Radiance Caching for Differentiable Path Tracing

ZIYI ZHANG, École Polytechnique Fédérale de Lausanne (EPFL) and Google, Switzerland
DELIO VICINI, Google, Switzerland
SEBASTIAN WINBERG, Google, Switzerland
STEPHAN GARBIN, Google, United Kingdom
WENZEL JAKOB, École Polytechnique Fédérale de Lausanne (EPFL), Switzerland

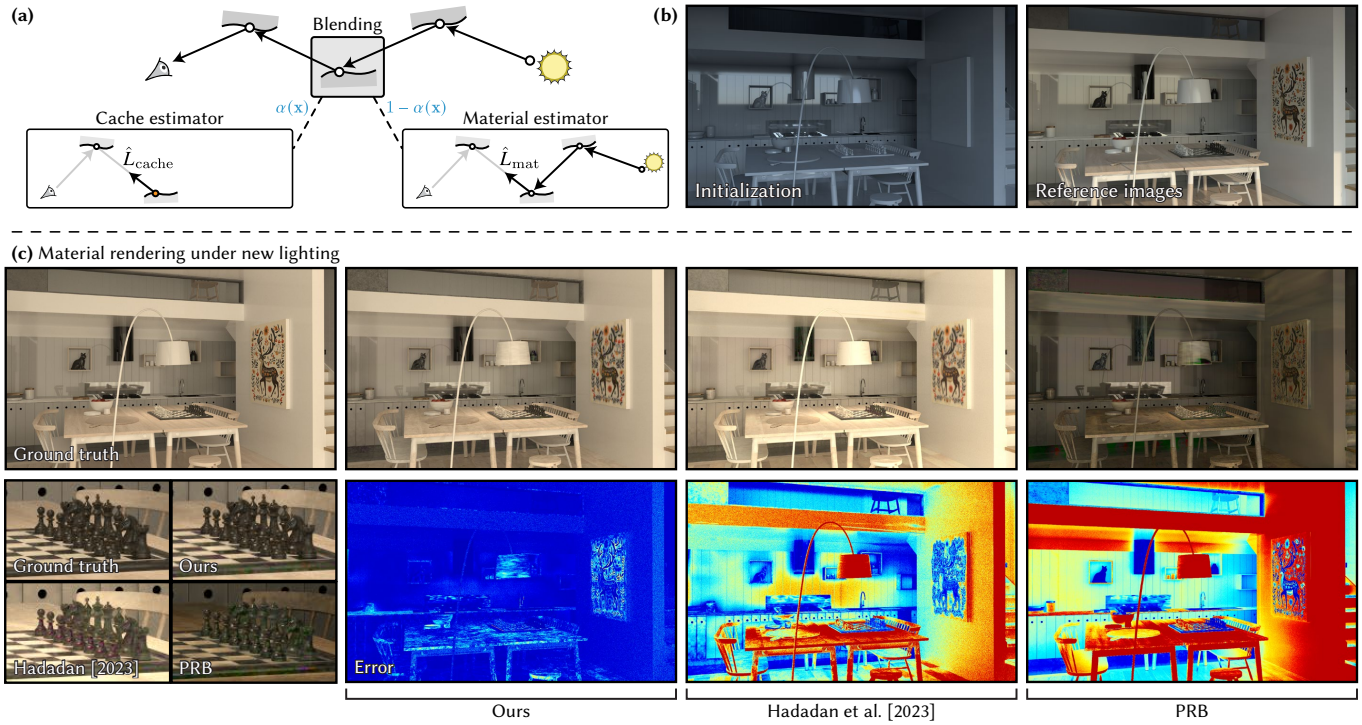


Fig. 1. **(a)** At each surface interaction, we estimate outgoing radiance by blending a cache estimator (querying a learned radiance cache) and a material estimator (continuing path tracing through BSDFs). A spatial blending field $\alpha(\mathbf{x})$ learns where to trust each estimator. Our training discourages degenerate decompositions where either estimator is only meaningful inside the blend, enabling reuse of the recovered materials for editing and relighting. **(b)** Starting from a rough material initialization (visualized under unknown lighting), we jointly optimize cache, materials, and α. **(c)** We discard cache and α and render *material-only* under an unseen relighting condition, comparing against Hadadan et al. [2023] and PRB [Vicini et al. 2021].

Differentiable path tracing offers a principled route to recovering physical material and lighting parameters, but the combination of high variance and poor numerical conditioning often makes it too brittle to use in practice. This is especially the case when lighting is altogether unknown, or when the scene contains complex light transport effects. Prior work recently showed that the variance reduction provided by a radiance cache can alleviate these challenges.

We revisit the combination of inverse rendering and radiance caching with a twist, by introducing a spatial blending field that locally interpolates between the cache and standard unbiased estimators. Recursive application of this idea yields a rich design space of evaluation strategies and inter-estimator consistency losses; we map this space and identify effective components. A surprising property of the resulting algorithm is that it can accurately recover material parameters even when the lighting is not uniquely identifiable from the observations. Our experiments demonstrate significant improvements in speed and robustness over prior work, making a strong case for including radiance caching as a standard component of future physically based inverse rendering systems.

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

Additional Key Words and Phrases: differentiable rendering, inverse rendering, path tracing, radiance caching, material reconstruction

Authors' Contact Information: Ziyi Zhang, École Polytechnique Fédérale de Lausanne (EPFL) and Google, Lausanne, Switzerland; Delio Vicini, Google, Zurich, Switzerland; Sebastian Winberg, Google, Zurich, Switzerland; Stephan Garbin, Google, London, United Kingdom; Wenzel Jakob, École Polytechnique Fédérale de Lausanne (EPFL), Lausanne, Switzerland.

ACM Reference Format:
Ziyi Zhang, Delio Vicini, Sebastian Winberg, Stephan Garbin, and Wenzel Jakob. 2026. Radiance Caching for Differentiable Path Tracing. *ACM Trans. Graph.* 45, 4, Article 135 (July 2026), 17 pages. https://doi.org/10.1145/3811398

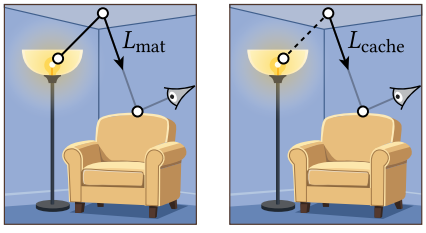
1 Introduction

Joint physically based recovery of geometry, materials, and lighting is a long-standing goal in graphics and vision, but the ambiguous nature of this problem keeps it out of reach of current methods. We tackle an important subproblem: given reference images and geometry (e.g., from a sensor, multi-view reconstruction, or a tentative estimate in an iterative pipeline), recover materials under unknown illumination. Even this reduced problem defeats current differentiable path tracing methods, which are fragile and often fail to recover acceptable solutions.

Three fundamental issues compound to cause this fragility: high-variance radiance estimates in the path tracer that lead to noisy gradients, poor numerical conditioning that amplifies sensitivity to noise and initialization, and an optimization landscape that is riddled with local minima. Increasing the sample count can reduce Monte Carlo noise, but the cost of reducing it to acceptable levels is prohibitive and would address only one of the three issues.

Along a light path, multiple unknown materials contribute multiplicatively to the final pixel value, making it difficult for gradient descent to disentangle their individual contributions. Unknown lighting worsens this problem because the optimizer can explain reference images simply by placing emission everywhere, which technically fits the data but is not the desired material-lighting decomposition.

A radiance cache can alleviate the noise issue by replacing high-variance Monte Carlo estimates with a learned cache that approximates the outgoing radiance at surface points. Radiance caches have proven highly effective in forward rendering; for example, Müller et al. [2021] demonstrated that the overhead of training and querying a neural radiance cache can be low enough for real-time path tracing. Despite pioneering work by Hadadan et al. [2023] for variance reduction, the use of radiance caching for physically based inverse rendering remains underexplored. We show that this is a potent combination that can improve the well-posedness of the problem, and present a general framework of which Hadadan et al.'s method is a special case.



To build intuition, consider light from a lamp reaching a chair indirectly after reflecting off the ceiling. A standard path tracer would estimate this radiance by tracing a ray to the ceiling and then either sampling or evaluating its BSDF to connect to the light source. We refer to this approach as the *material estimator* because it simulates the material's reflectance behavior. Alternatively, if we maintain a radiance cache storing outgoing radiance on surfaces, we could query it directly at the ceiling intersection and terminate the path. This *cache estimator* avoids further path tracing and BSDF evaluation; with an accurate cache, the two estimators agree. In practice, they exhibit complementary failure modes: the cache estimator is fast and low-variance but may be unreliable if the cache is unconverged, has insufficient capacity, or cannot represent high-frequency directional variation; the material estimator captures such effects more easily but produces noisy estimates that can impede or break the optimization.

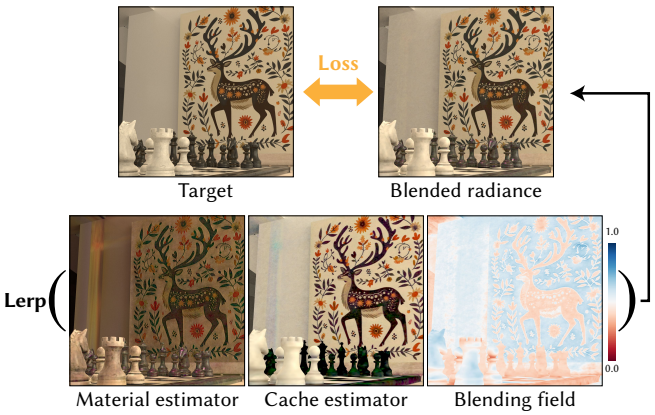


Fig. 2. **Naive blending is ill-posed.** Optimizing only the blended radiance can match the reference image, yet the cache-only estimate (queried at camera-ray intersections) and the material-only estimate (path tracing without cache) remain incorrect. The resulting parameters are not meaningful in isolation and cannot be reliably reused for relighting or editing.

A natural idea is to blend these two estimators adaptively. We introduce a spatially varying *blending field* $\alpha(\mathbf{x}) \in [0,1]$ and define a *blended estimator*

$$\hat{L}_o(\mathbf{x},\boldsymbol{\omega}) = \big(1-\alpha(\mathbf{x})\big)\,\hat{L}_{\text{mat}}(\mathbf{x},\boldsymbol{\omega}) + \alpha(\mathbf{x})\,\hat{L}_{\text{cache}}(\mathbf{x},\boldsymbol{\omega}), \tag{1}$$

where $\hat{L}_{\text{mat}}$ and $\hat{L}_{\text{cache}}$ are radiance estimates from the material and cache estimators, respectively. Intuitively, $\alpha(\mathbf{x})$ encodes how much we trust the cache at $\mathbf{x}$: when $\alpha(\mathbf{x}) \approx 1$, we terminate paths early using the cache; when $\alpha(\mathbf{x}) \approx 0$, we fall back to the material estimator to capture complex directional effects.

The blended estimator is simple to implement with a few additional lines of code, and it quickly drives the renderer toward the reference. However, as illustrated in Figure 2, it suffers from a fundamental decomposition problem: optimizing the blend constrains the estimators in combination but not individually. Even when the blend perfectly matches the reference, the material and cache estimators on their own may be far from the true solution. In other words, the reconstruction is only valid for the particular spatially varying blend used during training, which limits its reuse for downstream tasks such as rasterization, relighting, or material editing.

In the remainder of this paper, we take a step toward a more comprehensive understanding of radiance caching for inverse rendering. We explore and compare several optimization strategies beyond naive blending to pursue two goals: to encourage the cache

and material estimators to remain useful in isolation, and to learn a blending field that automatically favors the more reliable estimator at each location. Compared to pure differentiable path tracing, our experiments suggest that this joint formulation is particularly beneficial in three respects:

(1) **Robustness to unknown lighting:** We obtain stable reconstructions without known emitters. Light paths need not reach an emitter; the cache provides reasonable incident radiance at intermediate points, preventing the optimizer from collapsing to degenerate solutions.
(2) **Spatially adaptive estimator selection:** The learned blending field $\alpha(\mathbf{x})$ allows the optimizer to rely on the cache in regions requiring long indirect paths, while falling back to the material estimator near glossy surfaces where the cache is biased.
(3) **Improved behavior in high-variance regimes:** In scenes with complex global illumination, the cache terminates paths earlier and reduces variance, making optimization tractable in cases where pure path tracing struggles.

