates separate, uncorrelated Monte Carlo simulation of primal rendering and derivative transport.

manifests in Equation 5 as a direct *summation* of non-local perturbations. This summation is not directly suitable for primal rendering as it yields unbounded values. For all experiments in this work, we compute an *average* over all perturbations (different from an average over all possible surface renderings) by scaling Equation 5 by a factor of $1/s$:

$$L_i(0) = \frac{1}{s}\int_0^s \Big(\alpha(t)\,L_o^S(t) + [1-\alpha(t)]\;L_o^{\bar{S}}(s)\Big)\,dt. \tag{8}$$

The scaling is an empirically motivated choice rather than a theoretically unique solution. Our experiments demonstrate that this normalization is straightforward to implement and produces meaningful adjoint radiances ($\frac{\partial \text{loss}}{\partial L_i}$) that enable rapid convergence.

Relation to "radiance field loss". A recent work [Zhang et al. 2025] instantiated our many-worlds framework for radiance field reconstruction, specializing it to the simplified setting of pure emission without scattering. This enables several simplifications: (1) due to the lack of BRDF and light integration, noise-free radiance values can be retrieved from an appropriate representation (typically a neural network), (2) ray marching replaces Monte Carlo sampling, and (3) rays are traced from fixed pixel centers, so the pixel footprint integral also disappears.

As a result, there is a 1:1 mapping between surface radiance and reference pixel values, allowing individual losses to be defined for each potential surface, which is unattainable in our setting. Since radiance contributions from different perturbations are never summed, their method sidesteps the scaling factor in Equation 8. In contrast, this work addresses the more difficult nested integral problem inherent to physically based rendering, where such simplifications do not apply.

4 Discussion

Pseudocode. We present one possible implementation of our many-worlds framework in Mitsuba 3 [Jakob et al. 2022]. Algorithm 1 uses a variable `mode` to distinguish between the primal rendering pass and the derivative propagation pass.

Relation to surface derivatives. Prior work on geometry differentiation have proposed *local derivative formulations* [Zhang et al. 2020, Zhang et al. 2023a] to quantify how small perturbations of a surface affect radiative transport across the entire scene in physical light simulation.

Appendix A extends such formulation to measure how tiny changes of *any* hypothetical surface patch within S influence radiative transport. This offers a quantitative way to re-derive the many-worlds derivative transport law using established theory. We made the following observations:

(1) Our method *does the right thing* near the surface: the optimization behavior matches surface differentiation algorithms without requiring explicit silhouette sampling.
(2) Visibility and shading derivatives are combined into a unified expression in the extended parameter space. This unification is impossible on the surface $\bar{S}$, as the two derivatives are defined on different domains. Unlike prior work—which required separate algorithms to compute these two types of derivatives—our method drastically reduces algorithmic complexity.

Listing 1. Pseudocode of the Many-Worlds primal/backward pass

```
mode = "Primal"  # Or "Backward"

def Li(x, $\omega$):
    # Pick a segment to interact with a surface patch
    $k_{mw}$ = $\lfloor k_{max}$ * rand()$\rfloor$
    return Li_k(x, $\omega$, $k_{mw}$, 0)

def Li_k(x, $\omega$, $k_{mw}$, $k$):
    if $k$ > $k_{max}$: # Path length exceeds limit
        return 0

    # Radiance estimate from background surface
    s = ray_intersect($\bar{S}$, x, $\omega$)
    x' = x + s * $\omega$ # Advance to surface
    $\omega'$, $w_{brdf}$ = sample_brdf(x', -$\omega$)
    L_bg = Le(x',-$\omega$) + Li_k(x', $\omega'$, $k_{mw}$, $k$ + 1) * $w_{brdf}$
    if $k$ != $k_{mw}$: # Segment is before/after the sampled segment
        return L_bg

    # Radiance estimate from sampled surface patch
    t, $w_{surf}$ = sample_surface(rand(), s)
    if mode == "Primal":
        weight = $w_{surf}$ / s # Primal pass: not differentiated
    else:
        weight = $w_{surf}$ # Backward pass: derivative propagation
    x' = x + t * $\omega$ # Advance to sampled surface
    $\omega'$, $w_{brdf}$ = sample_brdf(x', -$\omega$)
    occupancy = $\alpha$(x') # Occupancy at sampled point
    L_fg = Le(x',-$\omega$) + Li_k(x', $\omega'$, $k_{mw}$, $k$ + 1) * $w_{brdf}$
    return lerp(L_bg, L_fg, occupancy) * weight
```

Relation to volume rendering. At a high level, both inverse volume rendering [Nimier-David et al. 2022] and our method can be interpreted as optimizing *a distribution of surfaces*. However, the two approaches differ fundamentally in how they interact with this distribution: when multiple potential surfaces exist along a ray, how do we model their interplay?

- Exponential volume rendering treats interactions as statistically independent events, leading to a memoryless Poisson process where multiple interactions occur, weighted by relative occlusion probabilities [Bitterli et al. 2018].
- Our method treats interactions as mutually exclusive events, ensuring potential surfaces along a ray do not interact via shadowing or scattering.

Consider a distribution S modeling a nearly empty scene containing a single infinite plane with an uncertain offset. Within any realization of this distribution, light traveling towards this plane will scatter *exactly once*. In contrast, the volume model diffuses the ray-surface interaction into a band of microflakes that cause *arbitrarily long* scattering chains. This not only leads to a significant increase in computational cost, but also discombobulates the optical interpretation of these interactions: the effect of multiple scattering must later be approximated as a surface BRDF, a process that in general admits no exact solution and relies on approximation.

Traditional inverse rendering pipeline (which renders the scene to compute a loss) collapses under the mutually exclusive assumption,

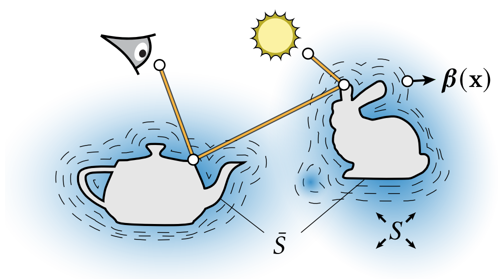


Fig. 5. Our method refines a surface $\bar{S}$ by optimizing a distribution S of possible surfaces. Each hypothetical surface patch drawn from S is independently adjusted to improve the matching between $\bar{S}$ and the target image. To define the behavior of possible surfaces in space, S is parameterized by an occupancy field $\alpha(\mathbf{x})$ and an orientation field $\boldsymbol{\beta}(\mathbf{x})$.

because it would render each potential surface as a separate scene, thus unrealistically requiring *every* potential surface to match the reference image. Instead, we optimize potential surfaces by refining the rendering of a background surface $\bar{S}$, giving the algorithm a clear convergence target (Figure 5). Such comparative adjustments lead to the notion of *non-local surface perturbations*, and simultaneous optimization of all perturbations leads to our *many-worlds* derivative transport.

5 Results

Prior work on physically based inverse rendering often includes comparisons of *forward derivatives* to other competing methods or finite difference-based reference derivatives. They reveal the change in rendered pixels when perturbing a single scene parameter. However, such a comparison is neither applicable nor meaningful in our method: first, our method computes the extended derivative on a higher-dimensional domain, which therefore cannot be compared to existing methods. Second, the need for such comparisons is motivated by the complexity of formulations that deal with discontinuous visibility, but differentiation our representation is "trivial" and therefore not interesting. Because of this, we demonstrate the correctness and performance of our method solely through end-to-end optimizations.

All results presented in this paper exclusively use many-worlds derivatives and *explicitly disable surface derivatives* on S, even for the albedo optimization demonstrated in Figure 7.

Multi-view reconstructions. Figure 6 presents multi-view object reconstructions of a known material in various settings using our method. All experiments use the Adam optimizer [Kingma and Ba 2014]. We found that disabling momentum in the early iterations helped avoid excessive changes while the occupancy field was still very far from convergence. Alternatively, a stochastic gradient descent optimizer produces comparable results. During the reconstruction, each view is rendered at a 512×512 resolution.

The last four rows of Figure 6 show reconstructions involving perfectly specular surfaces. No prior PBR method could handle such scenes: reparameterization-based methods would need to account for the extra distortion produced by specular interactions, while silhouette segment sampling methods would need to find directional emitters through specular chains. Both are complex additional requirements that would be difficult to solve in practice, while the problem simply disappears with the many-worlds formulation.

Albedo reconstruction. This paper primarily focuses on geometry, but our method can also be used to optimize materials or lighting. Figure 7 presents experiments where we jointly optimize the geometry and albedo texture of an object. We store this spatially varying albedo on an additional 3D volume that parameterizes the BSDF of the many-worlds representation.

Benefits of assumption-free geometry priors. Figure 8 compares our method to a technique that evolves an SDF using reparameterizations [Vicini et al. 2022]. These experiments demonstrate that our method requires significantly fewer iterations because new surfaces can be materialized from the very first iteration.

We use 20 random optimization views, with each view rendered at 256×256 pixels. Both methods utilize a geometry grid resolution of 128^3. The time required to perform one iteration of the optimization for one view is 0.25 seconds for our method, 0.38 seconds for the large sphere initialization and 0.32 seconds for the small sphere initialization. Our method benefits from the efficiency of ray-triangle intersections, whereas the SDF representation relies on a more costly iterative sphere tracing algorithm. These measurements also show how sphere tracing slows down with smaller steps due to complicated geometry near a ray, as evidenced by the slower performance of the larger sphere initialization.

Optimizing all positions at once. Figure 9 replicates the chair reconstruction experiment of a recent work by Zhang et al. [2023a] to demonstrate the benefits of the extended parameter space.

Interior topological changes. Figure 10 illustrates a limitation of our method: it does not robustly handle interior topological changes. This issue arises because many-worlds derivatives extend only to the exterior of $\bar{S}$, leaving the interior unsupervised, much like traditional surface evolution methods. When attempting to create a hole, we rely on shading derivatives to bend the surface inward, which sometimes produces a hole as desired (top row). However, optimization like this is sensitive to lighting conditions; as shown in the bottom row, a different lighting can cause the optimization to stall.

To address this limitation, Mehta et al. [2023] proposed explicitly testing whether creating a cone-shaped hole is beneficial, and Zhang et al. [2025] suggested sampling the background surface stochasticity to occasionally permit visibility through high-occupancy regions. We leave the exploration of these extensions for future work.

The sphere initialization is used solely to demonstrate this limitation; with an assumption-free initialization, our method converges correctly and more quickly in both lighting conditions.

Subtractive changes. Figure 11 shows the optimization states for an initialization with random geometry. We used 16 optimization views and the same setup as in Figure 6. Since the many-worlds derivative extends the surface derivative domain without approximations, our method naturally address scenarios that surface derivatives can handle—in this case, removing superfluous geometry and deforming the rest to reconstruct the desired object.

