Eurographics Symposium on Rendering (2026)
I. Gkioulekas and A. Jarabo (Editors)

Differentiable Radiosity for Inverse Rendering

Gaspard Thévenon[1], David Coeurjolly[1], Julie Digne[1], Nicolas Bonneel[1]

[1]CNRS, Université Lyon 1, INSA Lyon, France

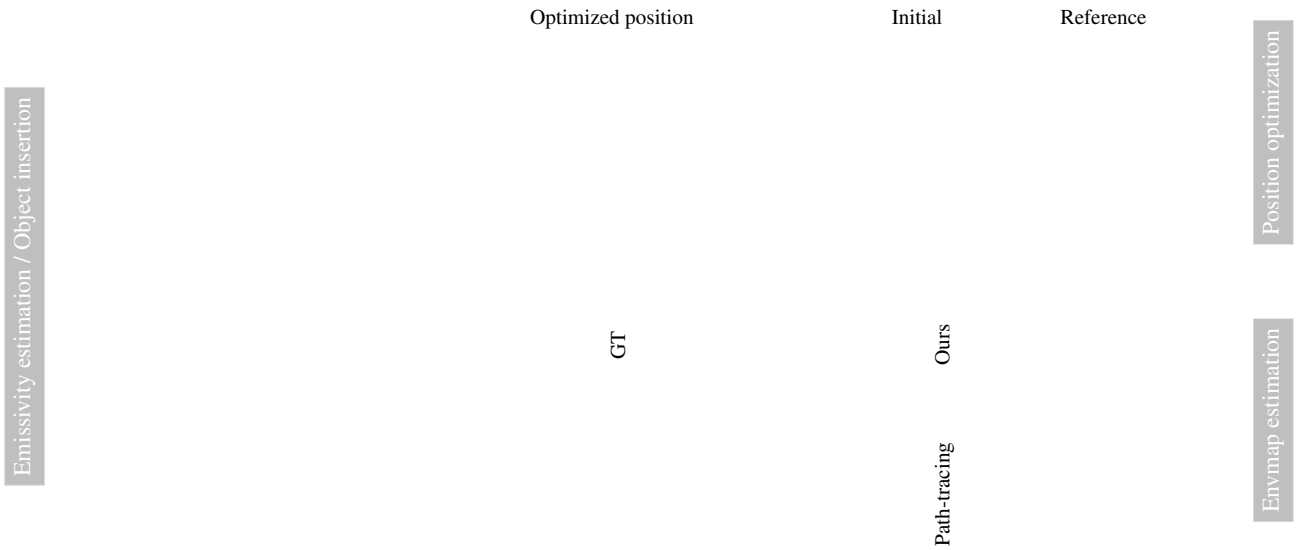

Figure 1: Using our differentiable radiosity framework, we are able to tackle several inverse rendering tasks: emissivity estimation from real-world photographs in order to perform synthetic object insertion (left), estimation of the position of a hidden object based on indirect information like color cast (top right), and environment map estimation from renderings of an object (bottom right).

Abstract

Inverse rendering recovers the physical properties of a scene, such as materials and lighting, from images. Recent progress has been driven by differentiable path-tracing methods. However, non-differentiable visibility and noisy estimates make the optimization difficult. Instead, we explore an alternative approach based on radiosity, a classical rendering technique that models global illumination as the solution of a linear system. Based on an antiradiance formulation that explicitly avoids visibility computations, we present a novel differentiable radiosity renderer, derive the analytical gradients required for optimization, and propose a general framework for inverse rendering. We also provide mathematical insights on the ill-posedness of our problem. We compare our method to a state-of-the-art differentiable path-tracer, and demonstrate that our approach achieves competitive or improved performance in terms of convergence and reconstruction quality.

CCS Concepts

• ***Computing methodologies*** → ***Rendering;*** *Machine learning;*

1. Introduction

Inverse rendering seeks to recover the physical properties of a scene, such as materials, lighting, or geometry, from observed images. While forward rendering synthesizes images from a known scene description, inverse rendering tackles the ill-posed problem of inferring scene parameters that best explain the visual data. The wide range of applications of inverse rendering includes scene reconstruction, relighting, and material digitization.