2 Related Work

2.1 Radiance caching

Early work on *irradiance caching* [Ward et al. 1988; Greger et al. 1998] reduced the cost of global illumination by evaluating low-frequency indirect irradiance at sparse points and interpolating it during rendering. Krivanek et al. [2005] extended this idea to *radiance caching*, which stores direction-dependent outgoing radiance. A key challenge for caching methods is artifact-free interpolation from sparse samples without blur or light leaks [Křivánek et al. 2007; Jarosz et al. 2008; Marco et al. 2018; Binder et al. 2018]. *Neural radiance caching* [Müller et al. 2021] replaces explicit interpolation of precomputed samples with an online-learned, continuous radiance predictor trained directly from noisy Monte Carlo estimates. We similarly fit a cache online but focus on its use in inverse rendering. Recently, variations of radiance caching have seen widespread use for real-time path tracing [Müller et al. 2021; Silvennoinen and Lehtinen 2017; Dereviannykh et al. 2025; AMD 2025; NVIDIA 2024]. Recent forward-rendering work also studies surface-based cache representations and explicit path-termination tradeoffs: Tatzgern et al. [2024] use on-surface radiance caches for real-time global illumination, while Kandlbinder et al. [2024] optimize when paths should terminate into a cache by explicitly trading bias and variance. Our setting differs because the cache, materials, and blending field are optimized from image observations, so termination choices must also provide useful inverse-rendering supervision.

2.2 Differentiable path tracing

There is a rich literature on differentiable path tracing for inverse rendering [Gkioulekas et al. 2016; Li et al. 2018; Zhang et al. 2019; Azinović et al. 2019; Zhao et al. 2016; Khungurn et al. 2015; Velinov et al. 2018; Nimier-David et al. 2019]. Central challenges for differentiable path tracing are visibility discontinuities [Li et al. 2018; Loubet et al. 2019; Zhang et al. 2019; Bangaru et al. 2020; Zhang et al. 2023a] and the memory use of the backward pass [Nimier-David et al. 2020; Vicini et al. 2021]. Besides the usual Monte Carlo noise in incident radiance estimates, gradient estimation introduces additional derivative terms that are often badly matched to primal sampling strategies; consequently, the variance of gradients can become arbitrarily large [Zeltner et al. 2021; Nimier-David et al. 2022]. High variance can slow convergence or cause the optimization to fail, especially with nonlinear loss functions [Nicolet et al. 2023]. Variance reduction strategies include importance sampling of local gradient terms [Zeltner et al. 2021; Belhe et al. 2024], path guiding [Fan et al. 2024], reservoir sampling [Wang et al. 2023] and control variates [Nicolet et al. 2023; Lu et al. 2025]. The impact of gradient noise can also be mitigated using preconditioning [Nicolet et al. 2021; Weier et al. 2025] or filtering [Chang et al. 2024]. We focus on reducing variance due to complex global illumination and unknown emitters using radiance caching. This is largely orthogonal and could be combined with many of the above techniques. In particular, Parameter-space ReSTIR improves sampling efficiency by reusing samples in parameter space, whereas our method changes the estimator decomposition through a cache/material split and separated image-space losses; ReSTIR-style sample reuse could in principle be used inside our material estimator or separated-loss samples [Chang et al. 2023]. Adjacent to our work are *variance-aware* optimization techniques [Weier et al. 2021; Yan et al. 2024]. Accounting for rendering variance explicitly is an interesting future direction.

2.3 Radiance caching in inverse rendering

Several works have explored the application of radiance caching for inverse rendering. The most similar to our work is the *neural radiometric prior* [Hadadan et al. 2023], which we discuss in detail below. We also review the use of radiance caching with radiance field representations (e.g., NeRFs) and alternative approaches.

Neural radiometric prior. Hadadan et al. [2023] augmented a differentiable path tracer with a neural radiance cache, demonstrating improvements over unbiased differentiable path tracing [Vicini et al. 2021]. Their method evaluates the cache at a fixed path depth (typically at the first surface hit) and only accumulates gradients into the material at that interaction, even if the path continues further. Our method generalizes their approach by removing both limitations and introducing a more general family of consistency losses. Section 4 provides detailed comparisons.

Radiance field methods. Following the seminal work on neural radiance fields (NeRFs) [Mildenhall et al. 2020], there has been continued interest in developing *relightable* variants of radiance-field-like representations. Due to the high evaluation cost of neural radiance fields, most methods in that space try to limit the number of evaluated ray queries or interactions. Initially, Zhang et al. [2021] augmented NeRF with a direct illumination term. Since NeRFs represent a scene's radiance field, a natural extension is to try to leverage that field directly as a radiance cache to render indirect illumination [Zhang et al. 2022; Jin et al. 2023; Sun et al. 2025]. Gu et al. [2025] similarly compute a single indirect bounce on a 2D Gaussian representation [Kerbl et al. 2023; Huang et al. 2024]. Variations of this idea are *neural incident radiance fields* [Yao et al. 2022; Wu et al. 2023; Zhang et al. 2023b; Wu et al. 2025]. These methods represent

both outgoing and incident radiance fields, which allows them to avoid recursive ray tracing to evaluate the radiance cache. Similarly, neural fields have also been used for more accurate emitter modeling in physically based inverse rendering [Ling et al. 2024]. Generally, NeRF-based methods avoid multi-bounce ray tracing due to the high computational cost. For that reason, the quality of results can deteriorate if the cache is not expressive enough. Attal et al. [2024] recognized this problem and proposed a few strategies to reduce bias, but their method also does not trace multiple light bounces. We instead use multi-bounce path tracing to disentangle materials from the cache and further reduce bias from the inherent limitations of radiance caching. Our method thus exposes the classic bias-variance tradeoff.

Alternative approaches. Radiance caching is not the only way to avoid recursive ray tracing in inverse rendering. Jiang et al. [2025] perform an iterative radiosity solve directly on 3D Gaussian primitives. Poirier-Ginter et al. [2025] only path trace specular transport and cache the diffuse component. This improves novel view synthesis but does not directly enable relighting. Finally, Hadadan and Zwicker [2024] cache differential quantities at the cost of limiting reconstruction to low-dimensional parameter spaces.

3 Method

As previously discussed, naive optimization of the blended estimator is under-constrained, as the model can perfectly match the reference by arbitrarily blending two individually incorrect estimators. In this section, we explore several loss formulations to resolve this ambiguity and furthermore learn the blending field $\alpha(\mathbf{x})$.

Our starting point is the rendering equation, which relates outgoing and incident radiance at a surface point:

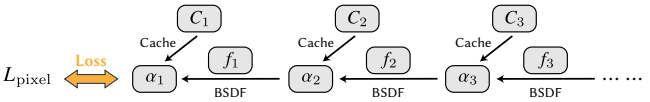
$$L_o(\mathbf{x},\omega) = L_e(\mathbf{x},\omega) + \int_{S^2} L_i(\mathbf{x},\omega')\, f(\mathbf{x},\omega',\omega)\, \mathrm{d}\omega'^{\perp}, \tag{2}$$

where $L_e(\mathbf{x},\omega)$ and $L_i(\mathbf{x},\omega')$ denote emitted and incident radiance, $f(\mathbf{x},\omega',\omega)$ is the BSDF, and $\mathrm{d}\omega'^{\perp}$ includes the projected-solid-angle factor. A path tracer estimates the integral by sampling a direction ω' and recursively estimating incident radiance. For now, we assume the emission is either known or zero. At each surface interaction, our blended estimator interpolates two approaches:

$$\hat{L}_o(\mathbf{x},\omega) = (1-\alpha(\mathbf{x}))\underbrace{\frac{f(\mathbf{x},\omega',\omega)\cos\theta'}{p(\omega')}\hat{L}_i(\mathbf{x},\omega')}_{\text{Material}} + \alpha(\mathbf{x})\underbrace{C(\mathbf{x},\omega)}_{\text{Cache}}, \tag{3}$$

where $\alpha(\mathbf{x}) \in [0,1]$ models our trust in the cache at position $\mathbf{x}$, C is a learned cache storing outgoing radiance, θ' is the angle between ω' and the surface normal, and $p(\omega')$ is the sampling probability. In the known-emission discussion below, C may be interpreted as scattered outgoing radiance; in our unknown-emission formulation in Section 3.2, it stores total outgoing radiance including emission.

The illustration below shows the computation graph of a direct optimization of this recursive estimator. All terms receive gradients (gray boxes), but only their weighted combination is constrained, which leads to an ill-posed problem.



During training, we render with this blended estimator, but the novelty of our method lies in how we train the full set of parameters (cache, materials, and α) by imposing additional losses that enforce the individual correctness of the estimators.

3.1 Consistency losses

We first present two intuitive consistency-loss formulations to address this issue but then show that both are inherently flawed.

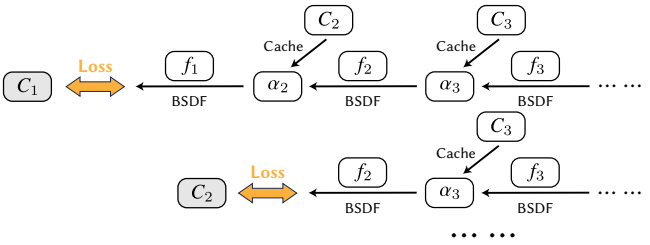
Depending on the optimized parameter, we can enforce consistency in either direction: by training the cache to match the material estimator (Section 3.1.1), or vice versa (Section 3.1.2). Both losses follow directly from the rendering equation, but require care in choosing which terms to differentiate and how to sample them to obtain unbiased gradients.

We assume here that the consistency losses are applied in the context of minimizing an image loss based on the blended estimator, which also optimizes the blending field. If both estimators are useful approximations of the same outgoing radiance, blending becomes primarily a bias-variance tradeoff: α selects whichever estimator is more reliable at each location.

3.1.1 Cache consistency loss. The first loss formulation trains the cache C to match the reflected-radiance integral in the rendering equation:

$$\mathcal{L}_{\text{cache}}(\mathbf{x},\omega) = \left\| \boxed{C(\mathbf{x},\omega)} - \int_{S^2} f(\mathbf{x},\omega',\omega)\cos\theta' L_i(\mathbf{x},\omega')\mathrm{d}\omega' \right\|^2_{\text{rel}}. \tag{4}$$

The loss is parameterized by $\mathbf{x}$ and ω. In practice, it would be used by accumulating gradients with respect to $\mathcal{L}_{\text{cache}}$ whenever the path tracer encounters a vertex $\mathbf{x}$ from direction ω and then samples a scattered direction ω', which also provides the direction for a single-sample estimate of the integral in Equation 4. This focuses the consistency optimization on the spatio-directional subset of ray space that actually contributes to the rendered image. Imposing this loss at different path depths leads to the following graph structure (with gray terms receiving gradients while others are held fixed).



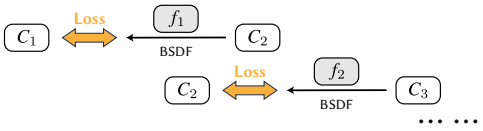
This is the loss used by Müller et al. [2021] to train a neural radiance cache in forward rendering, where materials and emitters are known and the target is therefore a consistent Monte Carlo estimate. The target is a one-sample Monte Carlo estimate, which is noisy; following their approach, we use a relative L^2 loss so that the cache learns the expected value rather than overfitting to individual samples [Lehtinen et al. 2018]. In practice, we substitute the blended

estimate for $\hat{L}_i$, which reduces variance and reuses values already computed along the path.

3.1.2 Material consistency loss. Conversely, we can train the BSDF to match the cache. If we use C to evaluate incident and outgoing radiance, the rendering equation provides a training signal for f:

$$\mathcal{L}_{\text{mat}}(\mathbf{x}, \omega) = \left\| C(\mathbf{x}, \omega) - \int_{S^2} f(\mathbf{x}, \omega', \omega) \cos\theta' C(\mathbf{x}', -\omega')\, \mathrm{d}\omega' \right\|^2_{\text{rel}}, \tag{5}$$

where $\mathbf{x}'$ is the surface point visible from $\mathbf{x}$ in direction ω'. Here, gradients only propagate through the BSDF (gray box), while the other terms are held fixed. As before, this loss can be imposed at different path depths:



Unfortunately, the simple approach for constructing a one-sample derivative estimator that worked in Section 3.1.1 now fails and produces biased gradients. The issue arises from the combination of the nonlinear L_2 loss and the fact that we are now propagating derivatives *into* the integral. Appendix A provides more detail and presents a decorrelated two-sample estimator that resolves the issue.