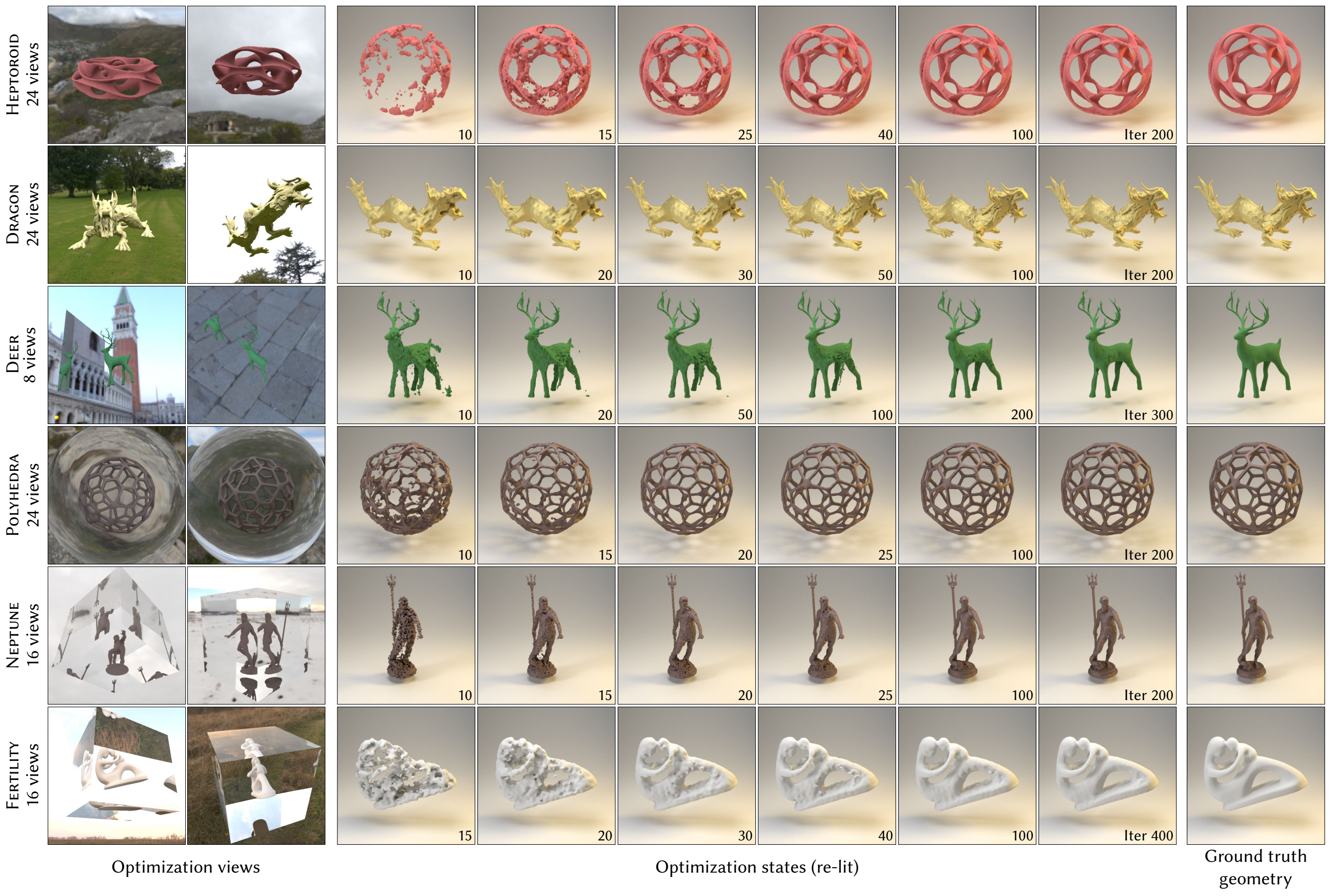


Fig. 6. **Multi-view geometry reconstruction**. For the Deer scene, all 8 views are behind the object and we only see the front side in the mirror. For Polyhedra, Neptune and Fertility, the object is inside a spherical or cubic smooth glass container. The material is known during optimization: the Dragon has a rough gold material, Polyhedra and Neptune are made of copper oxide, and others are diffuse.

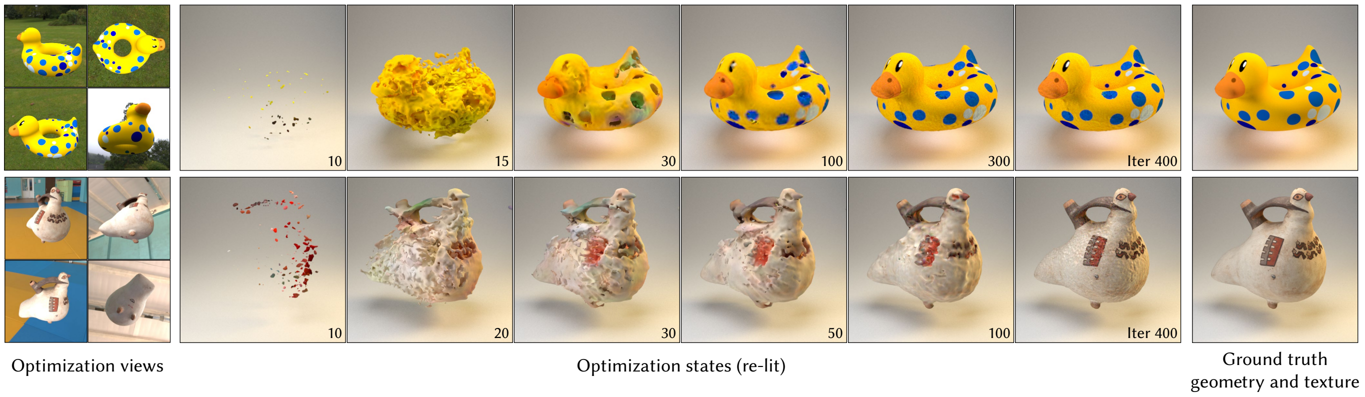

10
15
30
100
300
Iter 400
10
20
30
50
100
Iter 400
Optimization views
Optimization states (re-lit)
Ground truth geometry and texture

Fig. 7. **Material optimization**. This experiment demonstrates joint optimization of geometry and an albedo texture. Our method focuses on geometric optimization, but it is compatible with more general inverse rendering pipelines that furthermore target material and lighting.

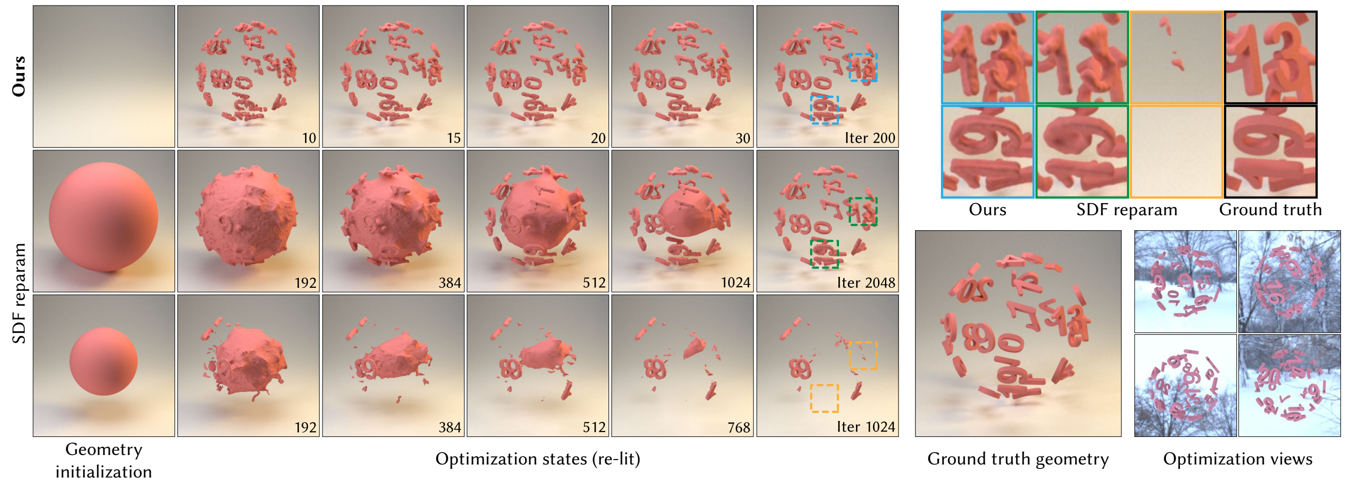


Fig. 8. **Assumption-free optimization.** Prior surface optimization methods require an initial guess, which affects the quality of their output. We compare our method to an unbiased surface derivative method (SDF reparameterization [Vicini et al. 2022]) using an initialization with a sphere that is either *bigger* (middle) or *smaller* (bottom) than the overall size of the target. The experiment shows that the method struggles to reconstruct the target from the smaller sphere, while carving it out of the bigger sphere seems more reliable. Our method (top) can optimize without such assumptions, i.e., starting from empty space.

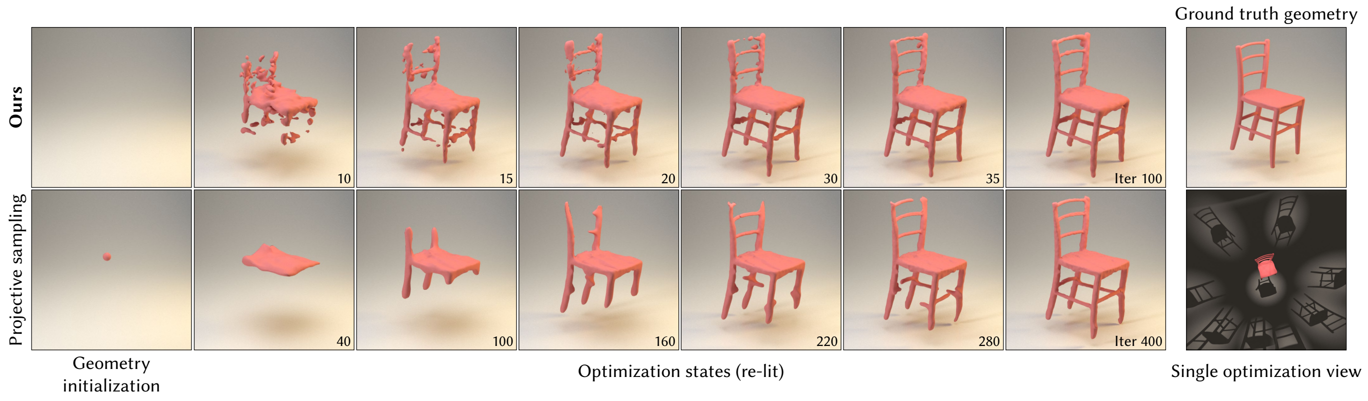


Fig. 9. **The benefit of simultaneously optimizing all positions.** We replicate an experiment of Zhang et al. [2023a] with our method to reconstruct a chair from a single reference image. The baseline employs preconditioned gradient descent [Nicolet et al. 2021] to locally deform a triangle mesh. In each iteration, the pixels rendering the legs propagate gradients to the scene, causing their progressive extrusion in a thin region of overlap between tentative object and the chair in the reference image. Derivatives outside of the region of overlap are discarded. Our method observes and uses all gradients starting from the very first iteration, enabling faster convergence.

Comparison with volume reconstructions. Figure 12 presents results and equal-iteration timings comparing our method to volume-based inversion. For the latter, we set the maximum path depth of the underlying volumetric path tracer to 3 interactions, as larger values significantly degrade performance without improving visual fidelity. All experiments use the same number of samples per pixel, and each optimization iteration uses all 12 optimization views. The anisotropic volume model uses the SGGX phase function [Heitz et al. 2015]. The volume models are initially as fast as ours but slow down significantly as the volume thickens, which is a consequence of iterative steps needed to resolve coupling between different parts of the exponential volume. Besides reducing speed, this coupling leads to degraded reconstruction quality at equal iteration count. The many-worlds approach is algorithmically simpler, produces a better result in less time, and directly outputs a mesh with materials that are ready to be relit, without the need for additional optimization to extract a surface BRDF from phase functions.

Timings. Table 1 lists the average computation time per gradient step and view for several scenes. These timings were measured on an AMD Ryzen 3970X Linux workstation with an NVIDIA RTX 3090 graphics card, which our implementation uses to accelerate ray tracing. We limited the maximum path depth for every scene to a reasonable value. For certain scenes, we also disabled gradient estimation for some path segments. For example, in the shape reconstructions inside a glass object, the first ray segment (from the camera to the glass interface) and the last ray segment cannot intersect the many-worlds representation.