This task has gained particular interest with the advent of differentiable rendering techniques. Differentiable renderers are not only capable of computing images given a scene description, but are also able to produce gradients of a loss with respect to scene parameters. They can then backward propagate image-based losses, and, when integrated into a machine learning framework, recover scene parameters to fit target renderings. Recently, differentiable path-tracing has become a widely used standard in this domain, thanks to its ability to handle complex light transport effects such as indirect illumination and caustics.

We propose an alternative differentiable rendering approach based on radiosity, a class of techniques that casts global illumination as the solution of a linear system via finite element discretization. Specifically, we build on the antiradiance algorithm, which replaces visibility computation with iterative shadow propagation, making the method inherently differentiable and addressing two fundamental drawbacks of differentiable path-tracing: visibility discontinuities and stochastic noise from Monte Carlo estimators. While radiosity has been known and studied for decades, to the best of our knowledge it has never been integrated into a differentiable rendering framework. In this work, we derive the required gradient formulae and propose both a theoretical analysis and practical implementation insights. Contrary to many existing inverse rendering methods, we propose a general physically-based framework with few hypotheses on lighting conditions or scene parameters. Our results show better convergence and accuracy than differentiable path-tracing, and highlight further desirable properties of our approach: computation time is practically independent of the number of views, and the process is completely deterministic, alleviating poor convergence due to noisy estimates.

To summarize, we make the following contributions:

- We propose a general framework for differentiable radiosity rendering, along with an open-source implementation (code available in supplementary materials).
- We study the correctness of our method for inverse rendering, and give physical and mathematical insights on the space of solutions.
- We compare our method to a differentiable path-tracer in the context of computer graphics applications.

2. Previous work

2.1. Radiosity algorithms

Radiosity algorithms are a class of techniques to solve the rendering equation [Kaj86], which have been developed for several decades. They use triangles from a mesh as a support for finite elements, and model light exchanges between them in the form of a linear system [GTGB84]. This seminal work led to a rich literature. Some methods were proposed to break the quadratic complexity, for example, by using hierarchical structures to limit the number of light exchanges [SAG94, SDS95]. Another line of work improved the estimation of form factors between triangles [SP89], especially with the use of hemicubes [CG85] that leveraged z-buffer hardware for visibility tests. Other methods have been proposed to use both radiosity and path-tracing to enhance the realism of rendered images [WCG87]. For an extensive study of this domain, we refer the readers to existing surveys, such as that of Cohen and Wallace [CW93]. While radiosity methods have mostly been abandoned due to their stringent requirements on tessellation and the simplicity of Monte Carlo path tracing, we instead bring radiosity back to light for its amenability to noise-free differentiation. Our implementation revives the *antiradiance* framework introduced by Dachsbacher et al. [DSDD07]. They present an alternative formulation of the radiosity linear system, replacing computationally heavy (and non-differentiable) visibility tests with an additional light operator. This operator iteratively propagates shadows using a virtual quantity named antiradiance. They build upon negative light, which approximates ambient occlusion and inter-reflection [Bun05], and the rendering equation derived by Pellegrini [Pel99] where visibility is substituted by negative light transmitted through surfaces. Similarly to antiradiance, we are not restricted to diffuse scenes, but we support moderately glossy materials.

Radiosity has previously been used for inverse rendering problems [YDMH99, Con02, ZCC16]. These methods achieve impressive results, but are specialized in handling specific inverse rendering problems: Zhang et al. develop a specialized method for indoor scenes [ZCC16], Contensin focuses on user constraint formulations [Con02], and Yu et al. assume explicitly measured material [YDMH99]. However, in contrast to our method, these approaches do not use target differentiable rendering.

Existing approaches [FLZ*25] already use inverse FEM for differentiable simulation of radiative heat transfer, solving a forward and adjoint linear system on a mesh discretization, which is a theoretical framework closely related to ours. However, their method targets thermal simulation and hybridizes FEM with GPU-accelerated photon tracing to handle specular effects, whereas to our knowledge our method is the first to apply inverse FEM in a purely radiosity rendering context.