3.1.3 Limitations. The consistency losses suffer from multiple fundamental limitations:

- **Directional mismatch.** The cache consistency loss propagates information from the emitters toward the camera and is therefore well-suited to forward rendering, where known emitter and material parameters are used to infer the observed image. In inverse rendering, the supervision comes from reference imagery at the camera. The loss therefore propagates information counter to the direction of supervision, making it inherently less suitable. Along a camera-to-light path $(\mathbf{x}_1, \ldots, \mathbf{x}_n)$, cache consistency trains $C(\mathbf{x}_i)$ from the suffix $(\mathbf{x}_{i+1}, \ldots, \mathbf{x}_n)$. This is appropriate when emitters/materials are known, but with unknown emission the reliable signal is the observed pixel at $\mathbf{x}_1$, so supervision flows away from the data source.
- **Garbage in, garbage out.** Early in optimization, both the materials and the cache are unreliable (e.g., randomly initialized). Training with a consistency loss then simply transfers bad data from one representation to the other.
  This is particularly problematic for material consistency: the cache has limited capacity and cannot represent the radiance function with high accuracy. High-frequency spatio-directional variation may be severely distorted or lost entirely, preventing the cache from ever becoming an authoritative training signal for materials.
- **Unknown emission.** Both losses assume non-emissive surfaces, yet emission must clearly be present for any radiance to exist. Generalizing the formulations to include an emission term does not help: reflected and emitted radiance cannot be separated without additional constraints, so the consistency relationship no longer defines a meaningful target for either estimator.

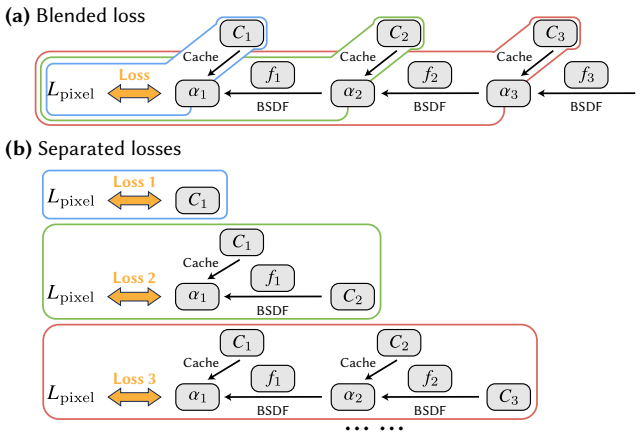


Fig. 3. **Separated losses.** (a) A single blended loss supervises only the recursively mixed estimator, which can leave the cache and material factors under-constrained in isolation. (b) We instead define pixel losses that force the path to terminate at a chosen bounce k, using the cache value C_k as the endpoint. Each such loss backpropagates to C_k and to the BSDF factors before it, removing the "only-the-blend-is-supervised" ambiguity.

These limitations motivate an alternative approach. We modify the role of the cache so that it represents total outgoing radiance (i.e., including both emission and scattering) instead of trying to separate their individual contributions. On top of this representation, we introduce a unified family of losses that simultaneously constrain both estimators with respect to the reference images. This drives gradients from the camera into the scene, aligning the natural flow of information with the inverse rendering problem.

3.2 Separated estimators and losses

The cache, material, and blended estimators all aim to recover outgoing radiance L_o. At any surface interaction along a path, we can therefore terminate via a cache lookup or continue recursively, yielding distinct image-space losses. Satisfying these losses discourages decompositions where either estimator is meaningless outside the blend.

Let $(\mathbf{x}_j, \omega_j)$ be the j-th surface interaction, with $j = 1$ at the primary hit. As illustrated in Figure 3, instead of matching only the blended rendering, we sample a forced termination depth; across iterations, different depths supervise different combinations of cache values and material factors. Terminating at depth k while using the blended estimator at earlier vertices gives:

$$\hat{L}_j^{(k)} = \begin{cases} (1-\alpha_j)\, \dfrac{f_j \cos\theta_j}{p_j}\, \hat{L}_{j+1}^{(k)} + \alpha_j\, C_j, & j < k, \\ C_k, & j = k. \end{cases} \tag{6}$$

Here $C_j = C(\mathbf{x}_j, \omega_j)$, $\alpha_j = \alpha(\mathbf{x}_j)$, and f_j is the sampled BSDF factor. For $k = 1$, the image loss directly trains the primary-hit cache; for $k = 2$, it trains the first-bounce BSDF using cached radiance at the second hit; larger k values supervise longer material chains before cache termination.

Listing 1. Pseudocode of the separated radiance estimator (Equation 6) using a blending field-aware stopping criterion (Appendix C) with threshold τ. The structure of this algorithm resembles standard (differentiable) path tracing and only requires minimal changes in existing systems.

```
def render_pixel(pixel_uv, τ):
    x, ω = generate_camera_ray(pixel_uv)
    return Li(x, ω, τ)

def Li(x, ω, τ):
    L = 0; β = 1; α_prod = 1
    for _ in range(MAX_DEPTH):
        x = ray_intersect(x, ω)
        α, C = eval_cache_alpha(x)
        # Path termination check (separated estimator)
        if α_prod * (1 - α) < τ:
            return L + β * C
        # Accumulate cache contribution and continue path
        L += β * α * C
        ω, β_bsdf = sample_bsdf(x, ω)
        β *= (1 - α) * β_bsdf; α_prod *= (1 - α)
    return L
```

Implementation. A naive optimization approach would simultaneously compute and optimize renderings for $k \in \{1, \dots k_{\max}\}$ up to some maximum path depth $k_{\max}$, but this is too memory-intensive. Instead, we can sample a different termination depth k each iteration, keeping memory constant. The distribution over k determines the relative weight of each *separated loss*.

Termination strategies. One relevant concern is that simple uniform sampling may terminate at vertices with a low value of α, propagating errors from an unreliable cache to upstream BSDFs.

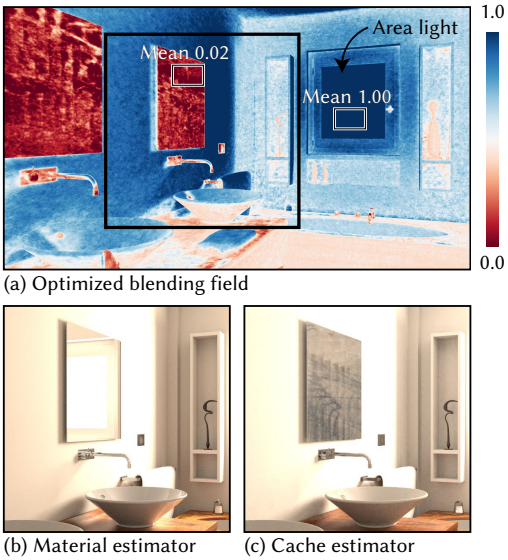
In practice, optimizers like Adam are fairly robust to occasional bad gradients. Nonetheless, an α-aware strategy that extends the path beyond the sampled depth until $\prod(1-\alpha)$ falls below a threshold avoids low-α endpoints and improves stability by a small margin (~0.4 dB PSNR across scenes). We detail options in Appendix C.

Listing 1 shows a pseudocode implementation of the forward pass. The reverse-mode derivative of this algorithm can be evaluated using path replay backpropagation [Vicini et al. 2021].

3.3 Optimizing the blending field

The blending field $\alpha(\mathbf{x})$ locally controls how much we rely on the cache versus the material estimator. Consider setting $\alpha = 0$ everywhere: paths would then use only the material estimator, terminating using the cache when reaching a maximum depth. This works but sacrifices two benefits: blending at earlier vertices reduces variance, and the existence of a family of losses that probe both estimators in different ways yields a better-posed inverse problem. Section 4 analyzes these benefits in more detail.

We instead optimize α jointly with the cache and scene parameters, treating it as a learned *confidence field*. Since the true outgoing radiance is unknown during optimization, either estimator may be more accurate at any given point and training stage. Early on, materials may be far from correct while the cache already captures much of the signal. Learning α lets the system favor the more reliable estimator at each location and adapt as both evolve.



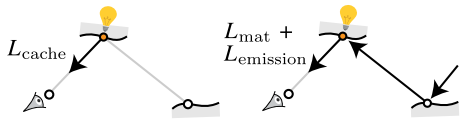
(a) Optimized blending field

(b) Material estimator (c) Cache estimator

Fig. 4. **Adaptive estimator selection via a learned blending field.** (a) Optimized blending field $\alpha(\mathbf{x})$. Blue colors indicate higher reliance on the cache estimator. (b) Rendering using the optimized materials with no cache. (c) Rendering using only the cache at camera-ray intersection points. The blending field $\alpha(\mathbf{x})$ increases reliance on the cache where indirect transport is high-variance and reduces reliance where the cache is wrong (e.g., specular effects).

Bias-driven optimization. We optimize α so that it favors the estimator with lower bias, which requires decorrelating the loss and gradient evaluations following standard practice in differentiable rendering [Gkioulekas et al. 2016; Azinović et al. 2019]. This provides a graceful fallback when either estimator struggles, which improves robustness. When the material estimator is unreliable in a region (e.g., due to high variance, poor initialization, or difficult transport), α naturally increases, and the system becomes more reliant on the cache. Conversely, where the cache is unreliable (e.g., on the mirror in Figure 4), α decreases and the system falls back to path tracing. This adaptive behavior prevents the failure of either estimator from destabilizing the optimization. A remaining failure mode is early alpha collapse: if $\alpha(\mathbf{x})$ reaches 1 too early in a region, the material branch there receives little or no gradient. We found this uncommon but possible. Regularizing α away from 1 can mitigate the issue, but it introduces a sensitive tuning parameter and slightly reduced quality in our experiments, so we do not use such a regularizer in the reported results.

Handling unknown emission. A unique challenge in inverse rendering is that emitter locations are often unknown. In our framework, the cache stores full outgoing radiance including emission, while the material estimator computes only reflected radiance. On emissive surfaces, setting $\alpha < 1$ would blend these inconsistent quantities.



Fortunately, the optimization naturally handles this: as shown in Figure 4a and other results, α converges to 1 on emissive surfaces, since this is the only configuration able to correctly model emission. Note that $\alpha = 1$ is not a definitive indicator of emission—it can also occur where material optimization struggles.

Variance-aware optimization. An alternative is to optimize α for both bias and variance by *not* decorrelating paths, which naturally favors the estimator with lower variance. This follows the principle of variance-aware inverse rendering [Yan et al. 2024], which shows that lower variance can lead to better optimization. While using correlated paths for material parameters yields biased gradients [Azinović et al. 2019; Nimier-David 2022], this is acceptable for α as both estimators target the same quantity. We include a comparison in Section 4 and leave further exploration for future work.

Alternative formulations. For completeness, we note that α can also be optimized via blended local losses computed at each bounce, without additional overhead. This idea has been used in Zhang et al. [2025] for optimizing blended losses over a surface distribution. We discuss this alternative in Appendix B.

3.4 Connection to Hadadan et al. [2023]

Hadadan et al. [2023] can be seen as a special case of our framework tailored to single-bounce material optimization. Key differences are *where* cache supervision is applied, *how* gradients are routed, and *whether* the estimator is allowed to adapt spatially and in path depth.

Mapping to our formulation. Hadadan et al. combine three ingredients: (i) a *primary-hit material loss* where materials at the first surface interaction are optimized using cached incident radiance from the next bounce; (ii) a *primary-hit cache loss* supervised directly by reference images; and (iii) a *cache-consistency* objective as in neural radiance caching [Müller et al. 2021] that propagates radiance information from emitters toward the camera.

In our terminology, (i) corresponds to a separated loss with $k = 2$ that updates materials using cached continuation beyond the first hit, without blending and with gradients detached from the cache; (ii) corresponds to a separated loss with $k = 1$ that directly supervises cache values at camera-ray intersections, and (iii) corresponds to a cache-consistency term that is effective for forward rendering but can become unreliable for inverse rendering, especially when emission is unknown.

Generalization in our method. Our approach removes the single-bounce restriction by optimizing a *family* of separated objectives, and introducing a learned blending field $\alpha(\mathbf{x})$ that adaptively selects between cache and material estimators across the scene. This makes the optimization more robust: when estimating incident radiance, cache values are only used where reliable, providing more accurate derivatives for materials along the entire light path.

Hadadan baseline.* In our experiments we also report a variant, Hadadan*, which disables the cache-consistency loss. This isolates the limitations of consistency terms in the unknown-lighting setting; we discuss the resulting behavior in Section 4.1.

4 Results

We evaluate our approach in three stages. First, we present results on a hard setting: room-scale material recovery under unknown lighting. Second, we analyze effects of key design choices. Third, we provide experiments under known lighting to show benefits of our method even in applications where lighting is available.

Evaluation methods. The method labeled "Ours" uses separated losses to train the cache, materials, and the blending field $\alpha(\mathbf{x})$ optimized for low bias (i.e., with decorrelated paths). At evaluation time, we discard both the cache and the blending field and path-trace images using only the optimized material information. This evaluates whether the recovered materials are physically plausible and reusable, rather than whether a training-time blend can match the input images. For all tables, we evaluate metrics by rendering unseen test views under a scene-specific adjusted relighting condition. All experiments use ground-truth scene geometry during optimization and evaluation. The reported improvements therefore isolate material-lighting disentanglement and variance reduction, not robustness to geometric reconstruction errors.