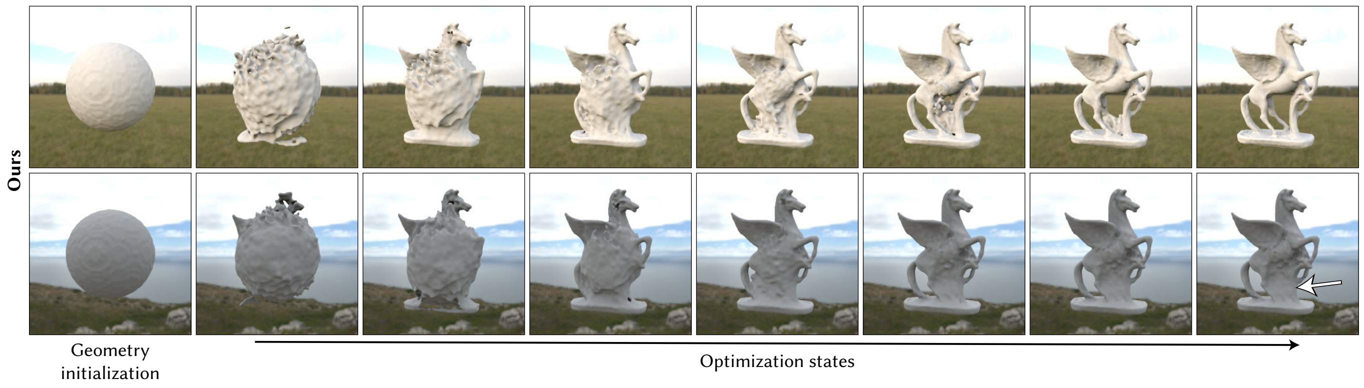


Fig. 10. **Limitation on handling interior topological changes.** Our method extends surface perturbations only to the exterior and does not accommodate interior topological changes — such as transforming a sphere into a donut. We demonstrate this using a sphere initialization under two lighting conditions. While shading derivatives can sometimes produce correct holes (top row), it could fail under alternative lighting conditions (bottom row).

6 Conclusion and future work

Correct handling of discontinuous visibility during physically based inversion of geometry has presented a formidable challenge in recent years. Instead of proposing yet another method to solve this challenging problem, many-worlds inverse rendering shows that there are completely different ways to approach it. Using a notion of non-local perturbations of a surface, our method successfully synthesizes complex geometries from an initially empty scene.

Our work lays the theoretical foundation and validates this theory with a first implementation. However, the details of this implementation are still far from optimal and could benefit from various enhancements. For example, we derive the local orientation from a field that also governs occupancy, which is straightforward but also introduces a difficult-to-optimize nonlinear coupling. Extending the model with a distribution of orientations would further allow it to consider multiple conflicting explanations at every point.

Reconstructing an object involves a balance between *exploration* to consider alternative explanations in S and *exploitation* to refine parameters of the current explanation $\bar{S}$. Our method aggressively pursues the exploration phase using a uniform sampling strategy but lacks a mechanism to effectively exploit its knowledge. We find that it often reconstructs a good approximation of a complex shape in as little as 20 iterations, impossibly fast compared to existing methods, but then requires 500 iterations for the seemingly trivial task of smoothing out little kinks.

Our implementation of the many-worlds derivative is based on a standard physically based path tracer, but previous works on differentiable rendering have shown the benefits of moving derivative computation into a separate phase using a *local formulation*. This phase starts the Monte Carlo sampling process where derivatives locally emerge (e.g., at edges of a triangle mesh in the case of visibility discontinuities). Integrating the local form of our model with occupancy-based sampling could refine the background surface with a more targeted optimization of its close neighborhood.

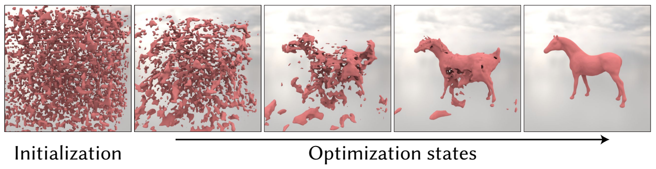


Fig. 11. **Subtractive changes.** Many-worlds derivatives match surface derivatives when sampled close to the surface. We initialize the scene with dense geometry to demonstrate the robustness of our method against subtractive changes.

Table 1. We measure the average time needed to optimize one 512^2 pixel image for different scenes. The reported time covers all overheads including the primal rendering pass, the uncorrelated derivative propagation pass, the optimizer step and scene update.

	Path depth	AD depth	spp	grad spp	time (s)
Heptoroid	2	2	128	32	0.41
Dragon	2	2	128	32	0.42
Deer	4	2	128	32	0.75
Polyhedra	4	2	64	32	1.10
Neptune	5	3	256	32	2.56
Fertility	5	3	256	32	2.06

Acknowledgments

This project has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation program (grant agreement No 948846).

References

Sai Bangaru, Michael Gharbi, Tzu-Mao Li, Fujun Luan, Kalyan Sunkavalli, Milos Hasan, Sai Bi, Zexiang Xu, Gilbert Bernstein, and Fredo Durand. 2022. Differentiable Rendering of Neural SDFs through Reparameterization. In *ACM SIGGRAPH Asia 2022 Conference Proceedings* (Daegu, Republic of Korea) *(SIGGRAPH Asia '22)*. Association for Computing Machinery, New York, NY, USA, Article 22, 9 pages. doi:10.1145/3550469.3555397

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.

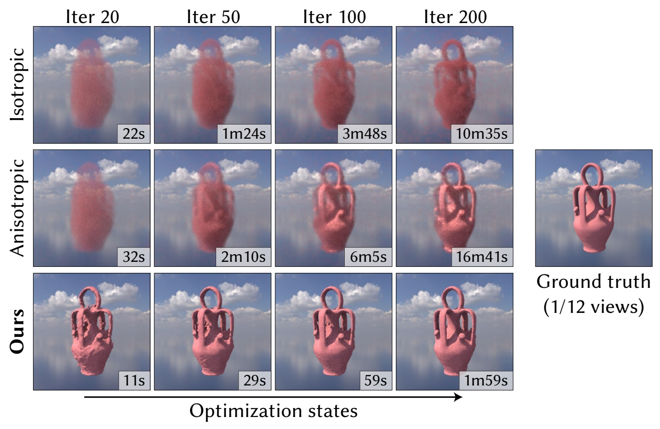


Fig. 12. **Comparison with volume reconstructions.** We compare reconstruction results using exponential volume inversion and the proposed many-worlds approach. Isotropic volumes (top) cannot replicate the surface appearance, while anisotropic micro-flakes (middle) perform somewhat better. However, simulating for interactions between worlds (e.g. to model exponential attenuation) introduces substantial computational overheads in both volume reconstruction methods. Extracting the final BSDF and surface from the volume poses additional challenges. Our method (bottom) produces a high-fidelity surface reconstruction in a fraction of the time.

Benedikt Bitterli, Srinath Ravichandran, Thomas Müller, Magnus Wrenninge, Jan Novák, Steve Marschner, and Wojciech Jarosz. 2018. A radiative transfer framework for non-exponential media. *ACM Trans. Graph.* 37, 6, Article 225 (Dec. 2018), 17 pages. doi:10.1145/3272127.3275103

Mark Boss, Raphael Braun, Varun Jampani, Jonathan T Barron, Ce Liu, and Hendrik Lensch. 2021. Nerd: Neural reflectance decomposition from image collections. In *Proceedings of the IEEE/CVF International Conference on Computer Vision.* 12684–12694.

Jonathan Dupuy, Eric Heitz, and Eugene d'Eon. 2016. Additional Progress Towards the Unification of Microfacet and Microflake Theories. In *EGSR (EI&I).* 55–63.

Michael Fischer and Tobias Ritschel. 2023. Plateau-Reduced Differentiable Path Tracing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition.* 4285–4294.

Eric Heitz, Jonathan Dupuy, Cyril Crassin, and Carsten Dachsbacher. 2015. The SGGX microflake distribution. *ACM Transactions on Graphics (TOG)* 34, 4 (2015), 1–11.

Eric Heitz, Johannes Hanika, Eugene d'Eon, and Carsten Dachsbacher. 2016. Multiple-scattering microfacet BSDFs with the Smith model. *ACM Transactions on Graphics (TOG)* 35, 4 (2016), 1–14.

Wenzel Jakob, Adam Arbree, Jonathan T. Moon, Kavita Bala, and Steve Marschner. 2010. A radiative transfer framework for rendering materials with anisotropic structure. *ACM Trans. Graph.* 29, 4, Article 53 (jul 2010), 13 pages. doi:10.1145/1778765.1778790

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

Haian Jin, Isabella Liu, Peijia Xu, Xiaoshuai Zhang, Songfang Han, Sai Bi, Xiaowei Zhou, Zexiang Xu, and Hao Su. 2023. Tensoir: Tensorial inverse rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition.* 165–174.

Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 2023. 3D Gaussian Splatting for Real-Time Radiance Field Rendering. *ACM Transactions on Graphics* 42, 4 (July 2023).

Diederik P. Kingma and Jimmy Ba. 2014. Adam: A Method for Stochastic Optimization. *CoRR* abs/1412.6980 (2014). https://api.semanticscholar.org/CorpusID:6628106

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* 39, 6 (2020).

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.

Zhaoshuo Li, Thomas Müller, Alex Evans, Russell H Taylor, Mathias Unberath, Ming-Yu Liu, and Chen-Hsuan Lin. 2023. Neuralangelo: High-fidelity neural surface reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition.* 8456–8465.

William E. Lorensen and Harvey E. Cline. 1987. Marching cubes: A high resolution 3D surface construction algorithm. *SIGGRAPH Comput. Graph.* 21, 4 (aug 1987), 163–169. doi:10.1145/37402.37422

Guillaume Loubet, Nicolas Holzschuch, and Wenzel Jakob. 2019. Reparameterizing Discontinuous Integrands for Differentiable Rendering. *Transactions on Graphics (Proceedings of SIGGRAPH Asia)* 38, 6 (Dec. 2019). doi:10.1145/3355089.3356510

Guillaume Loubet and Fabrice Neyret. 2018. A new microflake model with microscopic self-shadowing for accurate volume downsampling. *Computer Graphics Forum* 37, 2 (2018), 111–121.

Ishit Mehta, Manmohan Chandraker, and Ravi Ramamoorthi. 2023. A Theory of Topological Derivatives for Inverse Rendering of Geometry. In *Proceedings of the IEEE/CVF International Conference on Computer Vision.* 419–429.

Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. 2019. Occupancy networks: Learning 3d reconstruction in function space. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition.* 4460–4470.

Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. 2021. NeRF: representing scenes as neural radiance fields for view synthesis. *Commun. ACM* 65, 1 (dec 2021). doi:10.1145/3503250

Bailey Miller, Hanyu Chen, Alice Lai, and Ioannis Gkioulekas. 2024. Objects as Volumes: A Stochastic Geometry View of Opaque Solids. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR).* 87–97.

Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. 2022. Instant neural graphics primitives with a multiresolution hash encoding. *ACM Trans. Graph.* 41, 4, Article 102 (July 2022), 15 pages. doi:10.1145/3528223.3530127

Baptiste Nicolet, Alec Jacobson, and Wenzel Jakob. 2021. Large Steps in Inverse Rendering of Geometry. *ACM Transactions on Graphics (Proceedings of SIGGRAPH Asia)* 40, 6 (Dec. 2021).

Michael Niemeyer, Lars Mescheder, Michael Oechsle, and Andreas Geiger. 2020. Differentiable volumetric rendering: Learning implicit 3d representations without 3d supervision. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition.* 3504–3515.