2.2. Differentiable rendering

Differentiable rendering refers to a category of rendering techniques that are capable of providing gradients of image-based losses with respect to scene parameters, notably for solving inverse rendering problems. This domain is prolific, and we also refer the interested reader to existing surveys [KBM*20].

Previous works have proposed efficient implementations of differentiable rasterization, like OpenDR [LB14], DIB-R [CGL*19], or the more recent nvdiffrast library [LHK*20]. Although slower, our method is able to account for global illumination, whereas these methods are not. Recently, differentiable path-tracing has become a standard for inverse rendering problems, especially in scenes containing challenging light effects (e.g. indirect illumination or caustics) [LADL18, ALKN19]. It is notably implemented in well-known libraries such as Mitsuba [NDVZJ19] and Dr.Jit [JSRV22]. Differentiable path-tracing usually involves sophisticated methods to handle discontinuities [LADL18, LHJ19, ZRJ23] or to reduce variance [NRN*23]. Although initially limited to simple optimization setups, various efforts have been made to reduce memory and time complexities. With Radiative Backpropagation (RB), Nimier-David et al. [NDSRJ20] proposed to compute gradients by solving an adjoint transport equation (instead of using a very large

computation graph for auto-differentation), dramatically reducing memory requirements. Vicini et al. [VSJ21] improved this solution through Path Replay Backpropagation (PRB), and differentiates the adjoint equation locally by replaying each sampled path and applying reverse-mode differentiation along that path, achieving the same result in linear time and constant memory. Our method shares a strong similarity with RB: we also compute gradients by solving an adjoint transport equation in order to keep a tractable memory complexity. Our method is however based on radiosity instead of path-tracing. Several authors have proposed techniques to improve inverse rendering in Mitsuba by post-processing gradients [CYB*24] or specializing gradient-descent updates [NJJ21] at the optimization step. Post-processing can in principle be applicable beyond Mitsuba, though its effectiveness may vary depending on the gradient structure of the underlying renderer. In our comparison with Mitsuba, we do not include gradient post-processing.

Neural Radiosity methods [HCZ21, HZ22, HLN*23] represent the spatio-angular radiance distribution as a neural network, and use the residual of the transport equation as a prior to train the network and achieve global illumination. These approaches share similarities with our work: we both propose differentiable constructions of radiosity distributions specifically for inverse rendering tasks. However, Neural Radiosity represents those distributions as neural networks, whereas we develop a differentiable extension of existing radiosity techniques. Moreover, Hadadan et al. build their solution upon Mitsuba, and thus inherit the drawbacks of differentiable path-tracing and Monte-Carlo estimators.

Jiang et al. [JSL*25] recently proposed a differentiable method based on radiosity dedicated to scenes represented by semi-transparent Gaussian surfels. In order to achieve this, they resort to Monte-Carlo estimation to solve the light transport, thus scaling to the many surfels required to model these scenes. In our approach, we base our analysis on a deterministic radiosity formulation, and propose mathematical insights for our method. We apply our framework to different inverse rendering tasks (like non-line-of-sight position optimization), and compare to differentiable path-tracing.

Finally, radiance fields have attracted significant attention in the past few years. Instead of reconstructing the scene as a mesh with physical attributes, the scene is represented as a spatio-angular field. This can be done implicitly, as in NeRF [MST*20], or explicitly, as in 3D Gaussian Splatting [KKLD23]. Such representations are trained using specialized differentiable renderers and typically rely on a volumetric formulation, where only a radiance field is optimized. Our method, on the other hand, is based on a surface-based representation, and models light bounces in order to correctly take indirect light effects into account.

2.3. Intrinsic decomposition and inverse rendering

Another line of work related to our applications is the problem of intrinsic decomposition. This problem attempts to separate material parameters (e.g., albedo, roughness, metallicity...) and lighting, which is known to be an ambiguous task. Most often, this is performed on 2D photographs [BKPB17], but the lack of 3D information makes occlusion and interreflections more difficult to model [CRA11]. Leveraging a better 3D representation allows the user to edit 3D scenes, e.g., render them with different materials or under different lighting conditions.

Seitz et al. [SMK05] and Bai et al. [BCNR10] formalize inverse rendering as a dual counterpart of forward light transport, showing how to cancel individual contributions of different light bounces (namely to distinguish direct lighting from interreflections). Compared to our method, these approaches also use a dual formulation of light propagation, but do not use differentiable rendering, and do not target geometry optimization.