We do not rely on texture-space error as a primary metric because we optimize *all* materials jointly, including many regions that are occluded (e.g., floor under furniture) or never observed, making texture space evaluation unreliable.

4.1 Unknown lighting material optimization

When both lighting and materials are unknown, differentiable path tracing struggles to disentangle emission from reflectance. Gradient variance compounds the difficulty: every surface could be an emitter and must be sampled as such, leading to low-quality gradients that exacerbate an already degenerate problem.

Quantitative comparison. Table 1 reports reconstruction quality with test view material-only renderings. Our method substantially outperforms PRB and improves over Hadadan [2023]. PRB fails in this setting because it tries to explain most of the appearance via optimized emission, leading to poor material estimates that are not useful on their own.

We also report results for Hadadan*, which disables the cache consistency loss. This variant can improve numerical scores in some scenes because cache consistency propagates radiance from emitters toward the camera, leading to incorrect decompositions when the emission is unknown.

Qualitative comparison. Figure 5 visualizes representative reconstructions. For our method and Hadadan, we show the directly visible cache and the *material-only* rendering illuminated by the ground-truth lighting. PRB is shown both with its optimized emission and with ground-truth lighting. Consistent with the quantitative results, our recovered materials remain plausible, while PRB frequently produces materials that only explain the training images when paired with its optimized emission.

Metric	Method	Bedroom	Bathroom2	Kitchen	Living-rm	Living-rm-2	Living-rm-3	Staircase
PSNR ↑	Ours	**24.85**	**23.43**	**26.96**	**22.64**	**27.56**	29.57	**20.71**
	PRB	11.46	4.66	10.09	15.46	10.54	6.15	10.65
	Hadadan [2023]	16.47	14.82	23.92	18.50	20.33	30.42	12.15
	Hadadan [2023]*	18.26	15.24	24.21	18.89	19.91	**31.28**	17.33
SSIM ↑	Ours	**0.5225**	**0.8032**	**0.6724**	**0.8068**	**0.8331**	0.8769	**0.8350**
	PRB	0.4445	0.3357	0.4028	0.7101	0.5724	0.4464	0.3808
	Hadadan [2023]	0.4702	0.7182	0.6573	0.7474	0.8073	**0.8821**	0.6438
	Hadadan [2023]*	0.4904	0.7190	0.6588	0.7519	0.8049	0.8805	0.7401
LPIPS ↓	Ours	**0.2190**	**0.1254**	0.2154	**0.1257**	**0.1187**	0.0721	**0.1184**
	PRB	0.3113	0.4112	**0.2053**	0.1927	0.2030	0.2638	0.3098
	Hadadan [2023]	0.2698	0.1712	0.2284	0.1530	0.1411	**0.0689**	0.1910
	Hadadan [2023]*	0.2543	0.1581	0.2279	0.1521	0.1427	0.0738	0.1771

Table 1. **Unknown-lighting reconstruction quality.** Known geometry with unknown lighting and unknown materials. We evaluate by rendering test views using *material-only* path tracing (discarding cache and α) under a new lighting condition. Hadadan* denotes Hadadan et al. [2023] with cache consistency disabled. Best results per scene/metric are highlighted.

Blending reduces rendering variance. Figure 6 visualizes two renderings of the same optimized scene, one made with the pure material estimator (no cache, no α), and one made using the blended estimator, which produces a markedly cleaner image. These primal variance improvements directly translate into lower-variance gradients that stabilize the optimization and ultimately produce higher-quality reflectance parameters.

Simple scenes. One notable exception is the scene `Living-room-3` in Table 1, where Hadadan* can achieve higher PSNR. This scene is dominated by direct illumination and is sufficiently explained by single-bounce optimization; longer paths provide little benefit and can introduce additional Monte Carlo noise. We revisit this behavior in Section 4.2.2 as a limitation of multi-bounce training in simple transport regimes.

4.2 Design choices

We next analyze how key components contribute to performance in the unknown-lighting setting.

4.2.1 Importance of the blending field $\alpha(\mathbf{x})$. Table 2 compares our method against two alternatives: a constant blending field (in particular, $\alpha(\mathbf{x}) = 0.5$), and one that disables blending altogether ($\alpha(\mathbf{x}) = 0$). Learning a heterogeneous $\alpha(\mathbf{x})$ yields the best performance across scenes. A fixed blend helps but is consistently worse than an adaptive field, while disabling blending degrades performance severely.

Figure 4 illustrates why: the learned $\alpha(\mathbf{x})$ encodes a spatial map of estimator reliability, favoring the cache where material estimation struggles and falling back to Monte Carlo sampling where the cache bias is large. This adaptive routing is crucial under unknown lighting, where the optimizer must balance variance reduction against the risk of biased cache predictions.

Table 3 further tests whether a hand-crafted material heuristic can replace learning $\alpha(\mathbf{x})$. We set $\alpha(\mathbf{x}) = 0.45\, r_{\text{eff}}(\mathbf{x})$, where r_{eff} is a BSDF-derived effective roughness; the conservative cap prevents diffuse materials from fully suppressing material-path gradients. This baseline is viable but consistently worse than learning $\alpha(\mathbf{x})$. The result indicates that estimator reliability is not determined by local roughness alone: it also depends on cache accuracy, visibility, indirect transport, lighting ambiguity, and the current optimization state.

4.2.2 Maximum path depth. Table 4 studies the influence of the maximum path depth parameter $k_{\max}$ during training. Depth 2 is often insufficient because it forces cache usage at the second bounce regardless of $\alpha(\mathbf{x})$, which limits the optimizer's ability to select estimators based on reliability. Increasing to $k_{\max} = 3$ enables $\alpha(\mathbf{x})$ to meaningfully influence termination and generally improves reconstruction quality. Beyond depth 3, gains often plateau, suggesting most benchmark scenes are sufficiently captured by moderate depths.

Consistent with Section 4.1, `Living-room-3` favors shallow depth because transport is dominated by direct illumination.

4.2.3 Cache capacity. Figure 7 evaluates robustness to cache capacity over a wide range of hash map sizes. Performance remains stable across capacities, indicating that our approach is not overly sensitive to the cache representation. Intuitively, when cache approximation quality degrades due to limited capacity, the learned $\alpha(\mathbf{x})$ can reduce reliance on the cache and fall back to the material estimator.

4.3 Known lighting material optimization

Convergence in a high-variance scene. Figure 8 evaluates optimization in the Veach Ajar scene under known lighting. To isolate convergence effects in a noisy global-illumination setting, we report RGB-space MSE of the painting texture (rather than full-scene texture error). Both our method and Hadadan run at 1 spp during optimization. Our blended estimator converges faster than pure path tracing at low spp by reducing variance and stabilizing gradients.

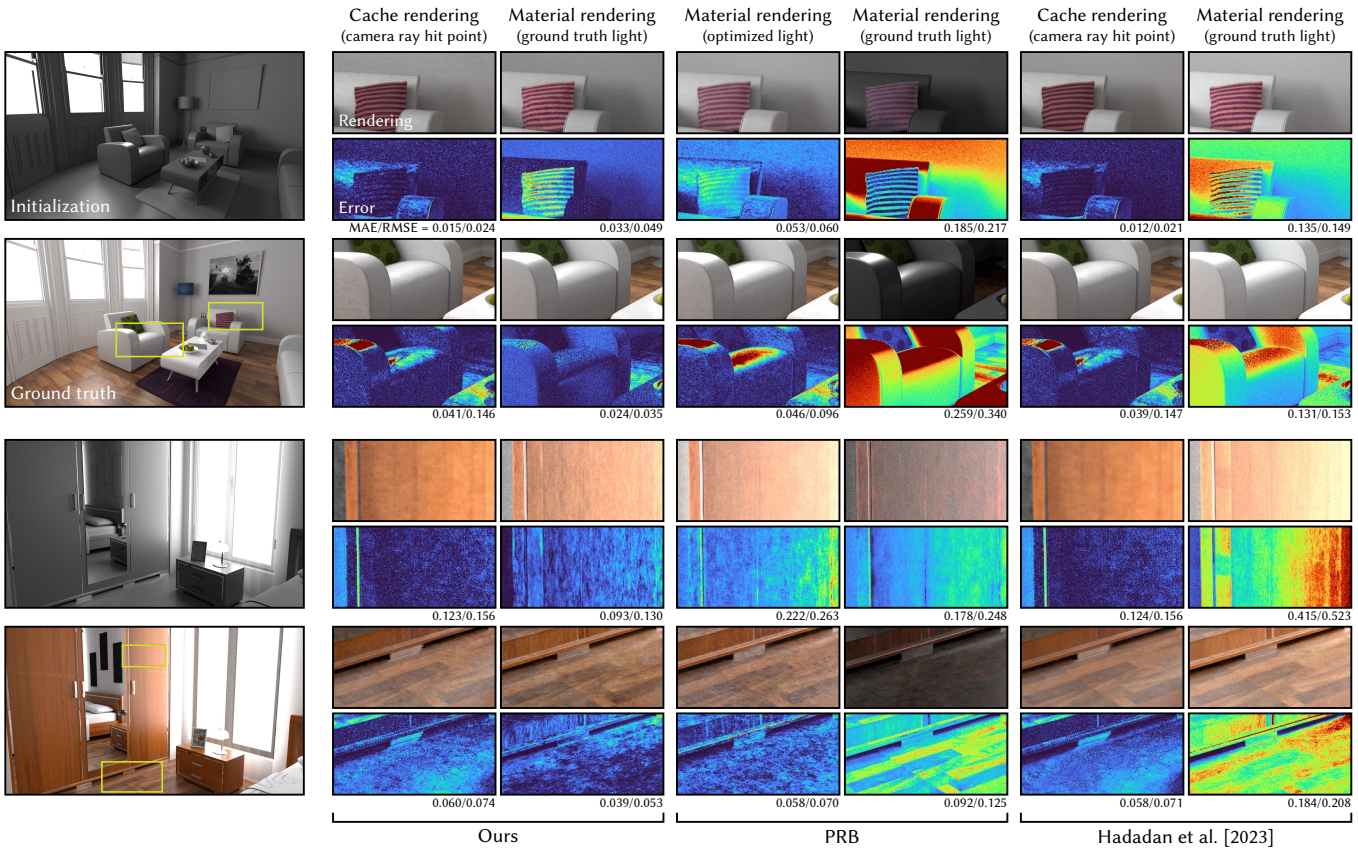


Fig. 5. **Material reconstruction from unknown lighting.** Visual comparison corresponding to Table 1. For ours and Hadadan et al. [2023], we show the cache rendering at camera-ray intersections and a *material-only* rendering illuminated by the ground-truth lighting (unknown during optimization) to verify decomposition. PRB optimizes both materials and emission; we show its optimized-light rendering (used during training) and its material-only rendering under ground-truth lighting.

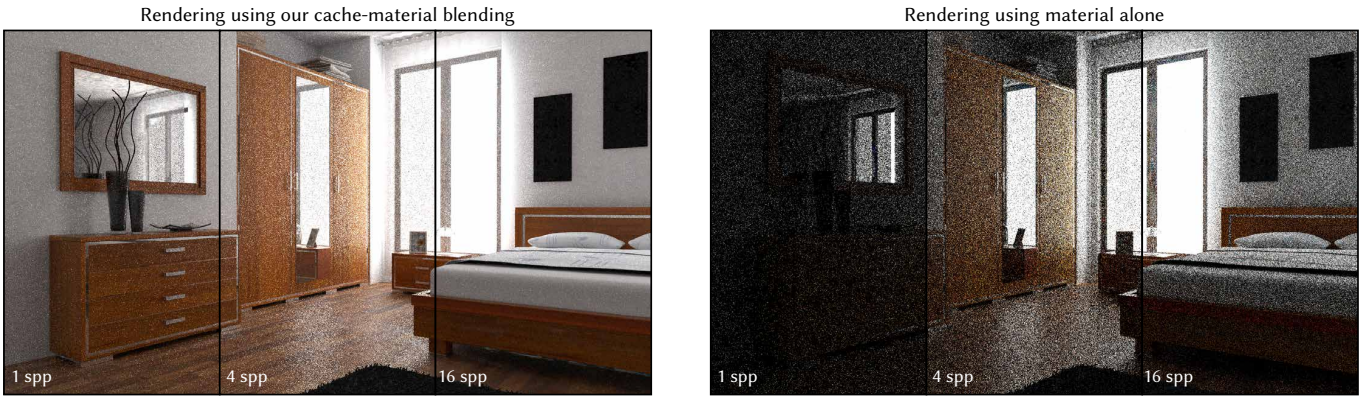


Fig. 6. **Blending reduces rendering variance.** We render the same optimized scene at low sample count via (left) our blended estimator and (right) the material estimator (without cache/blending field). Blending substantially reduces variance at equal sample count, which translates into cleaner optimization gradients.