Merlin Nimier-David, Thomas Müller, Alexander Keller, and Wenzel Jakob. 2022. Unbiased Inverse Volume Rendering with Differential Trackers. *ACM Trans. Graph.* 41, 4, Article 44 (July 2022), 20 pages. doi:10.1145/3528223.3530073

Merlin Nimier-David, Sébastien Speierer, Benoît Ruiz, and Wenzel Jakob. 2020. Radiative Backpropagation: An Adjoint Method for Lightning-Fast Differentiable Rendering. *Transactions on Graphics (Proceedings of SIGGRAPH)* 39, 4 (July 2020). doi:10.1145/3386569.3392406

Silvia Sellán and Alec Jacobson. 2022. Stochastic Poisson surface reconstruction. *ACM Transactions on Graphics (TOG)* 41, 6 (2022), 1–12.

Silvia Sellán and Alec Jacobson. 2023. Neural Stochastic Screened Poisson Reconstruction. *ACM Transactions on Graphics (Proc. SIGGRAPH Asia)* (2023).

Dario Seyb, Eugene d'Eon, Benedikt Bitterli, and Wojciech Jarosz. 2024. From microfacets to participating media: A unified theory of light transport with stochastic geometry. *ACM Transactions on Graphics (TOG)* 43, 4 (2024), 1–17.

J Kenneth Shultis and RB Myneni. 1988. Radiative transfer in vegetation canopies with anisotropic scattering. *Journal of Quantitative Spectroscopy and Radiative Transfer* 39, 2 (1988), 115–129.

Jos Stam and Ryan Schmidt. 2011. On the velocity of an implicit surface. *ACM Transactions on Graphics (TOG)* 30, 3 (2011), 1–7.

Eric Veach. 1997. *Robust Monte Carlo Methods for Light Transport Simulation.* Ph. D. Dissertation. Stanford University, Stanford, CA.

Delio Vicini, Wenzel Jakob, and Anton Kaplanyan. 2021a. A non-exponential transmittance model for volumetric scene representations. *ACM Transactions on Graphics (TOG)* 40, 4 (2021), 1–16.

Delio Vicini, Sébastien Speierer, and Wenzel Jakob. 2021b. 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. doi: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.

Peng Wang, Lingjie Liu, Yuan Liu, Christian Theobalt, Taku Komura, and Wenping Wang. 2021. NeuS: Learning Neural Implicit Surfaces by Volume Rendering for Multi-view Reconstruction. In *Advances in Neural Information Processing Systems*, M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan (Eds.), Vol. 34. Curran Associates, Inc., 27171–27183. https://proceedings.neurips.cc/paper_files/paper/2021/file/e41e164f7485ec4a28741a2d0ea41c74-Paper.pdf

Zichen Wang, Xi Deng, Ziyi Zhang, Wenzel Jakob, and Steve Marschner. 2024. A Simple Approach to Differentiable Rendering of SDFs. In *SIGGRAPH Asia 2024 Conference Papers (SA '24).* Association for Computing Machinery, New York, NY, USA, Article 119, 11 pages. doi:10.1145/3680528.3687573

Oliver Williams and Andrew Fitzgibbon. 2006. Gaussian process implicit surfaces. In *Gaussian Processes in Practice.*

Peiyu Xu, Sai Bangaru, Tzu-Mao Li, and Shuang Zhao. 2023. Warped-area reparameterization of differential path integrals. *ACM Transactions on Graphics (TOG)* 42, 6

(2023), 1–18.
Peiyu Xu, Sai Bangaru, Tzu-Mao Li, and Shuang Zhao. 2024. Markov-Chain Monte Carlo Sampling of Visibility Boundaries for Differentiable Rendering. In *SIGGRAPH Asia 2024 Conference Papers* (Tokyo, Japan) *(SA '24)*. Association for Computing Machinery, New York, NY, USA, Article 118, 11 pages. doi:10.1145/3680528.3687622
Kai Yan, Christoph Lassner, Brian Budge, Zhao Dong, and Shuang Zhao. 2022. Efficient estimation of boundary integrals for path-space differentiable rendering. *ACM Transactions on Graphics (TOG)* 41, 4 (2022), 1–13.
Lior Yariv, Jiatao Gu, Yoni Kasten, and Yaron Lipman. 2021. Volume rendering of neural implicit surfaces. In *Thirty-Fifth Conference on Neural Information Processing Systems*.
Tizian Zeltner, Sébastien Speierer, Iliyan Georgiev, and Wenzel Jakob. 2021. Monte Carlo Estimators for Differential Light Transport. *Transactions on Graphics (Proceedings of SIGGRAPH)* 40, 4 (Aug. 2021). doi:10.1145/3450626.3459807
Cheng Zhang, Bailey Miller, Kai Yan, Ioannis Gkioulekas, and Shuang Zhao. 2020. Path-space differentiable rendering. *ACM Trans. Graph.* 39, 4, Article 143 (Aug. 2020), 19 pages. doi:10.1145/3386569.3392383
Kai Zhang, Fujun Luan, Qianqian Wang, Kavita Bala, and Noah Snavely. 2021a. PhySG: Inverse Rendering with Spherical Gaussians for Physics-based Material Editing and Relighting. In *The IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.
Xiuming Zhang, Pratul P. Srinivasan, Boyang Deng, Paul Debevec, William T. Freeman, and Jonathan T. Barron. 2021b. NeRFactor: neural factorization of shape and reflectance under an unknown illumination. *ACM Trans. Graph.* 40, 6, Article 237 (Dec. 2021), 18 pages. doi:10.1145/3478513.3480496
Youjia Zhang, Teng Xu, Junqing Yu, Yuteng Ye, Junle Wang, Yanqing Jing, Jingyi Yu, and Wei Yang. 2023b. NeMF: Inverse Volume Rendering with Neural Microflake Field. *2023 IEEE/CVF International Conference on Computer Vision (ICCV)* (2023).
Ziyi Zhang, Nicolas Roussel, and Wenzel Jakob. 2023a. Projective Sampling for Differentiable Rendering of Geometry. *Transactions on Graphics (Proceedings of SIGGRAPH Asia)* 42, 6 (Dec. 2023). doi:10.1145/3618385
Ziyi Zhang, Nicolas Roussel, Thomas Müller, Tizian Zeltner, Merlin Nimier-David, Fabrice Rousselle, and Wenzel Jakob. 2025. Radiance Surfaces: Optimizing Surface Representations with a 5D Radiance Field Loss. *Proceedings of the ACM SIGGRAPH Conference Papers (SIGGRAPH Conference Papers '25)* (2025), 10 pages. doi:10.1145/3721238.3730713

A Extended surface derivatives

Our method extends surface optimization by introducing non-local perturbations to a surface $\bar{S}$. Along each light path, the method tests how potential surfaces *could* exist at sampled positions as perturbations to $\bar{S}$. By optimizing these hypothetical surfaces, we effectively explore an extended parameter space S to refine $\bar{S}$.

Previous work [Zhang et al. 2020, Zhang et al. 2023a] established the *local formulation* of surface derivatives, which measures how an infinitesimal surface change affects radiative transport in the entire scene. We extend their formulation to measure how infinitesimal changes in *any* hypothetical surface patch within S influence radiative transport. This provides a quantitative way to rederive our method. In this section, we show that the resulting derivatives match the many-worlds derivative transport.

This analysis leads to two critical results:

- Many-worlds derivatives reduce to conventional surface derivatives (shading and boundary) when evaluated on $\bar{S}$.
- We unify shading and boundary derivatives into a single term in the extended domain, enabling an algorithmically simpler implementation.

A.1 Conventional surface derivatives

In physically based rendering, surface differentiation involves two types of derivatives: the *shading derivative* and the *boundary derivative*. The shading derivative applies to the entire visible surface for a given viewpoint, primarily affecting appearance through the shading normal and local material properties. In contrast, the boundary derivative (corresponding to "occlusion" in Figure 1) is defined only along the visibility silhouette curve as seen from a viewpoint. It adjusts the object's visibility contour and is the main driver of shape optimization.

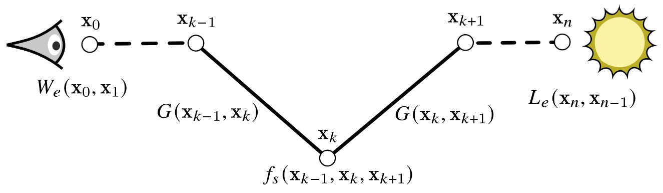

$\mathbf{x}_0$
$\mathbf{x}_{k-1}$
$\mathbf{x}_{k+1}$
$\mathbf{x}_n$
$W_e(\mathbf{x}_0, \mathbf{x}_1)$
$L_e(\mathbf{x}_n, \mathbf{x}_{n-1})$
$G(\mathbf{x}_{k-1}, \mathbf{x}_k)$
$G(\mathbf{x}_k, \mathbf{x}_{k+1})$
$\mathbf{x}_k$
$f_s(\mathbf{x}_{k-1}, \mathbf{x}_k, \mathbf{x}_{k+1})$

Fig. 13. A light path with n segments connecting the camera to the emitter.

Both derivatives can be expressed as integrals over path space, a high-dimensional domain encompassing all possible light paths. To simplify our derivation, we write them in the *three-point form* ([Veach 1997, Section 8.1]) that isolates the derivative contribution from a single ray segment. While the integral remains high-dimensional, this segment-specific form encapsulates most of the dimensions within the incident radiance term L_i and the incident importance term W_i, making the formulation more tractable.

A.1.1 Shading derivative. Consider the radiance contribution of a n-segment light path as illustrated in Figure 13. The measurement contribution function is given by:

$$f(\bar{\mathbf{x}}) = W_e(\mathbf{x}_0, \mathbf{x}_1)G(\mathbf{x}_0, \mathbf{x}_1) \left[\prod_{i=1}^{n-1} f_s(\mathbf{x}_{i-1}, \mathbf{x}_i, \mathbf{x}_{i+1})G(\mathbf{x}_i, \mathbf{x}_{i+1})\right] L_e(\mathbf{x}_n, \mathbf{x}_{n-1}), \tag{9}$$

where f_s is the BSDF function, G is the standard geometric term including visibility $\mathcal{V}$, and L_e and W_e refer to the emitted radiance and importance. This equation contributes to a measurement—for instance an image pixel color I—following the path space integral [Veach 1997]:

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

where $\mathrm{d}\mu = \prod_{i=0}^{n} \mathrm{d}A(\mathbf{x}_i)$ is the area product element.