Differentiable renderers can recover scene parameters via optimization, and are thus suited to the intrinsic decomposition problem. Since the problem is ill-posed, this usually requires some form of regularization or priors. In the domain of radiance fields, several methods add physical attributes (such as material parameters) to the representation, both for NeRFs [SDZ*20, JLX*24, YCB*23] and 3D Gaussian Splatting [GGL*23, SWW*24]. These attributes are optimized using the differentiable renderer and regularized losses to help circumvent the ambiguity of the problem. Our physically-based method instead uses an explicit mesh representation and handles indirect illumination.

3. Our method

We provide a differentiable radiosity solver that can be used to optimize different parameters of scenes represented by meshes. Our renderer produces view-dependent face color distributions, and a simple rasterizer can be used afterward to generate images. A high-level representation of our pipeline is given in Fig. 2.

For completeness, we first present the antiradiance algorithm introduced by Dachsbacher et al. [DSDD07] in the next section.

3.1. The Antiradiance framework

As in standard radiosity, we use the mesh to discretize the solving of light transport. To render view-dependent effects such as glossy materials, a radiance distribution is associated to each triangle, discretizing both hemispheres into bins (Fig. 2). For solving the rendering equation, we solve a system with $N \cdot B \cdot \Lambda$ elements, where N is the number of triangles, B the number of bins per triangle to represent radiance distributions, and Λ the number of spectral wavelengths. We use $\Lambda = 3$ in our implementation, but consider here the case $\Lambda = 1$ for the sake of clarity.

In our discrete settings, we denote the vector of radiance distribution C[†], the vector of emissivity distribution E and matrices with bold letters. Vectors are of size $N \cdot B \times 1$ while matrices are $N \cdot B \times N \cdot B$.

Definition of operators. In *antiradiance* [DSDD07], the rendering equation, with radiance I_C, emissivity I_E, BRDF f, form factor G, on the scene S

$$I_C(x, \omega_o) = I_E(x, \omega_o) + \int_{y \in S} f_x(\vec{xy}, \omega_o)\, I_C(y, \vec{yx})\, G(x, y)\, \mathrm{d}y\,,$$

† We denote $C = L - A$, the difference between unoccluded radiance and antiradiance, with the notations of Dachsbacher et al. [DSDD07].

Figure 2: General pipeline of our method. Black arrows represent forward flow, and red arrows represent gradient flow. Given a scene, the differentiable radiosity renderer produces radiance distributions for each triangle. These distributions are then sampled, once per requested viewpoint, to produce face colors (V colors per face, where V is the number of viewpoints), and final images are obtained by rasterization.

Figure 3: (left) Illustration of the three operators used in the antiradiance algorithm. The global transport operator **U** transports radiance (in yellow) without occlusion, the local reflection operator **K** models the light-material interactions, and the local go-trough operator **J** generates antiradiance (in red), effectively propagating negative light behind the surface. (right) At iteration 1, due to the lack of visibility computation, the bottom surface incorrectly receives radiance from the one on top, as it should be occluded by the middle surface. At iteration 2, the antiradiance generated by the middle surface compensates this unwanted radiance, and the bottom surface is correctly occluded.

is discretized into the following linear system:

$$C = E + (\mathbf{K} - \mathbf{J})\mathbf{U}C\,. \tag{1}$$

It involves three operators, illustrated in Fig. 3 (left). The first matrix **U** models energy exchange between any two triangles, *without accounting for occlusions*, as described next. **K** is the *local* reflection operator, modeling the interaction between light and material. **J** is the *local* go-through operator, responsible for the creation of a virtual *negative* quantity, called antiradiance. While standard radiosity formulations do not contain the **J** operator, the antiradiance approach replaces heavy visibility computations with an iterative shadow-propagation method, based on this additional operator **J**, as illustrated in Fig. 3 (right).