Scene	PSNR ↑			SSIM ↑			LPIPS ↓		
	Ours	$\alpha = 0.5$	$\alpha = 0$	Ours	$\alpha = 0.5$	$\alpha = 0$	Ours	$\alpha = 0.5$	$\alpha = 0$
Bedroom	**24.85**	23.12	12.07	**0.5225**	0.5158	0.3955	0.2190	**0.2183**	0.3653
Bathroom2	**23.43**	16.54	11.73	0.8032	**0.8132**	0.7609	**0.1254**	0.1518	0.2766
Kitchen	**26.96**	23.58	10.62	**0.6724**	0.6355	0.4519	**0.2154**	0.2473	0.4127
Living-room	**22.64**	18.54	9.76	**0.8068**	0.7399	0.5095	**0.1257**	0.1563	0.3283
Living-room-2	**27.56**	18.75	8.47	**0.8331**	0.7808	0.6138	**0.1187**	0.1554	0.3368
Living-room-3	**29.57**	28.74	14.61	**0.8769**	0.8750	0.7771	**0.0721**	0.0728	0.1939
Staircase	**20.71**	16.52	6.98	**0.8350**	0.7569	0.4229	**0.1184**	0.1291	0.4826

Table 2. **Effect of optimizing the blending field.** Ablation under unknown lighting using the same evaluation as Table 1 (material-only relighting). We compare learning $\alpha(\mathbf{x})$ (ours) to fixing $\alpha(\mathbf{x}) = 0.5$ and disabling blending altogether ($\alpha(\mathbf{x}) = 0$). Learning $\alpha(\mathbf{x})$ provides consistent gains.

Scene	PSNR ↑	
	Learned α	Roughness α
Bathroom2	**22.99**	15.96
Bedroom	**24.82**	20.71
Kitchen	**26.98**	19.99
Living-room	**24.34**	16.37
Living-room-2	**27.60**	18.24
Living-room-3	**29.60**	26.77
Staircase	**20.72**	10.77
Average	**25.29**	18.40

Table 3. **Learned versus roughness-derived blending.** We compare our learned blending field to a fixed local heuristic $\alpha(\mathbf{x}) = 0.45\, r_{\text{eff}}(\mathbf{x})$, where r_{eff} is an effective material roughness in $[0, 1]$. The roughness heuristic preserves a material-gradient path but cannot model cache reliability, visibility, illumination, or optimization state; learning $\alpha(\mathbf{x})$ is therefore consistently more effective.

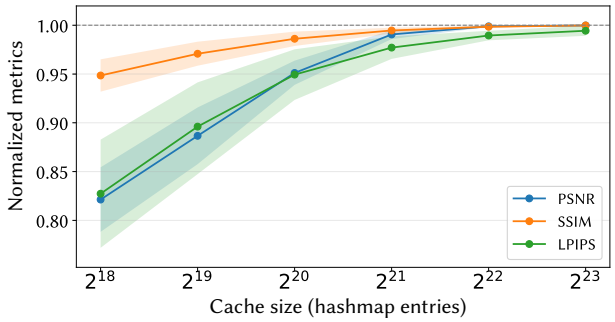


Fig. 7. **Sensitivity to cache capacity.** Scene-normalized mean performance across all scenes for PSNR/SSIM/LPIPS (best per scene = 100%) as a function of cache size (hash map entries, 2^{18}–2^{23}). Shaded bands indicate bootstrap confidence intervals. Our method is stable across a wide range of capacities, indicating robustness to cache representation size.

Limitation of material consistency loss. Figure 9 demonstrates a limitation of using the cache as a direct training target for materials via the material consistency loss (Equation 5). We optimize the albedo of a painting under known lighting while the cache represents outgoing radiance. For diffuse materials, limited cache capacity can blur fine details (the cache is trained on the scene scale); for glossy materials, directional variation is difficult for the cache to capture, and the resulting bias propagates to the material. This is why the glossy crops contain structured artifacts even though the underlying painting texture should remain spatially coherent: the material-consistency target contains cache approximation errors caused by view-dependent radiance. This experiment motivates why

Scene	Metric	Max light path depth				
		2	3	4	5	6
Bedroom	PSNR ↑	20.48	23.67	24.71	**24.85**	**24.85**
	SSIM ↑	0.5036	0.5169	0.5213	0.5222	**0.5225**
	LPIPS ↓	0.2403	0.2233	0.2195	0.2191	**0.2190**
Bathroom2	PSNR ↑	14.03	16.82	**24.57**	23.81	23.45
	SSIM ↑	0.8017	**0.8091**	0.8089	0.8050	0.8033
	LPIPS ↓	0.2043	0.1666	**0.1188**	0.1232	0.1253
Kitchen	PSNR ↑	25.45	**27.25**	27.07	26.97	26.97
	SSIM ↑	0.6506	0.6689	0.6714	0.6719	**0.6724**
	LPIPS ↓	0.2362	0.2184	0.2164	0.2160	**0.2154**
Living-rm	PSNR ↑	22.46	22.60	22.59	22.59	**22.64**
	SSIM ↑	0.8025	0.8061	0.8060	0.8058	**0.8067**
	LPIPS ↓	0.1270	0.1271	0.1262	0.1261	**0.1257**
Living-rm-2	PSNR ↑	21.44	27.38	27.44	27.51	**27.56**
	SSIM ↑	0.8124	0.8330	0.8330	**0.8331**	**0.8331**
	LPIPS ↓	0.1386	0.1190	**0.1187**	**0.1187**	**0.1187**
Living-rm-3	PSNR ↑	**35.06**	33.86	27.44	29.75	29.67
	SSIM ↑	0.8836	0.8813	**0.8840**	0.8774	0.8771
	LPIPS ↓	0.0672	0.0685	**0.0665**	0.0718	0.0719
Staircase	PSNR ↑	15.28	18.98	20.25	20.59	**20.67**
	SSIM ↑	0.7360	0.8012	0.8300	0.8338	**0.8346**
	LPIPS ↓	0.1277	0.1333	0.1223	0.1194	**0.1188**

Table 4. **Effect of maximum light-path depth.** Unknown-lighting training with our full method; evaluation follows Table 1. Increasing depth from 2 to 3 typically improves results by enabling $\alpha(\mathbf{x})$-guided termination and deeper transport, while gains beyond moderate depth often plateau.

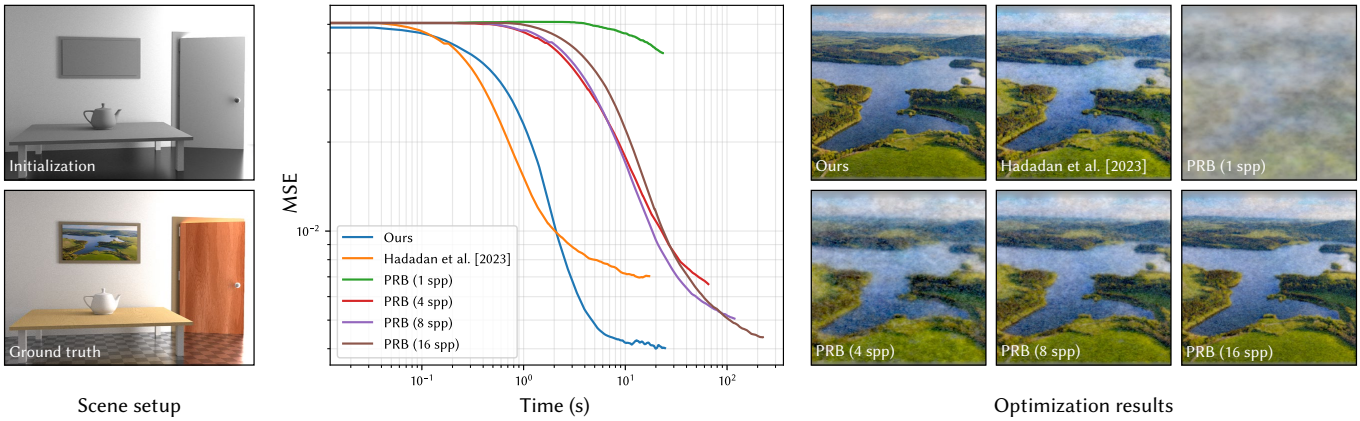


Fig. 8. **Texture optimization in a noisy scene with known lighting.** We track convergence of the painting on the wall while jointly optimizing all scene materials. Curves show MSE vs wall-clock time for ours, Hadadan et al. [2023] (1 spp), and PRB at 1/4/8/16 spp. All runs use 1000 iterations and a relative L_2 loss to avoid overfitting to noise.

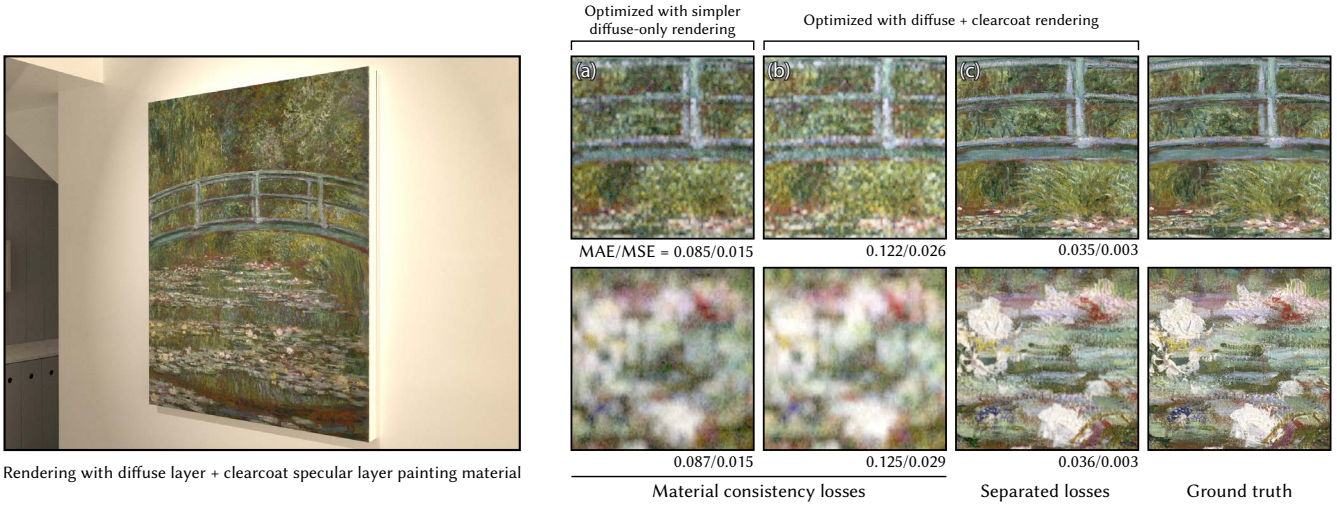


Fig. 9. **Limitation of material consistency loss.** We optimize the albedo of a painting under known lighting using the material consistency loss (Equation 5), which uses the cache as the training target. **(a)** Diffuse surface: limited cache resolution blurs fine spatial details, degrading the recovered texture. **(b-c)** Glossy surface: the cache struggles with high-frequency view-dependent variation, and bias propagates to the material, causing more artifacts. The artifacts are most visible in the glossy variants, where the optimized albedo inherits view-dependent cache errors that should instead be explained by directional scattering. We visualize the optimized albedo to isolate this failure from lighting changes. *Bridge over a Pond of Water Lilies*, Claude Monet, 1899, public domain.

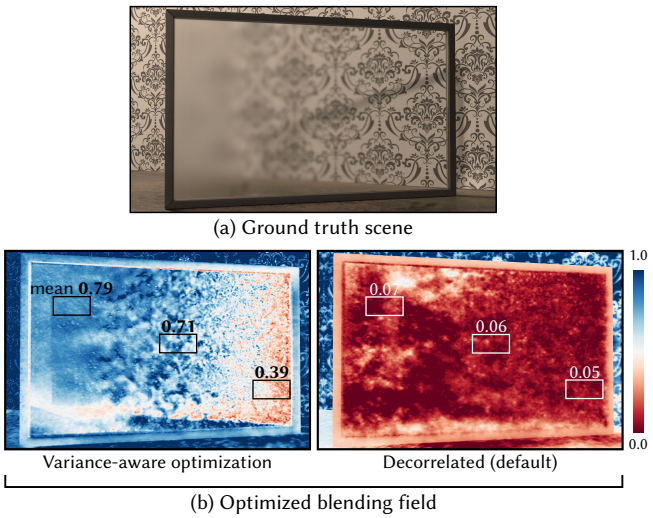
our main approach relies on separated losses to constrain estimators without requiring the cache to serve as a universally accurate direct supervision signal for outgoing radiance.