Without loss of generality, we assume that the emitted radiance L_e and the camera model do not depend on the scene parameter θ (i.e., detached). Differentiation of the contribution function with respect to θ yields:

$$\frac{\partial f(\bar{\mathbf{x}})}{\partial \theta} = \sum_{k=1}^{n-1} W_i(\mathbf{x}_{\mathrm{k}-1}, \mathbf{x}_{\mathrm{k}})G(\mathbf{x}_{\mathrm{k}-1}, \mathbf{x}_{\mathrm{k}}) \partial_\theta \Big[f_s(\mathbf{x}_{\mathrm{k}-1}, \mathbf{x}_{\mathrm{k}}, \mathbf{x}_{\mathrm{k}+1})G(\mathbf{x}_{\mathrm{k}}, \mathbf{x}_{\mathrm{k}+1})\Big] L_i(\mathbf{x}_{k+1}, \mathbf{x}_k), \tag{10}$$

where functions L_i and W_i denote incident radiance and importance:

$$L_i(\mathbf{x}_{k+1}, \mathbf{x}_k) = L_e(\mathbf{x}_n, \mathbf{x}_{n-1})\Pi_{i=k+1}^{n-1}[G(\mathbf{x}_i, \mathbf{x}_{i+1})f_s(\mathbf{x}_{i-1}, \mathbf{x}_i, \mathbf{x}_{i+1})],$$
$$W_i(\mathbf{x}_{k-1}, \mathbf{x}_k) = W_e(\mathbf{x}_0, \mathbf{x}_1)\Pi_{i=1}^{k-1}[G(\mathbf{x}_{i-1}, \mathbf{x}_i)f_s(\mathbf{x}_{i-1}, \mathbf{x}_i, \mathbf{x}_{i+1})].$$

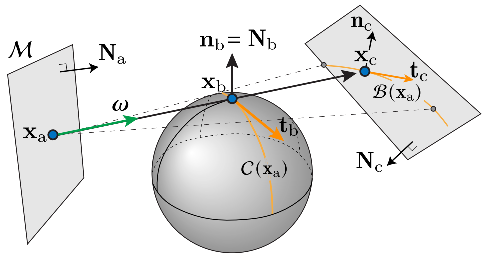


Fig. 14. **Surface boundary derivative**. The boundary derivative contains a motion term that tracks how fast the boundary segment $(\mathbf{x}_\mathrm{a}, \mathbf{x}_\mathrm{c})$ moves along the normal direction $\mathbf{n}_\mathrm{b}$, which is orthogonal to the viewing direction $\boldsymbol{\omega}$ and the silhouette curve direction $\mathbf{t}_\mathrm{b}$.

This gives the three-point form shading derivative of I:

$$\begin{aligned}\left[\frac{\partial I}{\partial \theta}\right]_s &= \int_\Omega \partial_\theta f(\bar{\mathbf{x}})\, \mathrm{d}\mu \\ &= \int_{\mathcal{M}\times\mathcal{M}} W_i(\mathbf{x}_{k-1}, \mathbf{x}_k)\, G(\mathbf{x}_{k-1}, \mathbf{x}_k) \\ &\quad \underbrace{\int_{\mathcal{M}} \partial_\theta \Big[G(\mathbf{x}_\mathrm{k}, \mathbf{x}_\mathrm{k+1}) f_s(\mathbf{x}_\mathrm{k-1}, \mathbf{x}_\mathrm{k}, \mathbf{x}_\mathrm{k+1}) \Big] L_i(\mathbf{x}_\mathrm{k+1}, \mathbf{x}_\mathrm{k})\ \mathrm{d}A(\mathbf{x}_\mathrm{k+1})}_{\partial_\theta \widehat{L_o}(\mathbf{x}_\mathrm{k}, \mathbf{x}_\mathrm{k-1})} \\ &\quad \mathrm{d}A(\mathbf{x}_\mathrm{k})\, \mathrm{d}A(\mathbf{x}_\mathrm{k-1}). \end{aligned} \tag{11}$$

The term in the curly bracket is abbreviated as $\partial_\theta \widehat{L_o}(\mathbf{x}_\mathrm{k}, \mathbf{x}_\mathrm{k-1})$, which captures the derivative stemming from only the *current interaction* at $\mathbf{x}_\mathrm{k}$. The derivatives from later interactions along the path are incorporated in their respective three-point form integrals.

A.1.2 Boundary derivative. We start from Equation (43) in Zhang et al.'s work [2020] to derive the segment-specific boundary derivative integral. The derivation is similar to the one shown in the supplementary material of [Zhang et al. 2023a] with a different objective—we aim to express the integral such that it is local to the boundary segment, rather than to the silhouette point. One can alternatively derive the same result starting from Equation (4) in [Zhang et al. 2023a].

The original boundary derivative integral, without reparameterization, states that (see Figure 14 for illustration):

$$\left[\frac{\partial I}{\partial \theta}\right]_b = \int_{\mathcal{M}} \int_{\mathcal{B}(\mathbf{x}_\mathrm{a})} L_d(\mathbf{x}_\mathrm{b}, \mathbf{x}_\mathrm{a})\, G(\mathbf{x}_\mathrm{a}, \mathbf{x}_\mathrm{c})\, W_i(\mathbf{x}_\mathrm{a}, \mathbf{x}_\mathrm{c}) \quad (\partial_\theta \mathbf{x}_\mathrm{c} \cdot \mathbf{n}_\mathrm{c})\ \mathrm{d}l(\mathbf{x}_\mathrm{c})\, \mathrm{d}A(\mathbf{x}_\mathrm{a}), \tag{12}$$

where $(\mathbf{x}_\mathrm{a}, \mathbf{x}_\mathrm{c})$ form a boundary segment that makes contact with the boundary point $\mathbf{x}_\mathrm{b}$, and the domain $\mathcal{B}(\mathbf{x}_\mathrm{a})$ is the boundary curve on the $\mathbf{x}_\mathrm{c}$ side surface as seen from $\mathbf{x}_\mathrm{a}$. The function L_d defines radiance difference between the foreground and the background as $L_d(\mathbf{x}_\mathrm{b}, \mathbf{x}_\mathrm{a}) = L_o(\mathbf{x}_\mathrm{b}, \boldsymbol{\omega}) - L_i(\mathbf{x}_\mathrm{b}, -\boldsymbol{\omega})$. The inner product measures the motion of $\mathbf{x}_\mathrm{c}$ along its in-surface normal direction $\mathbf{n}_\mathrm{c}$.

This integral is already specific to a ray segment, but the motion term $\partial_\theta \mathbf{x}_\mathrm{c} \cdot \mathbf{n}_\mathrm{c}$ is a derived quantity from $\partial_\theta \mathbf{x}_\mathrm{b} \cdot \mathbf{n}_\mathrm{b}$. They are related as follows [Zhang et al. 2023a, Supplementary Eq. (9)]:

$$\frac{\partial_\theta \mathbf{x}_\mathrm{c} \cdot \mathbf{n}_\mathrm{c}}{\partial_\theta \mathbf{x}_\mathrm{b} \cdot \mathbf{n}_\mathrm{b}} = \frac{l_\mathrm{ac}}{l_\mathrm{ab} \|\mathbf{n}_\mathrm{b} \times \mathbf{N}_\mathrm{c}\|}.$$

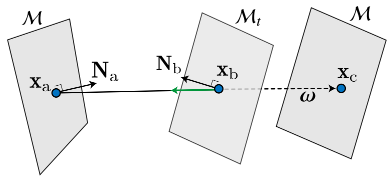


Fig. 15. **Shading derivatives in many-worlds transport**. Along a ray segment $(\mathbf{x}_\mathrm{a}, \mathbf{x}_\mathrm{c})$, we consider potential interactions with a hypothetical surface at $\mathbf{x}_\mathrm{b}$ as a perturbation to the background surface. The shading derivatives from all such interactions along the ray segment are accumulated.

We also change the length measure $\mathrm{d}l(\mathbf{x}_\mathrm{c})$ to $\mathrm{d}l(\mathbf{x}_\mathrm{b})$ [Zhang et al. 2023a, Supplementary Eq. (3)]:

$$\frac{\mathrm{d}l(\mathbf{x}_\mathrm{c})}{\mathrm{d}l(\mathbf{x}_\mathrm{b})} = \frac{l_\mathrm{ac} \|\boldsymbol{\omega} \times \mathbf{t}_\mathrm{b}\|}{l_\mathrm{ab} \|\boldsymbol{\omega} \times \mathbf{t}_\mathrm{c}\|}.$$

Futhermore, we use the following identity ([Zhang et al. 2023a, Supplementary Eq. (10)]):

$$\|\boldsymbol{\omega} \times \mathbf{t}_\mathrm{c}\|\, \|\mathbf{n}_\mathrm{b} \times \mathbf{N}_\mathrm{c}\| = |\boldsymbol{\omega} \cdot \mathbf{N}_\mathrm{c}|.$$

Combining these equations, we obtain:

$$\left[\frac{\partial I}{\partial \theta}\right]_b = \int_{\mathcal{M}} \int_{C(\mathbf{x}_\mathrm{a})} L_d(\mathbf{x}_\mathrm{b}, \mathbf{x}_\mathrm{a})\, \frac{|\boldsymbol{\omega} \cdot \mathbf{N}_\mathrm{a}|\, \|\boldsymbol{\omega} \times \mathbf{t}_\mathrm{b}\|}{l_\mathrm{ab}^2}\, W_i(\mathbf{x}_\mathrm{a}, \mathbf{x}_\mathrm{b}) \quad (\partial_\theta \mathbf{x}_\mathrm{b} \cdot \mathbf{n}_\mathrm{b})\ \mathrm{d}l(\mathbf{x}_\mathrm{b})\, \mathrm{d}A(\mathbf{x}_\mathrm{a}), \tag{13}$$

where the boundary domain $C(\mathbf{x}_\mathrm{a})$ is the visibility silhouette curve on the occluder as seen from $\mathbf{x}_\mathrm{a}$. Rewriting this result by converting the length measure from scene space to hemisphere space, we get:

$$\left[\frac{\partial I}{\partial \theta}\right]_b = \int_{\mathcal{M}} \int_{C(\mathbf{x}_\mathrm{a})} L_d(\mathbf{x}_\mathrm{b}, -\boldsymbol{\omega})\, W_i(\mathbf{x}_\mathrm{a}, \boldsymbol{\omega})\, |\boldsymbol{\omega} \cdot \mathbf{N}_\mathrm{a}| \quad (\partial_\theta \boldsymbol{\omega} \cdot \mathbf{n}_\mathrm{b})\ \mathrm{d}l(\boldsymbol{\omega})\, \mathrm{d}A(\mathbf{x}_\mathrm{a}). \tag{14}$$

Intuitively, the motion term $\partial_\theta \boldsymbol{\omega} \cdot \mathbf{n}_\mathrm{b}$ measures how rapidly the occluder moves in the direction perpendicular to the viewing ray. This motion is weighted by the radiance difference between the foreground and background, and the derivative arising from this motion is transported to the sensor akin to regular radiance [Nimier-David et al. 2020, Section 3.1].

A.2 Many-worlds derivatives

We now extend surface derivatives into the extended domain to also quantify how *potential surface patches* along a ray segment affect the radiative transport.

A.2.1 Shading derivative. We write the shading derivative as an integral over potential surface patches along the ray segment $(\mathbf{x}_\mathrm{a}, \mathbf{x}_\mathrm{c})$, weighted by the probability of each patch's existence:

$$\int_0^{l_\mathrm{ac}} \alpha(t) \left[\frac{\partial I}{\partial \theta}\right]_s \mathrm{d}t \overset{(11)}{=} \int_0^{l_\mathrm{ac}} \int_{\mathcal{M}\times\mathcal{M}_t} \alpha(t) \Big[W_i(\mathbf{x}_\mathrm{a}, \mathbf{x}_\mathrm{b})\, G(\mathbf{x}_\mathrm{a}, \mathbf{x}_\mathrm{b})\, \partial_\theta \widehat{L_o}(\mathbf{x}_\mathrm{b}, \mathbf{x}_\mathrm{a}) \Big]\ \mathrm{d}A(\mathbf{x}_\mathrm{b})\, \mathrm{d}A(\mathbf{x}_\mathrm{a})\, \mathrm{d}t, \tag{15}$$

where $\mathbf{x}_\mathrm{b}$ is on a hypothetical surface $\mathcal{M}_t$ (Figure 15). Note that the ray direction may not be perpendicular to the surface normal

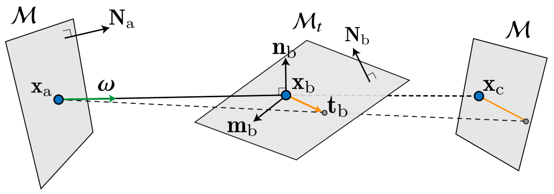


Fig. 16. **Extended form of the surface boundary derivative**. For conventional boundary derivatives, the silhouette direction $\mathbf{t}_b$ is only defined on the visibility silhouette. In the extended case, we define it for any surface, allowing silhouette occlusion to be interpreted as probabilistic surface existence.