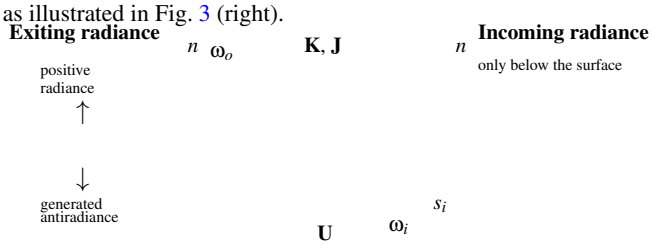

Figure 4: Notations used for the radiance distributions. Subscript i is used for "incoming", and o for "outgoing".

K can be expressed as a matrix indexed by $0 \leq i, j < N \cdot B$ stacking the triangle and bin indices. Such an index i (resp. j) can be decomposed into $i = Bi_t + i_b$ (resp. $j = Bj_t + j_b$), where i_t (resp. j_t) is the triangle index and i_b (resp. j_b) the bin index. Then $\mathbf{K}(i,j)$ is nonzero only if $i_t = j_t$, hence **K** is a block-diagonal matrix. More precisely:

$$\mathbf{K}(i,j) = \begin{cases} f_{i_t}(-\omega_{i_b}, \omega_{j_b}) s_{i_t} a_{i_t} \langle \omega_{j_b}, n_{i_t} \rangle \text{ if } i_t = j_t \\ 0 \text{ otherwise.} \end{cases} \tag{2}$$

where f_k, x_k, a_k and n_k are the BRDF, barycenter position, area, and normal to triangle k respectively. ω_ℓ is the direction of bin ℓ. A similar formulation holds for operator **J**, which shares the same block-diagonal structure:

$$\mathbf{J}(i,j) = \begin{cases} a_{i_t} \text{ if } i_t = j_t \\ 0 \text{ otherwise.} \end{cases} \tag{3}$$

Following the original approach [DSDD07], the local operators produce intensities instead of radiances for convenience, and C thus needs to be divided by the area and cosine factor to retrieve radiances.

The global transport operator **U** models light transport from triangle i_t to triangle j_t by transferring light between a single pair of bins. This pair of bins is determined by finding, for each hemisphere, which bin is crossed by the line joining both faces. Using

the same indexing as before:

$$\mathbf{U}(i,j) = \begin{cases} 0 \text{ if } i_t = j_t \\ \frac{1}{\|x_{i_t} - x_{j_t}\|^2} \max(\langle -\omega_{j_b}, n_{j_t} \rangle, 0) \text{ otherwise.} \end{cases} \quad (4)$$

The resulting matrix is zero on the diagonal and nonnegative everywhere else. Note that matrices are not stored explicitly as they are too dense and high-dimensional: their coefficients are instead evaluated on the fly when they are used.

However, this sampling method introduces heavy discontinuities. Instead of transferring light between a single pair of bins, we use bilinear interpolation, both for sampling and outputting radiance distributions. This yields the formula:

$$\mathbf{U}(i,j) = \frac{1}{\|x_{i_t} - x_{j_t}\|^2} \max\left(-x_{ji}^{(d)}.z, 0\right) \tilde{T}\left(x_{ji}^{(d)}\right) T\left(x_{ji}^{(s)}\right)^\top, \quad (5)$$

where $T(\omega)$ is the interpolation tensor in a local direction ω (for a fixed bin discretization), $\tilde{T}$ is T divided by the solid angles of the discretization, $x_{ji}^{(s)}$ is the normalized direction between triangles i_t and j_t expressed in a frame local to the source triangle, and similarly for $x_{ji}^{(d)}$ and the destination triangle.

Solving the linear system. Denoting $\mathbf{R} = (\mathbf{K} - \mathbf{J})\mathbf{U}$ and $\mathbf{\Gamma} = \mathbf{I} - \mathbf{R}$, we aim at solving:

$$C = \mathbf{\Gamma}^{-1} E, \quad (6)$$

which can be rewritten as:

$$C = \sum_{k \geq 0} \mathbf{R}^k E. \quad (7)$$

In standard radiosity, each term in the sum corresponds to a light bounce, but this is not the case in the antiradiance framework due to negative light propagation. In practice, the sum is computed through Jacobi iterations:

$$C = E + \mathbf{R}(E + \mathbf{R}(E + \dots)\dots), \quad (8)$$

and is truncated to the desired precision.