4.4 Variance-aware optimization of $\alpha(\mathbf{x})$

The results in this article optimize the blending field $\alpha(\mathbf{x})$ to minimize bias (i.e., using the decorrelated gradient estimator). Figure 10 explores an alternative variance-aware objective that additionally penalizes variance. On a rough dielectric surface with spatially varying roughness, variance-aware optimization drives $\alpha(\mathbf{x})$ higher overall (favoring the cache more), while still preferring the material estimator in glossy regions where cache bias is pronounced. This suggests variance-aware objectives may improve robustness in extremely noisy settings; we leave a broader evaluation for future work.

4.5 Optimization progress

Figure 11 visualizes how materials, the cache, and the blending field evolve during unknown-lighting optimization. We show (top) a *material-only* render relit with ground-truth lighting (unknown during optimization), (middle) cache values at camera-ray intersections, and (bottom) the learned $\alpha(\mathbf{x})$. As optimization progresses, both the material-only renders and cache predictions become more



(a) Ground truth scene

(b) Optimized blending field

Fig. 10. **Effect of variance-aware optimization of $\alpha(\mathbf{x})$.** (a) We optimize the blending field α on a rough dielectric with spatially varying roughness (left: diffuse, right: glossy). (b) Comparison of $\alpha(\mathbf{x})$ learned by default bias-driven (decorrelated) optimization versus variance-aware optimization. Variance-aware optimization favors the cache more overall while still reducing cache reliance in glossy regions where cache bias is pronounced.

consistent with the reference, while $\alpha(\mathbf{x})$ adapts over time to route gradients to the more reliable estimator.

4.6 Performance analysis

Our method incurs only a negligible computational overhead compared to PRB. Across all scenes, our method requires an average of 0.094 seconds per iteration, compared to 0.091 for PRB and 0.088 for Hadadan et al. [2023]. Cache evaluation and blending-field queries are efficient, and the small overhead is justified by the substantial improvement in reconstruction quality under unknown lighting. Detailed per-scene timing is provided in Table 5.

5 Conclusion

Inverse rendering involves a choice between two unsatisfying extremes. On one side lie radiance field representations that soak up data like a sponge, but sidestep the question of what in the world is actually emitting or reflecting. On the other side lie physically based models that promise semantic, editable parameters, but punish us with high-variance simulation and an unforgiving loss landscape that can be impossible to optimize.

By creating a slider that interpolates between these two views, we have surprisingly not compromised on physics but instead ended up with something richer: a place to absorb what the model cannot yet physically explain, which makes the reconstruction task better-posed and more efficient. We demonstrated this on material recovery with unknown lighting, a hard problem where inverse path tracing often fails outright. Our observations suggest that radiance caches may have a role to play in future inverse rendering systems.

The assumption of known geometry is somewhat artificial, as we will often want to recover it jointly with materials and lighting. It will be interesting to study how to best introduce a cache and blending field if geometry itself can nucleate, evolve, or disappear, whether represented by triangle meshes, messy triangle-soup geometry, or heterogeneous volumes. Subsurface scattering is another interesting application, since it is high-variance and requires long light paths that could benefit from early termination.

Our experimental setup is also synthetic in another sense: all tasks were designed so that the model can perfectly reproduce the data, which is never realistic in practice. When attempting to reconstruct scenes from photos, real-world appearance will inevitably lie outside the modeling capability of any BSDF, and so there is an unavoidable source of discrepancy. Investigating whether our framework or a similar decomposition could help in this case is an interesting direction for future work.

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

Advanced Micro Devices AMD. 2025. AMD FSR Radiance Caching. https://gpuopen.com/amd-fsr-radiancecaching/. [Online; accessed 14-January-2026].

Benjamin Attal, Dor Verbin, Ben Mildenhall, Peter Hedman, Jonathan T Barron, Matthew O'Toole, and Pratul P Srinivasan. 2024. Flash Cache: Reducing Bias in Radiance Cache Based Inverse Rendering. In *European Conference on Computer Vision (ECCV)*. Springer, 20–36.

Dejan Azinović, Tzu-Mao Li, Anton Kaplanyan, and Matthias Nießner. 2019. Inverse Path Tracing for Joint Material and Lighting Estimation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. doi:10.1109/CVPR.2019.00255

Sai Praveen Bangaru, Tzu-Mao Li, and Frédo Durand. 2020. Unbiased warped-area sampling for differentiable rendering. *ACM Trans. Graph.* 39, 6, Article 245 (November 2020), 18 pages. doi:10.1145/3414685.3417833

Yash Belhe, Bing Xu, Sai Praveen Bangaru, Ravi Ramamoorthi, and Tzu-Mao Li. 2024. Importance Sampling BRDF Derivatives. *ACM Trans. Graph.* 43, 3, Article 25 (April 2024), 21 pages. doi:10.1145/3648611

Nikolaus Binder, Sascha Fricke, and Alexander Keller. 2018. Fast path space filtering by jittered spatial hashing. In *ACM SIGGRAPH 2018 Talks* (Vancouver, British Columbia, Canada) *(SIGGRAPH '18)*. Association for Computing Machinery, New York, NY, USA, Article 71, 2 pages. doi:10.1145/3214745.3214806

Wesley Chang, Venkataram Sivaram, Derek Nowrouzezahrai, Toshiya Hachisuka, Ravi Ramamoorthi, and Tzu-Mao Li. 2023. Parameter-space ReSTIR for Differentiable and Inverse Rendering. In *ACM SIGGRAPH 2023 Conference Proceedings* (Los Angeles, CA, USA) *(SIGGRAPH '23)*. Association for Computing Machinery, New York, NY, USA, Article 18, 10 pages. doi:10.1145/3588432.3591512

Wesley Chang, Xuanda Yang, Yash Belhe, Ravi Ramamoorthi, and Tzu-Mao Li. 2024. Spatiotemporal Bilateral Gradient Filtering for Inverse Rendering. In *SIGGRAPH Asia 2024 Conference Papers* (Tokyo, Japan) *(SA '24)*. Association for Computing Machinery, New York, NY, USA, Article 70, 11 pages. doi:10.1145/3680528.3687606

Mikhail Dereviannykh, Dmitrii Klepikov, Johannes Hanika, and Carsten Dachsbacher. 2025. Neural Two-Level Monte Carlo Real-Time Rendering. In *Computer Graphics Forum*. Wiley Online Library, e70050.

Zhimin Fan, Pengcheng Shi, Mufan Guo, Ruoyu Fu, Yanwen Guo, and Jie Guo. 2024. Conditional Mixture Path Guiding for Differentiable Rendering. *ACM Trans. Graph.* 43, 4, Article 48 (July 2024), 11 pages. doi:10.1145/3658133

Ioannis Gkioulekas, Anat Levin, and Todd Zickler. 2016. An evaluation of computational imaging techniques for heterogeneous inverse scattering. In *European Conference on Computer Vision (ECCV)*. Springer International Publishing, 685–701. doi:10.1007/978-3-319-46487-9_42

G. Greger, P. Shirley, P.M. Hubbard, and D.P. Greenberg. 1998. The irradiance volume. *IEEE Computer Graphics and Applications* 18, 2 (1998), 32–43. doi:10.1109/38.656788

Chun Gu, Xiaofei Wei, Zixuan Zeng, Yuxuan Yao, and Li Zhang. 2025. IRGS: Inter-Reflective Gaussian Splatting with 2D Gaussian Ray Tracing. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Saeed Hadadan, Geng Lin, Jan Novák, Fabrice Rousselle, and Matthias Zwicker. 2023. Inverse Global Illumination using a Neural Radiometric Prior. In *ACM SIGGRAPH 2023 Conference Proceedings* (Los Angeles, CA, USA) *(SIGGRAPH '23)*. Association for Computing Machinery, New York, NY, USA, Article 17, 11 pages. doi:10.1145/3588432.3591553

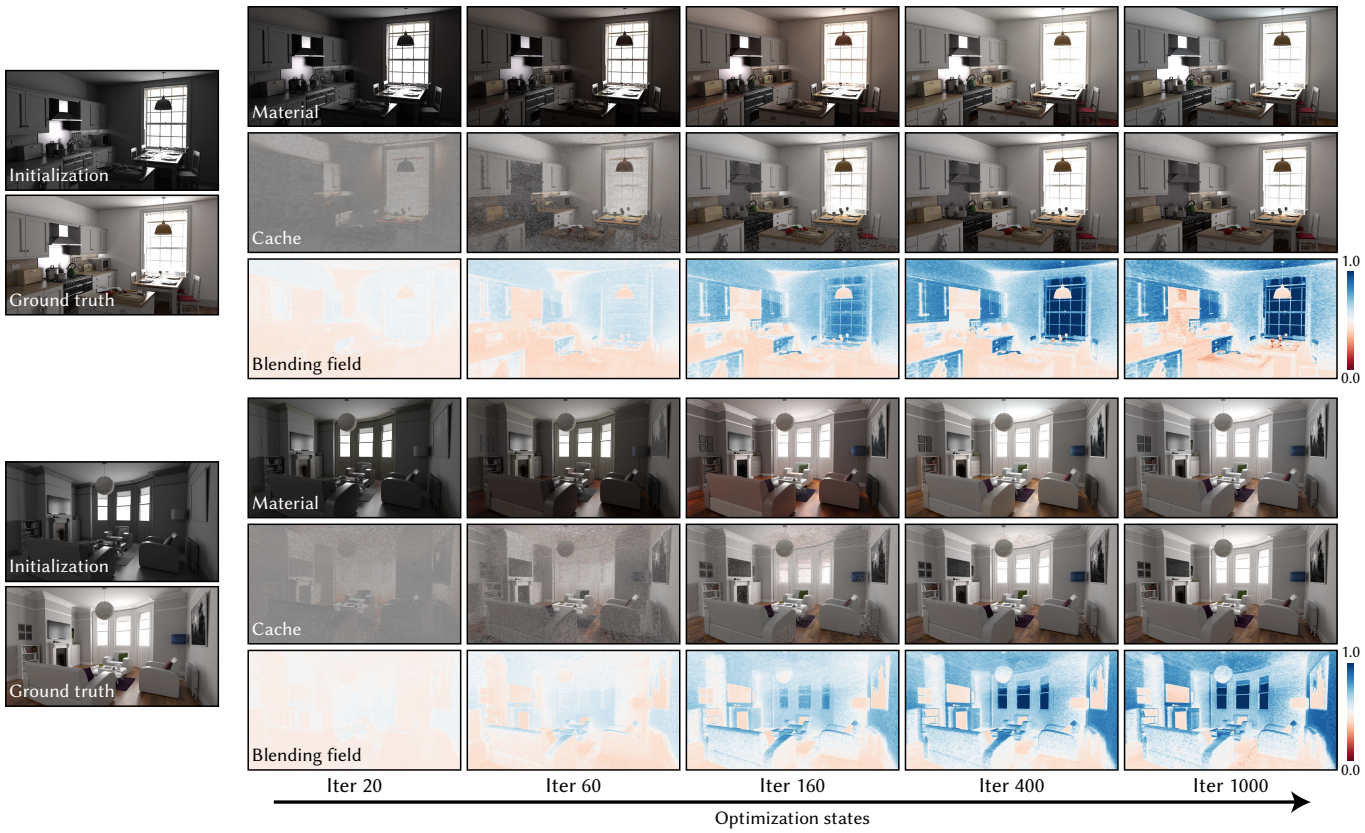


Fig. 11. **Evolution of materials, cache, and blending field during unknown-lighting optimization.** We visualize selected iterations for each scene: (top) *material-only* rendering relit with ground-truth lighting (unknown during optimization), (middle) cache queried at camera-ray intersections, and (bottom) learned blending field $\alpha(\mathbf{x})$. The blending field adapts over time, increasingly routing gradients to the more reliable estimator as optimization progresses.

Saeed Hadadan and Matthias Zwicker. 2024. Neural Differential Radiance Field: Learning the Differential Space Using a Neural Network. In *Advances in Computer Graphics*, Bin Sheng, Lei Bi, Jinman Kim, Nadia Magnenat-Thalmann, and Daniel Thalmann (Eds.). Springer Nature Switzerland, Cham, 93–104.