$\mathrm{d}A(\mathbf{x}_b)\,\mathrm{d}t = \mathrm{d}V(\mathbf{x}_b)/|\boldsymbol{\omega}\cdot\mathbf{N}_b|$, allowing us to write the integral as:

$$= \int_{\mathcal{R}^3}\int_{\mathcal{M}} \alpha(\mathbf{x}_b)\, W_i(\mathbf{x}_a,\mathbf{x}_b) \frac{|\boldsymbol{\omega}\cdot\mathbf{N}_a|\mathcal{V}(\mathbf{x}_a\leftrightarrow\mathbf{x}_b)}{l_{ab}^2}\,\partial_\theta\widehat{L_o}(\mathbf{x}_b,\mathbf{x}_a)\ \mathrm{d}A(\mathbf{x}_a)\,\mathrm{d}V(\mathbf{x}_b). \tag{16}$$

By chaging from an area measure to a solid angle measure, we can cancel the remaining geometric term and absorb the visibility function $\mathcal{V}$ to obtain:

$$= \int_{\mathcal{R}^3}\int_{S^2} \alpha(\mathbf{x}_b)\, W_o(\mathbf{x}_b,-\boldsymbol{\omega})\,\partial_\theta\widehat{L_o}(\mathbf{x}_b,-\boldsymbol{\omega})\ \mathrm{d}\boldsymbol{\omega}\,\mathrm{d}V(\mathbf{x}_b). \tag{17}$$

This derivative has a simple form from which we can identify the following properties:

- It is local to the position $\mathbf{x}_b$ where the interaction happens.
- The shading derivative originating from this interaction is weighted by the probability of the surface's existence and is uniformly radiated in all directions to be received by the sensor.

A.2.2 Boundary derivative. A challenge in analyzing the boundary derivative for hypothetical surfaces is that existing theory from prior work only analyzes *visibility silhouettes* on occluders. This becomes problematic as a potential surface patch is not always on such a silhouette: its surface normal $\mathbf{N}_b$ may not be orthogonal to the viewing direction $\boldsymbol{\omega}$. Such a patch still influences radiative transport as it occludes radiance traveling from the background. However, this type of occlusion does not occur in local surface evolution, where occlusion is restricted to the visibility silhouette.

We therefore introduce an extended form of the boundary derivative whose value matches the conventional boundary derivative on a visibility silhouette curve, but is also well-defined for any surface.

Does such an extension make sense? Intuitively, the boundary derivative examines the color difference between the foreground and the background, and the motion term determines if more of the foreground or the background should be visible. This concept closely mirrors the derivative of *occupancy*, where adjusting the probability of the surface existence decides the visibility balance between the surface and the background. From this perspective, the "boundary" derivative logically extends to hypothetical surfaces with non-tangential ray intersections. Although "boundary" may be a misnomer in these cases, we retain the term for consistency with existing literature.

Extended form of surface boundary derivatives. For any surface located at $\mathbf{x}_b$ with normal $\mathbf{N}_b$, we define the *extended silhouette direction* $\mathbf{t}_b$ as the in-surface direction along which the dot product $\boldsymbol{\omega}\cdot\mathbf{N}_b$ remains constant. The conventional silhouette direction is a special case where $\boldsymbol{\omega}\cdot\mathbf{N}_b = 0$.

Let $\mathbf{n}_b$ be the viewing direction normal, which is orthogonal to both the viewing direction $\boldsymbol{\omega}$ and the extended silhouette direction $\mathbf{t}_b$ (Figure 16). In the extended case, it is important to distinguish between the surface normal $\mathbf{N}_b$ and the viewing direction normal $\mathbf{n}_b$. This subtle difference is not present in the conventional case, as $\mathbf{N}_b = \mathbf{n}_b$ on the visibility silhouette.

We denote the local motion at $\mathbf{x}_b$ as

$$\mathbf{v}_\theta(\mathbf{x}_b) := \partial_\theta\mathbf{x}_b\cdot\mathbf{N}_b \tag{18}$$

since it quantifies the rate at which occlusion changes at the interaction point.

Let $\mathbf{m}_b$ be a unit vector in the surface such that $\{\mathbf{t}_b,\mathbf{N}_b,\mathbf{m}_b\}$ forms an orthonormal basis. the motion relates to ray direction $\boldsymbol{\omega}$ as [Zhang et al. 2023a, Supplementary Eq. (9)]:

$$\partial_\theta\boldsymbol{\omega}\cdot\mathbf{n}_b = \frac{\|\mathbf{n}_b\times\mathbf{m}_b\|}{l_{ab}}\,\partial_\theta\mathbf{x}_b\cdot\mathbf{N}_b. \tag{19}$$

The length measure $\mathrm{d}l(\boldsymbol{\omega})$ is in the solid angle domain, and it relates to the length measure $\mathrm{d}l(\mathbf{x}_b)$ on the surface as

$$\mathrm{d}l(\boldsymbol{\omega}) = \frac{\|\boldsymbol{\omega}\times\mathbf{t}_b\|}{l_{ab}}\,\mathrm{d}l(\mathbf{x}_b). \tag{20}$$

Using these equations, the extended boundary derivative from Equation 14 can be rewritten as:

$$\begin{aligned}\left[\frac{\partial I}{\partial\theta}\right]_b &\overset{(14)}{=} \int_{\mathcal{M}}\int_{C(\mathbf{x}_a)} L_d(\mathbf{x}_b,-\boldsymbol{\omega})\,W_i(\mathbf{x}_a,\boldsymbol{\omega})\,|\boldsymbol{\omega}\cdot\mathbf{N}_a| \\ &\qquad (\partial_\theta\boldsymbol{\omega}\cdot\mathbf{n}_b)\ \mathrm{d}l(\boldsymbol{\omega})\,\mathrm{d}A(\mathbf{x}_a) \\ &\overset{(18,19)}{=} \int_{\mathcal{M}}\int_{C(\mathbf{x}_a)} L_d(\mathbf{x}_b,-\boldsymbol{\omega})\,W_i(\mathbf{x}_a,\boldsymbol{\omega})\,|\boldsymbol{\omega}\cdot\mathbf{N}_a| \\ &\qquad \frac{\|\mathbf{n}_b\times\mathbf{m}_b\|}{l_{ab}}\,\mathbf{v}_\theta(\mathbf{x}_b)\ \mathrm{d}l(\boldsymbol{\omega})\,\mathrm{d}A(\mathbf{x}_a) \\ &\overset{(20)}{=} \int_{\mathcal{M}}\int_{C(\mathbf{x}_a)} L_d(\mathbf{x}_b,-\boldsymbol{\omega})\,W_i(\mathbf{x}_a,\boldsymbol{\omega})\,|\boldsymbol{\omega}\cdot\mathbf{N}_a| \\ &\qquad \frac{\|\mathbf{n}_b\times\mathbf{m}_b\|\,\|\boldsymbol{\omega}\times\mathbf{t}_b\|}{l_{ab}^2}\,\mathbf{v}_\theta(\mathbf{x}_b)\ \mathrm{d}l(\mathbf{x}_b)\,\mathrm{d}A(\mathbf{x}_a).\end{aligned} \tag{21}$$

Boundary derivative in many-worlds transport. At each interaction position $\mathbf{x}_b$, let $\mathrm{d}l(\mathbf{x}_b)$ be the length measure along the extended silhouette direction $\mathbf{t}_b$ and $\mathrm{d}n(\mathbf{x}_b)$ be the length measure along the surface normal $\mathbf{N}_b$. Together, they define an area element $\mathrm{d}A(\mathbf{x}_b) = \mathrm{d}l(\mathbf{x}_b)\,\mathrm{d}n(\mathbf{x}_b)$ which quantifies how this patch occludes the background.

Analogous to shading derivatives, we write the many-worlds boundary derivative of I as an integral over all possible surface interactions along the ray segment $(\mathbf{x}_a,\mathbf{x}_c)$:

$$\iint\left[\frac{\partial I}{\partial\theta}\right]_b \mathrm{d}n\,\mathrm{d}t \overset{(21)}{=} \int_0^{l_{ac}}\int_{\mathcal{M}\times\mathcal{M}_t} L_d(\mathbf{x}_b,-\boldsymbol{\omega})\,W_i(\mathbf{x}_a,\boldsymbol{\omega}) \\ |\boldsymbol{\omega}\cdot\mathbf{N}_a|\,\frac{\|\mathbf{n}_b\times\mathbf{m}_b\|\,\|\boldsymbol{\omega}\times\mathbf{t}_b\|}{l_{ab}^2}\,\mathbf{v}_\theta(\mathbf{x}_b)\ \mathrm{d}A(\mathbf{x}_b)\,\mathrm{d}A(\mathbf{x}_a)\,\mathrm{d}t, \tag{22}$$

The basis $\{\mathbf{t}_\mathrm{b}, \mathbf{N}_\mathrm{b}, \boldsymbol{\omega}\}$ leads to a volume element

$$\mathrm{d}V(\mathbf{x}_\mathrm{b}) = |\boldsymbol{\omega} \cdot \mathbf{m}_\mathrm{b}| \;\mathrm{d}l(\mathbf{x}_\mathrm{b})\,\mathrm{d}n(\mathbf{x}_\mathrm{b})\,\mathrm{d}t. \tag{23}$$

Note that $\mathbf{m}_\mathrm{b}$, $\mathbf{n}_\mathrm{b}$ and $\mathbf{N}_\mathrm{b}$ are co-planar since they are all orthogonal to $\mathbf{t}_\mathrm{b}$. We therefore obtain an identity [Zhang et al. 2023a, Supplementary Eq. (10)]:

$$\|\boldsymbol{\omega} \times \mathbf{t}_\mathrm{b}\|\,\|\mathbf{n}_\mathrm{b} \times \mathbf{m}_\mathrm{b}\| = |\boldsymbol{\omega} \cdot \mathbf{m}_\mathrm{b}|. \tag{24}$$

Using these equations, we simplify the boundary derivative as:

$$\begin{aligned}(22) &\overset{(23)}{=} \int_{\mathcal{M}} \int_{\mathcal{R}^3} W_i(\mathbf{x}_\mathrm{a}, \boldsymbol{\omega})\, L_d(\mathbf{x}_\mathrm{b}, -\boldsymbol{\omega})\, \frac{|\boldsymbol{\omega} \cdot \mathbf{N}_\mathrm{a}|}{l_\mathrm{ab}^2} \\ &\qquad\qquad \frac{\|\boldsymbol{\omega} \times \mathbf{t}_\mathrm{b}\|\,\|\mathbf{n}_\mathrm{b} \times \mathbf{m}_\mathrm{b}\|}{|\boldsymbol{\omega} \cdot \mathbf{m}_\mathrm{b}|}\, \mathbf{v}_\theta(\mathbf{x}_\mathrm{b})\;\mathrm{d}V(\mathbf{x}_\mathrm{b})\,\mathrm{d}A(\mathbf{x}_\mathrm{a}) \\ &\overset{(24)}{=} \int_{\mathcal{R}^3} \int_{\mathcal{S}^2} W_o(\mathbf{x}_\mathrm{b}, -\boldsymbol{\omega})\, L_d(\mathbf{x}_\mathrm{b}, -\boldsymbol{\omega})\, \mathbf{v}_\theta(\mathbf{x}_\mathrm{b})\;\mathrm{d}\boldsymbol{\omega}\,\mathrm{d}V(\mathbf{x}_\mathrm{b}). \end{aligned} \tag{25}$$

We define the local motion $\mathbf{v}_\theta(\mathbf{x}_\mathrm{b})$ within its canonical space:

$$\mathbf{v}_\theta(\mathbf{x}_\mathrm{b}) = \partial_\theta \alpha(\mathbf{x}_\mathrm{b}). \tag{26}$$

This definition contrasts with prior work that computes the surface boundary derivative using an implicit field representation. In those works, the motion is defined in the scene space, resulting in the following Jacobian determinant [Stam and Schmidt 2011]:

$$\partial_\theta \mathbf{p} \cdot \mathbf{N} = \frac{\partial_\theta \alpha(\mathbf{p})}{\|\nabla \alpha(\mathbf{p})\|}, \tag{27}$$

that relates the normal velocity of a surface point $\mathbf{p}$ to the derivative of its implicit representation. This Jacobian determinant is not required since our method optimize the surface patch directly by modifying its occupancy value, rather than locally deforming it. The scaling factor in Equation 27 can bias the many-worlds derivative by erroneously considering neighboring occupancy values. This is more apparent when considering the fact that the gradient norm $\|\nabla\alpha\|$ could be infinitely large near the mean surface or the opposite, it could drop to zero in the case of an initialization where all positions share the same occupancy value.