These computations result in a discrete radiance distribution for each triangle. Rendering images of the scene only amounts to sampling these distributions while rasterizing triangles. We write this with a sampling operator $\mathcal{S}$, that samples the distributions for every viewpoint, and depends on the camera and face positions:

$$\bar{C} = \mathcal{S}C \quad (9)$$

where, for a triangle i centered at x_i and a camera c centered at x_c, we have:

$$\mathcal{S}(i,c) = \max\left(\frac{1}{x_{ci}^{(s)}.z}, 0\right) \frac{1}{a_i} T\left(x_{ci}^{(s)}\right), \quad (10)$$

where, as before, we note $x_{ci}^{(s)}$ the normalized direction from triangle i to camera c in a frame local to i.

As radiance values are precomputed, rendering new views is relatively cheap (only one rasterization step per view), while standard differentiable path-tracing needs to perform a new Monte Carlo simulation for each view (see Fig. 14).

3.2. Differentiating antiradiance

To optimize a (column) vector parameter $\alpha = (\alpha_0, \dots, \alpha_p)^\top$, one typically provides a loss $\mathcal{L} := \mathcal{L}(\bar{C})$ (*e.g.*, the L_2 norm between an input and a rendered image), and performs a gradient descent or other optimization technique, requiring $\nabla_\alpha \mathcal{L}$. A classical approach would obtain gradients by automatic differentiation. However, standard reverse-mode differentiation may require intractable memory usage. Instead, we provide closed-form expressions that do not require any matrix or tensor storage.

This tractability stems from a key structural property of the antiradiance formulation: in standard radiosity, the visibility term couples each pair of elements to the rest of the scene geometry, introducing discontinuities that make derivative computations more difficult. In antiradiance, by contrast, the transport between any pair of triangles is computed independently of all other triangles: visibility is handled implicitly through iterated shadow propagation, so each operator can be differentiated directly and in closed form.

Applying the chain rule, $\nabla_\alpha \mathcal{L}$ is computed as:

$$\nabla_\alpha \mathcal{L} = \left[\frac{\partial \bar{C}_i}{\partial \alpha_j}\right]^\top \nabla_{\bar{C}} \mathcal{L}, \quad (11)$$

with $\left[\frac{\partial \bar{C}_i}{\partial \alpha_j}\right]$ the Jacobian matrix of $\bar{C}$, that we will simply denote $\frac{\partial \bar{C}}{\partial \alpha}$. We assume $\nabla_{\bar{C}} \mathcal{L}$ is obtained using a differentiable rasterizer, and focus on the remaining terms. Using Equation 9, we already have $\nabla_C \mathcal{L} = \mathcal{S}^\top \nabla_{\bar{C}} \mathcal{L}$. We validate our gradients using finite differences, as illustrated in supplementary materials, Appendix E.

Emissivity. Since C is linear with the emissivity E (see Eq. 6), Eq. 11 applied to $\alpha = E$ amounts to:

$$\nabla_E \mathcal{L} = (\mathbf{\Gamma}^\top)^{-1} \nabla_C \mathcal{L}. \quad (12)$$

We thus need to solve a *transposed* light transport problem. This system can be solved by Jacobi iterations, analogously to Eq. 7:

$$\nabla_E \mathcal{L} = \sum_{k \geq 0} (\mathbf{R}^\top)^k \nabla_C \mathcal{L} \quad (13)$$

involving transposed operators:

$$\mathbf{R}^\top = \mathbf{U}^\top (\mathbf{K}^\top - \mathbf{J}^\top) \quad (14)$$

The required memory for gradient computation does not depend on the number of light bounces (Fig. 14) as operators are evaluated on the fly. Note that this feature is inherent to our method, contrary to differentiable path-tracing (which needs special methods like PRB [VSJ21] to circumvent this issue).