Binbin Huang, Zehao Yu, Anpei Chen, Andreas Geiger, and Shenghua Gao. 2024. 2D Gaussian Splatting for Geometrically Accurate Radiance Fields. In *ACM SIGGRAPH 2024 Conference Papers* (Denver, CO, USA) *(SIGGRAPH '24)*. Association for Computing Machinery, New York, NY, USA, Article 32, 11 pages. doi:10.1145/3641519.3657428

Wojciech Jarosz, Craig Donner, Matthias Zwicker, and Henrik Wann Jensen. 2008. Radiance caching for participating media. *ACM Trans. Graph.* 27, 1, Article 7 (March 2008), 11 pages. doi:10.1145/1330511.1330518

Kaiwen Jiang, Jia-Mu Sun, Zilu Li, Dan Wang, Tzu-Mao Li, and Ravi Ramamoorthi. 2025. Differentiable Light Transport with Gaussian Surfels via Adapted Radiosity for Efficient Relighting and Geometry Reconstruction. *ACM Trans. Graph.* 44, 6, Article 210 (December 2025), 25 pages. doi:10.1145/3763305

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 *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Lukas Kandlbinder, Addis Dittebrandt, Alexander Schipek, and Carsten Dachsbacher. 2024. Optimizing Path Termination for Radiance Caching Through Explicit Variance Trading. *Proc. ACM Comput. Graph. Interact. Tech.* 7, 3, Article 33 (Aug. 2024), 19 pages. doi:10.1145/3675381

Bernhard Kerbl, Georgios Kopanas, Thomas Leimkuehler, and George Drettakis. 2023. 3D Gaussian Splatting for Real-Time Radiance Field Rendering. *ACM Trans. Graph.* 42, 4, Article 139 (July 2023), 14 pages. doi:10.1145/3592433

Pramook Khungurn, Daniel Schroeder, Shuang Zhao, Kavita Bala, and Steve Marschner. 2015. Matching Real Fabrics with Micro-Appearance Models. *ACM Trans. Graph.* 35, 1, Article 1 (December 2015), 26 pages. doi:10.1145/2818648

J. Krivanek, P. Gautron, S. Pattanaik, and K. Bouatouch. 2005. Radiance caching for efficient global illumination computation. *IEEE Transactions on Visualization and Computer Graphics* 11, 5 (2005), 550–561. doi:10.1109/TVCG.2005.83

Jaroslav Křivánek, Pascal Gautron, Greg Ward, Okan Arikan, and Henrik Wann Jensen. 2007. Practical global illumination with irradiance caching. In *ACM SIGGRAPH 2007 Courses* (San Diego, California) *(SIGGRAPH '07)*. Association for Computing Machinery, New York, NY, USA, 1–es. doi:10.1145/1281500.1281617

Jaakko Lehtinen, Jacob Munkberg, Jon Hasselgren, Samuli Laine, Tero Karras, Miika Aittala, and Timo Aila. 2018. Noise2Noise: Learning Image Restoration without Clean Data. In *Proceedings of the 35th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 80)*, Jennifer Dy and Andreas Krause (Eds.). PMLR, 2965–2974. https://proceedings.mlr.press/v80/lehtinen18a.html

Tzu-Mao Li, Miika Aittala, Frédo Durand, and Jaakko Lehtinen. 2018. Differentiable Monte Carlo ray tracing through edge sampling. *ACM Trans. Graph.* 37, 6, Article 222 (December 2018), 11 pages. doi:10.1145/3272127.3275109

Jingwang Ling, Ruihan Yu, Feng Xu, Chun Du, and Shuang Zhao. 2024. NeRF as a Non-Distant Environment Emitter in Physics-based Inverse Rendering. In *ACM SIGGRAPH 2024 Conference Papers* (Denver, CO, USA) *(SIGGRAPH '24)*. Association for Computing Machinery, New York, NY, USA, Article 39, 12 pages. doi:10.1145/3641519.3657404

Guillaume Loubet, Nicolas Holzschuch, and Wenzel Jakob. 2019. Reparameterizing discontinuous integrands for differentiable rendering. *ACM Trans. Graph.* 38, 6, Article 228 (November 2019), 14 pages. doi:10.1145/3355089.3356510

Haolin Lu, Delio Vicini, Wesley Chang, and Tzu-Mao Li. 2025. Vector-Valued Monte Carlo Integration Using Ratio Control Variates. *ACM Trans. Graph.* 44, 4, Article 100 (July 2025), 16 pages. doi:10.1145/3731175

Julio Marco, Adrian Jarabo, Wojciech Jarosz, and Diego Gutierrez. 2018. Second-Order Occlusion-Aware Volumetric Radiance Caching. *ACM Trans. Graph.* 37, 2, Article 20 (July 2018), 14 pages. doi:10.1145/3185225

Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. 2020. NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis. In *European Conference on Computer Vision (ECCV)*.

Thomas Müller, Fabrice Rousselle, Jan Novák, and Alexander Keller. 2021. Real-time neural radiance caching for path tracing. *ACM Trans. Graph.* 40, 4, Article 36 (July 2021), 16 pages. doi:10.1145/3450626.3459812

Baptiste Nicolet, Alec Jacobson, and Wenzel Jakob. 2021. Large steps in inverse rendering of geometry. *ACM Trans. Graph.* 40, 6, Article 248 (December 2021), 13 pages. doi:10.1145/3478513.3480501

Baptiste Nicolet, Fabrice Rousselle, Jan Novak, Alexander Keller, Wenzel Jakob, and Thomas Müller. 2023. Recursive Control Variates for Inverse Rendering. *ACM Trans. Graph.* 42, 4, Article 62 (July 2023), 13 pages. doi:10.1145/3592139

Merlin Nimier-David. 2022. *Differentiable Physically Based Rendering: Algorithms, Systems and Applications.* PhD Dissertation. École Polytechnique Fédérale de Lausanne (EPFL), Lausanne, Switzerland. https://merl.in/phd-thesis.pdf Thesis No. 9123.

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. *ACM Trans. Graph.* 39, 4, Article 146 (August 2020), 15 pages. doi:10.1145/3386569.3392406

Merlin Nimier-David, Delio Vicini, Tizian Zeltner, and Wenzel Jakob. 2019. Mitsuba 2: a retargetable forward and inverse renderer. *ACM Trans. Graph.* 38, 6, Article 203 (November 2019), 17 pages. doi:10.1145/3355089.3356498

NVIDIA. 2024. NVIDIA RTXGI. https://github.com/NVIDIAGameWorks/RTXGI [Online; accessed 14-January-2026].

Yohan Poirier-Ginter, Jeffrey Hu, Jean-Francois Lalonde, and George Drettakis. 2025. Editable Physically-based Reflections in Raytraced Gaussian Radiance Fields. In *Proceedings of the SIGGRAPH Asia 2025 Conference Papers (SA Conference Papers '25).* Association for Computing Machinery, New York, NY, USA, Article 201, 12 pages. doi:10.1145/3757377.3763971

Ari Silvennoinen and Jaakko Lehtinen. 2017. Real-time global illumination by precomputed local reconstruction from sparse radiance probes. *ACM Trans. Graph.* 36, 6, Article 230 (November 2017), 13 pages. doi:10.1145/3130800.3130852

Jiakai Sun, Weijing Zhang, Zhanjie Zhang, Tianyi Chu, Guangyuan Li, Lei Zhao, and Wei Xing. 2025. Joint Optimization of Triangle Mesh, Material, and Light from Neural Fields with Neural Radiance Cache. arXiv:2305.16800 [cs.GR]

Wolfgang Tatzgern, Alexander Weinrauch, Pascal Stadlbauer, Joerg H. Mueller, Martin Winter, and Markus Steinberger. 2024. Radiance Caching with On-Surface Caches for Real-Time Global Illumination. *Proc. ACM Comput. Graph. Interact. Tech.* 7, 3, Article 38 (Aug. 2024), 17 pages. doi:10.1145/3675382

Zdravko Velinov, Marios Papas, Derek Bradley, Paulo Gotardo, Parsa Mirdehghan, Steve Marschner, Jan Novák, and Thabo Beeler. 2018. Appearance capture and modeling of human teeth. *ACM Trans. Graph.* 37, 6, Article 207 (December 2018), 13 pages. doi:10.1145/3272127.3275098

Delio Vicini, Sébastien Speierer, and Wenzel Jakob. 2021. Path replay backpropagation: differentiating light paths using constant memory and linear time. *ACM Trans. Graph.* 40, 4, Article 108 (July 2021), 14 pages. doi:10.1145/3450626.3459804

Yu-Chen Wang, Chris Wyman, Lifan Wu, and Shuang Zhao. 2023. Amortizing Samples in Physics-Based Inverse Rendering Using ReSTIR. *ACM Trans. Graph.* 42, 6, Article 214 (December 2023), 17 pages. doi:10.1145/3618331

Gregory J. Ward, Francis M. Rubinstein, and Robert D. Clear. 1988. A ray tracing solution for diffuse interreflection. *SIGGRAPH Comput. Graph.* 22, 4 (June 1988), 85–92. doi:10.1145/378456.378490

Philippe Weier, Marc Droske, Johannes Hanika, Andrea Weidlich, and Jiří Vorba. 2021. Optimised Path Space Regularisation. *Computer Graphics Forum (Proc. EGSR)* 40, 4 (2021). doi:10.1111/cgf.14347

Philippe Weier, Jérémy Riviere, Ruslan Guseinov, Stephan Garbin, Philipp Slusallek, Bernd Bickel, Thabo Beeler, and Delio Vicini. 2025. Practical Inverse Rendering of Textured and Translucent Appearance. *ACM Trans. Graph.* 44, 4, Article 117 (July 2025), 16 pages. doi:10.1145/3730855

Haoqian Wu, Zhipeng Hu, Lincheng Li, Yongqiang Zhang, Changjie Fan, and Xin Yu. 2023. NeFII: Inverse Rendering for Reflectance Decomposition with Near-Field Indirect Illumination. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR).* doi:10.1109/CVPR52729.2023.00418

Sean Wu, Shamik Basu, Tim Broedermann, Luc Van Gool, and Christos Sakaridis. 2025. PBR-NeRF: Inverse Rendering with Physics-Based Neural Fields. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR).*

Kai Yan, Vincent Pegoraro, Marc Droske, Jiří Vorba, and Shuang Zhao. 2024. Differentiating Variance for Variance-Aware Inverse Rendering. In *SIGGRAPH Asia 2024 Conference Papers* (Tokyo, Japan) *(SA '24).* Association for Computing Machinery, New York, NY, USA, Article 117, 10 pages. doi:10.1145/3680528.3687603

Yao Yao, Jingyang Zhang, Jingbo Liu, Yihang Qu, Tian Fang, David McKinnon, Yanghai Tsin, and Long Quan. 2022. NeILF: Neural Incident Light Field for Physically-based Material Estimation. In *European Conference on Computer Vision (ECCV).*

Tizian Zeltner, Sébastien Speierer, Iliyan Georgiev, and Wenzel Jakob. 2021. Monte Carlo estimators for differential light transport. *ACM Trans. Graph.* 40, 4, Article 78 (July 2021), 16 pages. doi:10.1145/3450626.3459807

Cheng Zhang, Lifan Wu, Changxi Zheng, Ioannis Gkioulekas, Ravi Ramamoorthi, and Shuang Zhao. 2019. A differential theory of radiative transfer. *ACM Trans. Graph.* 38, 6, Article 227 (November 2019), 16 pages. doi:10.1145/3355089.3356522

Jingyang Zhang, Yao Yao, Shiwei Li, Jingbo Liu, Tian Fang, David McKinnon, Yanghai Tsin, and Long Quan. 2023b. NeILF++: Inter-reflectable Light Fields for Geometry and Material Estimation. In *IEEE International Conference on Computer Vision (ICCV).*

Xiuming Zhang, Pratul P. Srinivasan, Boyang Deng, Paul Debevec, William T. Freeman, and Jonathan T. Barron. 2021. NeRFactor: neural factorization of shape and reflectance under an unknown illumination. *ACM Trans. Graph.* 40, 6, Article 237 (December 2021), 18 pages. doi:10.1145/3478513.3480496

Yuanqing Zhang, Jiaming Sun, Xingyi He, Huan Fu, Rongfei Jia, and Xiaowei Zhou. 2022. Modeling Indirect Illumination for Inverse Rendering. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR).* doi:10.1109/CVPR52688.2022.01809

Ziyi Zhang, Nicolas Roussel, and Wenzel Jakob. 2023a. Projective Sampling for Differentiable Rendering of Geometry. *ACM Trans. Graph.* 42, 6, Article 212 (Dec. 2023), 14 pages. doi:10.1145/3618385

Ziyi Zhang, Nicolas Roussel, Thomas Muller, Tizian Zeltner, Merlin Nimier-David, Fabrice Rousselle, and Wenzel Jakob. 2025. Radiance Surfaces: Optimizing Surface Representations with a 5D Radiance Field Loss. In *Proceedings of the Special Interest Group on Computer Graphics and Interactive Techniques Conference Conference Papers (SIGGRAPH Conference Papers '25).* Association for Computing Machinery, New York, NY, USA, Article 21, 10 pages. doi:10.1145/3721238.3730713

Shuang Zhao, Lifan Wu, Frédo Durand, and Ravi Ramamoorthi. 2016. Downsampling scattering parameters for rendering anisotropic media. *ACM Trans. Graph.* 35, 6, Article 166 (December 2016), 11 pages. doi:10.1145/2980179.2980228

Appendix

A Decorrelated gradient estimator

In Section 3.1.2, we claimed that naively optimizing the material consistency loss leads to biased gradients. The issue has the same root as in differentiating Monte Carlo renderers with noisy renderings, and we adopt a similar decorrelation strategy as used by Azinović et al. [2019].