This yields the final form of the boundary derivative in many-worlds transport:

$$(25) \overset{(26)}{=} \int_{\mathcal{R}^3} \int_{\mathcal{S}^2} W_o(\mathbf{x}_\mathrm{b}, -\boldsymbol{\omega})\, L_d(\mathbf{x}_\mathrm{b}, -\boldsymbol{\omega})\, \partial_\theta \alpha(\mathbf{x}_\mathrm{b})\;\mathrm{d}\boldsymbol{\omega}\,\mathrm{d}V(\mathbf{x}_\mathrm{b}). \tag{28}$$

A.2.3 Many-worlds derivative. Combining Equation 17 and Equation 28, we derive a derivative integral that captures the overall impact of non-local surface perturbations:

$$\begin{aligned}\frac{\partial I}{\partial \theta} &= \int_{\mathcal{R}^3} \int_{\mathcal{S}^2} W_o(\mathbf{x}_\mathrm{b}, -\boldsymbol{\omega}) \Big[\alpha(\mathbf{x}_\mathrm{b})\, \partial_\theta \widehat{L_o}(\mathbf{x}_\mathrm{b}, -\boldsymbol{\omega}) + \\ &\qquad\qquad \partial_\theta \alpha(\mathbf{x}_\mathrm{b})\, L_d(\mathbf{x}_\mathrm{b}, -\boldsymbol{\omega})\Big]\;\mathrm{d}\boldsymbol{\omega}\,\mathrm{d}V(\mathbf{x}_\mathrm{b}) \\ &= \int_{\mathcal{R}^3} \int_{\mathcal{S}^2} W_o(\mathbf{x}_\mathrm{b}, \boldsymbol{\omega})\, \partial_\theta \Big[L_d(\mathbf{x}_\mathrm{b}, \boldsymbol{\omega})\, \alpha(\mathbf{x}_\mathrm{b})\Big]\;\mathrm{d}\boldsymbol{\omega}\,\mathrm{d}V(\mathbf{x}_\mathrm{b}), \end{aligned} \tag{29}$$

where the background surface in the radiance difference function L_d is *detached.* Equation 29 represents the final form of the many-worlds derivative, expressed in an alternative domain but equivalent to Equation 6.

This derivation suggests that our approach is an extension of standard surface evolution methods. Conventional techniques evolve surfaces by locally modifying shading properties or adjusting geometry to change occlusion relationships, with such perturbations restricted to the surface itself. Instead, we generalize this concept by analyzing hypothetical surface patches in space, leading to the same well-defined shading and occlusion-based perturbations in an extended domain.

A key challenge in this extension is that any position along a ray can now contribute to surface evolution, rather than being restricted to a single intersection point. To address this, we adopt the "many-worlds" perspective: treating all possible interactions as mutually exclusive events. This ensures that a desired change is uniformly applied across all potential interactions.

B A random volume perspective

B.1 Transport in an exponential random volume

We begin by reviewing light transport in an exponential random volume as described by the radiative transfer equation. This equation gives the incident radiance L_i along a ray $(\mathbf{x}, \boldsymbol{\omega})$, accounting for radiative gains (in-scattering, emission) and losses (out-scattering, absorption) along the segment reaching up to the nearest ray-surface intersection at $s = \inf\{s' \mid \mathbf{x} + s'\boldsymbol{\omega} \in \mathcal{M}\}$ or $s = \infty$ if none exists:

$$L_\mathrm{i}(0) = \int_0^s T(t)\,[\mu_\mathrm{s}(t)\,L_\mathrm{s}(t) + \mu_\mathrm{a}(t)\,L_\mathrm{e}(t)]\,\mathrm{d}t + T(s)\,L_\mathrm{o}(s). \tag{30}$$

The first term models contributions of the volume, while the second accounts for a potential surface at $t = s$, whose outgoing radiance $L_\mathrm{o}(s)$ is attenuated by the medium.

The functions $\mu_\mathrm{a}(t)$ and $\mu_\mathrm{s}(t)$ specify the volume's *absorption* and *scattering* coefficient, whose sum gives the *extinction* $\mu_\mathrm{t} = \mu_\mathrm{a} + \mu_\mathrm{s}$. The in-scattered radiance

$$L_\mathrm{s}(t) = \int_{\mathcal{S}^2} L_i(t, \boldsymbol{\omega}')\, f_p(t, \boldsymbol{\omega}, \boldsymbol{\omega}')\,\mathrm{d}\boldsymbol{\omega}' \tag{31}$$

is the product integral of incident radiance and the *phase function* f_p. Finally, the *transmittance*

$$T(t) = \exp\left(-\int_0^t \mu_t(t')\,\mathrm{d}t'\right) \tag{32}$$

ties everything together: it establishes the connection to particle distributions and ensures energy conservation by accounting for self-shadowing (or more generally, self-extinction).

B.2 Many-worlds transport

Building on the discussion in Section 3.1, we aim to prevent nonsensical scattering and shadowing between multiple potential surfaces along a ray. This can be achieved by modeling the interaction with a *tenuous* volume, whose density is diluted by a factor of ε, which reduces the scattering and extinction coefficients μ_s and μ_t. This low amount of extinction ensures at most one interaction with the volume along a ray segment, leading to the following transmittance approximation:

$$T(t) \approx 1 - \int_0^t \mu_t(t')\,\mathrm{d}t', \tag{33}$$

which follows from $e^{-x} \approx 1 - x + O(x^2)$. Plugging this into the RTE (30) and discarding a second-order term ($\mu_\mathrm{s}\mu_\mathrm{t} \sim \varepsilon^2$) yields the approximation $L_{\mathrm{i}}(0) \approx L_{\mathrm{i}}^{\varepsilon}(0)$, where the latter is defined as

$$L_{\mathrm{i}}^{\varepsilon}(0) := \int_0^s \left[\mu_{\mathrm{s}}(t)\, L_{\mathrm{s}}(t) + \mu_{\mathrm{a}}(t)\, L_{\mathrm{e}}(t) - \mu_{\mathrm{t}}(t)\, L_{\mathrm{o}}(s)\right] \mathrm{d}t + L_o(s), \quad (34)$$

In other words, a tenuous volume is governed by a linearized RTE, where higher-order terms vanish. The superscript ε refers to quantities with a dilution factor ε.

To complete this model, we must instill meaning into the terms μ_{t}, μ_{s}, and f_p. Prior work often did so using volumetric analogs of microfacet surface models known as *microflakes* [Jakob et al. 2010]. Applications include optimization, volumetric level of detail, energy-conserving random walks, and neural fields [Heitz et al. 2015, 2016; Vicini et al. 2021a; Loubet and Neyret 2018; Zhang et al. 2023b].

It is worth noting that the theory of microflakes actually builds on a more general notion of *anisotropic radiative transport* [Shultis and Myneni 1988; Jakob et al. 2010], which adopts directionally varying extinction and scattering coefficients arising from a random distribution of oriented particles:

$$\mu_{\mathrm{t}}(\mathbf{x}, \boldsymbol{\omega}) = \rho(\mathbf{x}) \int_{S^2} \sigma(\boldsymbol{\omega}, \boldsymbol{\omega}')\, D(\mathbf{x}, \boldsymbol{\omega}')\, \mathrm{d}\boldsymbol{\omega}' \quad (35)$$

$$\mu_{\mathrm{s}}(\mathbf{x}, \boldsymbol{\omega}) = \rho(\mathbf{x}) \int_{S^2} a(\mathbf{x}, \boldsymbol{\omega}, \boldsymbol{\omega}')\, \sigma(\boldsymbol{\omega}, \boldsymbol{\omega}')\, D(\mathbf{x}, \boldsymbol{\omega}')\, \mathrm{d}\boldsymbol{\omega}', \quad (36)$$

where $\rho(\mathbf{x})$ is the particles' number density at $\mathbf{x}$, $D(\boldsymbol{\omega})$ models the density of their directional orientations, $\sigma(\boldsymbol{\omega}, \boldsymbol{\omega}')$ gives the cross-sectional area of a single particle with orientation $\boldsymbol{\omega}'$ observed from direction $\boldsymbol{\omega}$, and $a(\mathbf{x}, \boldsymbol{\omega}, \boldsymbol{\omega}')$ models the particles' scattering albedo ($\in [0, 1]$). Microflake theory can then be derived from these expressions by setting $\sigma(\boldsymbol{\omega}, \boldsymbol{\omega}') = \sigma\, |\boldsymbol{\omega} \cdot \boldsymbol{\omega}'|$ to model the projected area of a facet with surface area σ, and by constructing a phase function f_p based on the principle of specular reflection from such a flake.

However, we do not model a directional distribution $D(\boldsymbol{\omega})$ in this work. Instead, we adopt a far simpler particle model that associates *a single* particle orientation $\boldsymbol{\beta}(\mathbf{x})$ with every point $\mathbf{x}$ (Figure 3).

Since there is only one orientation at $\mathbf{x}$, the distribution D collapses to a Dirac delta function: $D(\mathbf{x}, \boldsymbol{\omega}) = \delta(\boldsymbol{\omega} - \boldsymbol{\beta}(\mathbf{x}))$, which in turn reduces the extinction to a simple product of number density and cross-sectional area

$$\mu_{\mathrm{t}}(\mathbf{x}, \boldsymbol{\omega}) = \rho(\mathbf{x})\, \sigma(\boldsymbol{\omega}, \boldsymbol{\beta}(\mathbf{x})). \quad (37)$$

Here, we do not specifically model the split into a separate number density and cross-section. Instead, we define an extinction function that directly evaluates their product:

$$\mu_{\mathrm{t}}(\mathbf{x}, \boldsymbol{\omega}) = \begin{cases} \varepsilon\, o(\mathbf{x}), & \text{if } \boldsymbol{\omega} \cdot \boldsymbol{\beta}(\mathbf{x}) > 0, \\ 0, & \text{otherwise}, \end{cases} \quad (38)$$

where $o(\mathbf{x})$ is the *occupancy*, i.e., the point-wise discrete probability that $\mathbf{x}$ is inside S (Equation 7). The branch condition encodes an important *nonreciprocal*[3] behavior: light interacting with a back-facing surface presents a nonsensical case, and this term masks those parts of the volume.

The extinction μ_{t} models the *stopping power* of the volume. For example, the flakes of a microflake volume obstruct light to a lesser degree when it propagates nearly parallel to them, and this manifests via a foreshortening term $|\boldsymbol{\omega} \cdot \boldsymbol{\omega}'|$ in the definition of μ_{t}. This is important to obtain a sensible (energy-conserving, reciprocal) model because these particles mutually interact. On the other hand, when focusing on a single world in a many-worlds representation, light stops with probability 1 when it encounters a surface regardless of how it is oriented (except for mentioned back-facing case). This is why our model does not include a similar foreshortening term. The term $\alpha(\mathbf{x})$ models the discrete probability of the world within the ensemble.