Consider the material consistency loss at a single bounce. For brevity, let $C = C(\mathbf{x}, \omega)$ be the target outgoing radiance and $C'(\omega') = C(\mathbf{x}', -\omega')$ be the incident radiance from the cache. The loss is:

$$\mathcal{L} = \left(C - \mathbb{E}_{\omega'} \left[C'(\omega') \cdot f(\omega') \cdot \frac{\cos\theta'}{p(\omega')} \right] \right)^2, \tag{7}$$

where C and C' are treated as known (from the detached cache), and we optimize the BSDF $f(\omega') \equiv f(\mathbf{x}, \omega', \omega)$. Let $R = C - \mathbb{E}[C' \cdot f \cdot \cos\theta'/p]$ denote the residual. The true gradient is

$$\nabla_f \mathcal{L} = -2\, R \cdot \mathbb{E}\left[\nabla_f \left(C' \cdot f \cdot \frac{\cos\theta'}{p} \right) \right]. \tag{8}$$

Naive estimator (biased). If we use the same sample ω'_1 for both the residual and the gradient, we obtain

$$\widehat{\nabla_f \mathcal{L}} = -2 \left(C - C'(\omega'_1) \cdot f(\omega'_1) \cdot \frac{\cos\theta'_1}{p_1} \right) \cdot \nabla_f \left(C'(\omega'_1) \cdot f(\omega'_1) \cdot \frac{\cos\theta'_1}{p_1} \right). \tag{9}$$

The two factors are correlated through the shared sample ω'_1, so $\mathbb{E}[A \cdot B] \neq \mathbb{E}[A] \cdot \mathbb{E}[B]$. This estimator is biased: in the extreme one-sample case, we are asking each sampled direction to individually match C, when in fact only the integral over all directions should match C.

Decorrelated estimator (unbiased). To obtain an unbiased gradient, we sample two independent directions: ω_1' for the residual and ω_2' for the gradient:

$$\widehat{\nabla_f \mathcal{L}} = -2\left(C - C'(\omega_1') \cdot f(\omega_1') \cdot \frac{\cos\theta_1'}{p_1}\right) \cdot \nabla_f \left(C'(\omega_2') \cdot f(\omega_2') \cdot \frac{\cos\theta_2'}{p_2}\right). \quad (10)$$

Since ω_1' and ω_2' are independent, the expectation factorizes correctly:

$$\mathbb{E}\left[\widehat{\nabla_f \mathcal{L}}\right] = -2\,\mathbb{E}[R] \cdot \mathbb{E}\left[\nabla_f \left(C' \cdot f \cdot \frac{\cos\theta'}{p}\right)\right] = \nabla_f \mathcal{L}. \quad (11)$$

B Alpha optimization via blended termination losses

In Section 3.3, we described optimizing α by differentiating the blended image loss. Here we discuss an alternative approach that optimizes α more directly, without additional overhead. The idea, used by Zhang et al. [2025], is to define two losses at each bounce—one assuming we terminate with the cache, one assuming we continue with the material—and let α blend between them.

At each bounce i along a light path, we compare the pixel color against the radiance estimate obtained by each strategy:

$$\mathcal{L}_{\text{cache},i} = (I - T_i \cdot C_i)^2, \quad (12)$$

$$\mathcal{L}_{\text{mat},i} = \left(I - T_i \cdot \frac{f_i \cos\theta_i}{p_i} L_{\text{o}}^{(i+1)}\right)^2, \quad (13)$$

where T_i is the path throughput from the camera to bounce i, and I is the reference pixel color. We then optimize α_i to minimize the blended loss:

$$\mathcal{L}_{\text{blend},i} = (1-\alpha_i)\,\mathcal{L}_{\text{mat},i} + \alpha_i\,\mathcal{L}_{\text{cache},i}. \quad (14)$$

Intuitively, α_i learns to favor whichever estimator yields a smaller loss at that point.

Optimizing alpha. This formulation gives unbiased gradients for α. To see why, observe that the gradient is simply a difference of two loss values:

$$\nabla_{\alpha_i} \mathcal{L}_{\text{blend},i} = \mathcal{L}_{\text{cache},i} - \mathcal{L}_{\text{mat},i}. \quad (15)$$

Since α_i sits outside the nonlinear (squared) part of the loss, there is no product of correlated terms. Each loss value is an unbiased estimate of its expectation, so:

$$\mathbb{E}\left[\nabla_{\alpha_i} \mathcal{L}_{\text{blend},i}\right] = \mathbb{E}\left[\mathcal{L}_{\text{cache},i}\right] - \mathbb{E}\left[\mathcal{L}_{\text{mat},i}\right]. \quad (16)$$

The gradient pushes α_i toward the estimator with lower expected loss.

Limitations for cache and material. One might hope to use these losses to optimize the cache or material parameters as well. However, this leads to biased gradients—even for the cache, which itself has no estimation variance.

The issue is the path throughput T_i. Consider the gradient for the cache:

$$\nabla_{C_i} \mathcal{L}_{\text{cache},i} = -2\,T_i \cdot (I - T_i \cdot C_i). \quad (17)$$

The throughput T_i appears twice: once as a direct multiplier and once inside the residual. Both instances are the same one-sample estimate, so they are correlated, and the expectation does not factorize correctly. This is the same bias issue discussed in Appendix A.

To obtain unbiased gradients, we would need two independent paths connecting the same pixel to the same point $\mathbf{x}_i$—one for the residual and one for the gradient multiplier. This is not efficient in practice, so we conclude that these blended termination losses should only be used to optimize α, not the cache or material parameters.

Bias-only vs. variance-aware. As with the image-space losses (Section 3.3), we can optimize α for bias alone or for combined bias and variance.

For variance-aware optimization, we compute $\mathcal{L}_{\text{cache},i}$ and $\mathcal{L}_{\text{mat},i}$ with a single sample. The expected squared error includes both bias and variance:

$$\mathbb{E}\left[(I - T \cdot R)^2\right] = \text{bias}^2 + \text{variance}, \quad (18)$$

so α is pushed toward the estimator with lower mean squared error.

For bias-only optimization, we split the squared loss into two terms estimated with independent samples A and B:

$$\mathcal{L}_{\text{cache},i} = \left(I - T_i^A \cdot C_i\right) \cdot \left(I - T_i^B \cdot C_i\right). \quad (19)$$

Since A and B are independent, the expectation factorizes:

$$\mathbb{E}\left[(I - T^A \cdot R)(I - T^B \cdot R)\right] = (I - \mathbb{E}[T \cdot R])^2 = \text{bias}^2, \quad (20)$$

removing the variance term from the optimization objective.

C Termination strategies for separated losses

The separated loss (Section 3.2) requires choosing where to terminate paths in each iteration. Here we describe strategies in detail.

Uniform depth sampling. The simplest approach samples a termination depth k uniformly from $\{1, \ldots, k_{\max}\}$ each iteration. All pixels share the same k, producing a coherent image for the loss. This baseline is easy to implement and works well in practice.

α-aware termination. Uniform sampling may terminate paths where α is low, indicating the cache is unreliable. To mitigate this, we extend paths beyond the sampled depth deterministically until reaching a more reliable region:

(1) Sample a minimum depth k uniformly at random.
(2) Starting from bounce k, compute the accumulated product $\prod_{i \geq k}(1 - \alpha_i)$.
(3) Terminate at the first bounce where this product drops below 0.5; no additional random number is drawn.

Intuitively, we continue until enough "trust the cache" has accumulated along the path.

Throughput-based termination. An alternative measures accumulated trust from the path start. We compute the product $(1-\alpha_1)(1-\alpha_2)\cdots(1-\alpha_{k-1})$, which represents how much the path has favored the material estimator. Each iteration, we draw a threshold $\tau \in [0, 1]$ and terminate at the first bounce where this product drops below τ. This naturally biases termination toward high-α regions.

Comparison. In our benchmarks, uniform depth sampling is approximately 0.4 dB lower than the α-aware strategies. The α-aware and throughput-based strategies perform comparably, with no distinguishable difference between them. We use α-aware termination for all main results.

Computing all losses simultaneously. Rather than sampling one termination depth, we can compute losses at all depths in a single pass by storing each $\hat{I}_k$ in a separate output buffer. The total loss is then $\sum_k w_k \mathcal{L}_k$ for some weights w_k. This avoids variance from sampling but requires $O(\text{depth})$ memory per pixel. We use the sampling approach in all experiments for its lower memory footprint.

Per-ray vs. per-iteration termination. The termination criterion should be decided per-iteration, not per-ray. If termination were decided per-ray—for example, terminating at bounce k with probability α_k— the expected image over infinite samples would recover the blended estimator from Equation 3, losing the direct supervision that separated losses provide.

D Additional details and results

Optimized BSDF parameters. We use a script to convert known scenes to trainable versions, where all materials are replaced with trainable ones initialized to be neutral gray. We replace reflectance-like slots of non-smooth BSDFs by replacing any existing reflectance (diffuse or specular) with trainable textures. IOR and conductor `eta`/`k` remain fixed, and smooth BSDFs (e.g., mirror or glass) are left untouched. In practice, this covers diffuse walls/paints, rough plastic and other plastic variants (wood, fabric, rubber, plastics), and rough conductors (metals, chrome, gold). Smooth conductors are skipped.

Cache architecture and defaults. The cache is a scene-level volume implemented as a position-conditioned neural texture. By default, we use a hashgrid encoder (base resolution 128, 8 levels, per-level scale 1.5, per-level hash map size 2^{22}, 4 features/level) paired with a linear decoder storing RGBA values in fp16. Alternatively, a neural decoder can be used, where the same hashgrid encoding feeds a small MLP (2 hidden layers of size 64).

Scene material configuration. We replace every reflectance parameter in non-smooth BSDFs with a trainable texture, regardless of whether the original value was spatially varying or constant. These textures are parameterized using a Laplacian pyramid [Weier et al. 2025] (base resolution 16, scale factor 2.0). While standard textures could also be used, we found that Laplacian pyramids improve convergence speed and yield cleaner recovered materials.

Memory footprint. Across 7 scenes, the cache parameters are constant at 240 MB. Scene-side trainable material parameters range from 44.2 MB up to 255.3 MB; the mean is 133.8 MB.

Because our method is implemented with PRB, optimization requires no more memory than a primal rendering pass.

Per-scene timing. Table 5 reports detailed per-iteration wall-clock time for each scene. Our method incurs a small overhead compared to PRB due to cache and blending field evaluation. Hadadan et al. [2023] is marginally faster as it updates materials using only primary hits and does not simulate light paths.

Scene	Ours	PRB	Hadadan	Hadadan*
Bedroom	0.041	0.035	0.032	0.032
Bathroom2	0.039	0.037	0.030	0.030
Kitchen	0.167	0.167	0.169	0.164
Living-room	0.036	0.035	0.037	0.036
Living-room-2	0.091	0.089	0.090	0.090
Living-room-3	0.034	0.031	0.019	0.019
Staircase	0.249	0.240	0.239	0.237
Mean	0.094	0.091	0.088	0.086

Table 5. **Per-iteration training time.** Average wall-clock time (seconds) per iteration for optimizing 2^{18} pixels (all overhead included). Our method adds a small overhead over PRB due to cache and α evaluation. Hadadan et al. [2023] is faster because it updates only primary-hit materials and does not simulate full light paths.

Additional progress results. Figure 12 shows additional optimization progress visualizations, complementing Figure 11 in the main paper.

Fig. 12. **Additional optimization progress.** Complementary visualizations to Figure 11, showing (top) material-only relit renders, (middle) cache at camera-ray intersections, and (bottom) $\alpha(\mathbf{x})$ over iterations. The bathroom scene's warm ground-truth illumination makes the neutral-gray initialization appear tinted when rendered under that lighting.