We define the *phase function* f_p of the volume as

$$f_p(\mathbf{x}, \boldsymbol{\omega}, \boldsymbol{\omega}') = \frac{f_s(\mathbf{x}, \boldsymbol{\omega}, \boldsymbol{\omega}')\, |\boldsymbol{\beta}(\mathbf{x}) \cdot \boldsymbol{\omega}'|}{a(\mathbf{x}, \boldsymbol{\omega}, \boldsymbol{\beta}(\mathbf{x}))},$$

which wraps f_s, the *bidirectional scattering distribution function* (BSDF) of the many-worlds surface at the point $\mathbf{x}$. This definition follows the standard convention that the phase function integrates to 1, with absorption handled by other terms. The albedo a provides the necessary normalization constant:

$$a(\mathbf{x}, \boldsymbol{\omega}, \boldsymbol{\beta}(\mathbf{x})) = \int_{S^2} f_s(\mathbf{x}, \boldsymbol{\omega}, \boldsymbol{\omega}')\, |\boldsymbol{\beta}(\mathbf{x}) \cdot \boldsymbol{\omega}'|\, \mathrm{d}\boldsymbol{\omega}'. \quad (39)$$

With these definitions, μ_{s} reduces to the product of extinction and a:

$$\mu_{\mathrm{s}}(\mathbf{x}, \boldsymbol{\omega}) = a(\mathbf{x}, \boldsymbol{\omega}, \boldsymbol{\beta}(\mathbf{x}))\, \mu_{\mathrm{t}}(\mathbf{x}, \boldsymbol{\omega}). \quad (40)$$

The product of this scattering coefficient and in-scattered radiance L_{s} (Equation 31) can be seen to compute an extinction-weighted integral of the many-worlds BSDF over projected solid angles:

$$\mu_{\mathrm{s}}(\mathbf{x}, \boldsymbol{\omega}) L_{\mathrm{s}}(\mathbf{x}, \boldsymbol{\omega}) = \mu_{\mathrm{t}}(\mathbf{x}, \boldsymbol{\omega}) \int_{S^2} L_{\mathrm{i}}(\mathbf{x}, \boldsymbol{\omega}')\, f_s(\mathbf{x}, \boldsymbol{\omega}, \boldsymbol{\omega}')\, |\beta(\mathbf{x}) \cdot \boldsymbol{\omega}'|\, \mathrm{d}\boldsymbol{\omega}'.$$

We further use the sum of the above expression with the absorption-weighted emission to define an extended outgoing radiance function L_{o} for points $\mathbf{x}$ that lie within the many-worlds representation:

$$\mu_{\mathrm{s}}(\mathbf{x}, \boldsymbol{\omega}) L_{\mathrm{s}}(\mathbf{x}, \boldsymbol{\omega}) + \mu_{\mathrm{a}}(\mathbf{x}, \boldsymbol{\omega}) L_{\mathrm{o}}(\mathbf{x}, \boldsymbol{\omega}) =: \mu_{\mathrm{t}}(\mathbf{x}, \boldsymbol{\omega}) L_{\mathrm{o}}(\mathbf{x}, \boldsymbol{\omega}). \quad (41)$$

Substituting all of these expressions into Equation 34, we obtain

$$L_{\mathrm{i}}^{\varepsilon}(0) = \varepsilon \int_0^s \alpha(t)\, \left[L_{\mathrm{o}}(t) - L_{\mathrm{o}}(s)\right] \mathrm{d}t + L_o(s), \quad (42)$$

where α gives the occupancy at t and is zero in the back-facing ($\boldsymbol{\beta}(t) \cdot \boldsymbol{\omega} < 0$) case.

This equation quantifies how the many-worlds representation modifies light transport along individual path segments. For multi-segment paths, we do not model multiple interactions, which would introduce second-order derivative terms beyond our scope.

While Equation 42 captures infinitesimal volumetric effects, our many-worlds framework treats each perturbation as a standalone alternative world (Section 3.1). Removing the ε scaling reveals the *unit perturbation* effect:

$$L_{\mathrm{i}}(0) = \int_0^s \alpha(t)\, \left[L_{\mathrm{o}}(t) - L_{\mathrm{o}}(s)\right]\, \mathrm{d}t + L_o(s). \quad (43)$$

Under differentiation this gives the many-worlds derivative transport in Equation 6.

[3]This behavior is non-reciprocal because when the medium interacts along $\boldsymbol{\omega}$ (i.e., if $\mu_t(\mathbf{x}, \boldsymbol{\omega}) > 0$), then the opposite direction is non-interacting (i.e., $\mu_t(\mathbf{x}, -\boldsymbol{\omega}) = 0$).

C Technical Details

C.1 Stochastic surface model

Our method evolves a surface $\bar{S}$ by propagating gradients to a stochastic surface model S. We outline one possible parameterization of a stochastic surface model here [Williams and Fitzgibbon 2006; Sellán and Jacobson 2022; Sellán and Jacobson 2023; Miller et al. 2024; Seyb et al. 2024], noting that this is not our contribution and other formulations can be used. Our work focuses on the theoretical foundation for optimizing *a distribution of surfaces* without relying on exponential volumes (i.e., ray-surface interactions are mutually exclusive events, rather than statistically independent events), which is largely independent of the specific model used.

An implicit representation $\Phi(\mathbf{x})$ of a shape determines whether a position $\mathbf{x}$ is inside ($\Phi(\mathbf{x}) < 0$), outside ($\Phi(\mathbf{x}) > 0$), or on the boundary ($\Phi(\mathbf{x}) = 0$) of a solid. This geometric classification is independent of optical properties—for example, points $\mathbf{x}$ within a refractive material also count as *inside* ($\Phi(\mathbf{x}) < 0$).

In our case, the scene models a distribution of surfaces, which turns $\Phi(\mathbf{x})$ into a random variable. The occupancy $\alpha(\mathbf{x})$ then gives the probability of $\mathbf{x}$ being on or inside an object:

$$\alpha(\mathbf{x}) = \Pr\{\Phi(\mathbf{x}) \le 0\}. \tag{44}$$

We model $\Phi(x)$ as a *Gaussian process* (GP). Pointwise evaluations of the implicit function are normally distributed:

$$\Phi(\mathbf{x}) \sim \mathcal{N}(\mu(\mathbf{x}), \sigma^2(\mathbf{x})), \tag{45}$$

which yields an explicit form of the occupancy:

$$\alpha(\mathbf{x}) = \frac{1}{2}\left[1 - \operatorname{erf}\left(\frac{\mu(\mathbf{x})}{\sqrt{2\sigma^2(\mathbf{x})}}\right)\right]. \tag{46}$$

We assume a constant variance $\sigma^2(\mathbf{x}) = \sigma^2$ for all $\mathbf{x}$ in this work.

We also use this representation to assign an orientation to every point based on the expected gradient of the implicit function, i.e.,

$$\boldsymbol{\beta}(\mathbf{x}) = \frac{E\left[\nabla\Phi(\mathbf{x})\right]}{\|E\left[\nabla\Phi(\mathbf{x})\right]\|} = \frac{\nabla\mu(\mathbf{x})}{\|\nabla\mu(\mathbf{x})\|}.$$

Modeling orientation has a crucial impact on the robustness and speed of optimizations (Figure 4).

A GP also has the property that evaluations $\Phi(\mathbf{x}_1), \Phi(\mathbf{x}_2), \ldots$ follow a joint multivariate normal distribution. Their auto-correlation is often described using a *kernel* that depends on distance, e.g.:

$$\operatorname{corr}(\Phi(\mathbf{x}), \Phi(\mathbf{y})) = \exp\left(-\gamma\|\mathbf{x} - \mathbf{y}\|^2\right) \tag{47}$$

where corr() refers to Pearson's correlation coefficient. This ensures that nearby points become increasingly correlated, making sudden jumps in realizations of $\Phi(\mathbf{x})$ unlikely.

Autocorrelation would be a crucial property if our method involved steps such as sampling concrete surface realizations from S, or if we consider interaction with potential surfaces at multiple different locations, requiring careful modeling of the correlation between them. However, our method does not depend on such steps. Its sole interaction with S is through pointwise evaluations of probabilities, and the extraction of the *background surface* $\bar{S}$:

$$\bar{S} = \{\mathbf{x} \in \mathbb{R}^3 \mid \mu(\mathbf{x}) = 0\} = \{\mathbf{x} \in \mathbb{R}^3 \mid \alpha(\mathbf{x}) = 1/2\}. \tag{48}$$

Spatial correlation is therefore not a detail that must be modeled in our implementation of the algorithm. Other variants of the many-worlds algorithm, for instance those that involve a more general directional distribution, require more parameterization of the GP.

C.2 Implementation details

Occupancy and background surface. We query $\mu(\mathbf{x})$ from a grid-based texture using cubic interpolants in our implementation. Alternatively, a neural representation [Müller et al. 2022] could be employed for greater flexibility and resolution.

To simulate interactions with the detached background surface $\bar{S}$, we use Marching Cubes [Lorensen and Cline 1987] to extract a triangle mesh from $\mu(\mathbf{x})$, enabling the use of efficient ray-triangle intersection routines. Any other isosurface extraction algorithm could in principle be used, since the extraction step and the extracted surface do not need to be differentiable. Our pipeline does not involve sphere tracing, delta tracking or other iterative algorithms.

Alternative pamameterization. The derivation in Section 3 discusses the propagation of derivatives to some parameterization of the extended parameter space. Our choice of parameterization is simple but unnecessarily restrictive.

For instance, it should be possible to construct versions of this framework to model $\boldsymbol{\beta}$ as a normal distribution rather than a single normal direction. This could accelerate convergence by allowing the optimizer to simultaneously explore a wider range of surface orientations. Upon surface extraction, the normal distribution could be interpreted as micro-scale surface roughness, which we leave for future work.

Sampling strategy. A detail that must be addressed is how Equation 5 should be sampled in a Monte Carlo renderer. Since we don't know the value of the radiance, a natural choice is to sample other terms proportional to their known contribution—in this case, drawing positions proportional to $\alpha(t)$. While this is a sensible choice for primal rendering, it leads to a chicken-and-egg problem during optimization. Regions with $\alpha(\mathbf{x}) \approx 0$ are essentially never sampled, but we must clearly visit them *sometimes* to even consider the possibility of placing a surface there.

Recall the many-worlds derivative transport in Equation 6:

$$\partial_\pi L_\mathrm{i}(0) = \int_0^s \Big(\underbrace{\partial_\pi\alpha(t)\,[L_\mathrm{o}^\mathrm{S}(t) - L_\mathrm{o}^{\bar{\mathrm{S}}}(s)]}_{\text{(i)}} + \underbrace{\alpha(t)\,\partial_\pi L_\mathrm{o}^\mathrm{S}(t)}_{\text{(ii)}}\Big)\,\mathrm{d}t.$$

Term (i) of this expression states that the derivative of occupancy along the ray is proportional to $L_\mathrm{o}^\mathrm{S}(t) - L_\mathrm{o}^{\bar{\mathrm{S}}}(s)$, which is an unknown quantity that must be estimated. Given that there are *no remaining known factors* that could be used to sample t, the best strategy for this derivative is to *uniformly* pick points along the ray.

This isn't a new discovery: Nimier-David et al. [2022] discuss the same issue in inverse volume rendering and address it by introducing generalizations of volume trackers that sample proportional to pure transmittance $T(t)$ instead of the default extinction-weighted transmittance $\mu_\mathrm{t}(t)T(t)$. Many-worlds transport has no transmittance, so the solution ends up being even simpler: a uniform sampling strategy suffices.