Material parameters. We also compute gradients with respect to the material parameters, denoted θ hereafter, and thus consider $\alpha = \theta$. Only the reflection operator $\mathbf{K}$ depends on θ, differentiating Eq. 1 and using the general formula differentiating matrix inverses $\frac{\partial \mathbf{\Gamma}^{-1}}{\partial \theta} = -\mathbf{\Gamma}^{-1} \frac{\partial \mathbf{\Gamma}}{\partial \theta} \mathbf{\Gamma}^{-1}$ leads to:

$$\frac{\partial C}{\partial \theta} = \mathbf{\Gamma}^{-1} \frac{\partial \mathbf{K}}{\partial \theta} \mathbf{U} \mathbf{\Gamma}^{-1} E = \mathbf{\Gamma}^{-1} \frac{\partial \mathbf{K}}{\partial \theta} \mathbf{U} C. \quad (15)$$

We then compute the gradient using the chain rule and simplify to obtain:

$$\nabla_\theta \mathcal{L} = (\mathbf{U}C)^\top \frac{\partial \mathbf{K}}{\partial \theta}^\top \nabla_E \mathcal{L}. \tag{16}$$

To compute the gradients with respect to the material parameters, we thus reuse C from the forward rendering pass, apply $\mathbf{U}$ to it, compute $\nabla_E \mathcal{L}$ as described in Eq. 12, and then compute the inner product of both using $\frac{\partial \mathbf{K}}{\partial \theta}^\top$. We provide $\frac{\partial \mathbf{K}}{\partial \theta}$ for parameters of the Blinn-Phong BRDF in the supplementary materials (Appendix A).

Positions. Contrary to previous computations, we have to take the sampling operator $\mathcal{S}$ into account when differentiating with respect to positions. This leads to:

$$\nabla_P \mathcal{L} = C^\top \frac{\partial \mathbf{U}}{\partial P}^\top \left(\mathbf{K}^\top - \mathbf{J}^\top\right) \nabla_E \mathcal{L} + C^\top \frac{\partial \mathcal{S}}{\partial P} \nabla_{\bar{C}} \mathcal{L} \tag{17}$$

The computations of $\frac{\partial \mathbf{U}}{\partial P}$ and $\frac{\partial \mathcal{S}}{\partial P}$ are more complex and can be found in the supplementary materials (Appendix B).

Adjoint methods. With Equations 11, 12, 16, and 17, we can see that our approach computes gradients using an adjoint method. Indeed, similarly to Radiative Backpropagation [NDSRJ20], we compute gradients by solving adjoint equations, which involve the partial derivative of the radiances with respect to a trained parameter. We can thus make a similar interpretation as Nimier-David et al., with a strong duality between the forward and backward passes: the loss gradient is seen as a quantity which is emitted by faces in the scene, scattered with successive bounces, and received by faces with trained parameters. Forward rendering applies operators such as $\mathbf{R} = (\mathbf{K} - \mathbf{J})\mathbf{U}$ to propagate radiance: it maps scene parameters to the radiance distribution resulting from one light bounce. On the other hand, the transposed operator $\mathbf{R}^\top = \mathbf{U}^\top(\mathbf{K}^\top - \mathbf{J}^\top)$ transports sensitivities from receivers back to emitters, indicating how a small change in trained parameters affects the loss. Another consequence of the adjoint method for the backward pass is that gradient accuracy depends on the number of backward bounces: in order for the gradients to be accurate, both the forward and backward passes need to simulate enough bounces to converge.

3.3. The light bounce ambiguity

In the experiments described in Sec. 4.2, we optimize an emissivity distribution stored on triangles. In some cases, some parts of the scene might not be visible in any image. In that case, we observed that the optimization tends to converge to a distribution where the emissivities of the triangles are equal to their radiances after the first light bounce in the reference scene. In other words, there is an ambiguity between the emissivity and the first bounce when light sources are not visible. This effect was also observed with differentiable path-tracing: Nimier-David et al. [NDDJK21] presented experiments where scattered light is baked into trained emissivities.

We formalize this observation in the supplementary materials (Appendix C), and show that we can describe the set of solutions as a convex polytope $\mathcal{P}$. Furthermore, we prove in Appendix D that $\mathcal{P}$ is bounded, so it is equal to the convex combinations of its vertices.

In addition to this ambiguity, we explain why the optimization tends to prefer convergence to such *first-bounce emissivity* solutions. Indeed, we recall the expression of $\nabla_E \mathcal{L}$:

$$\nabla_E \mathcal{L} = \nabla_C \mathcal{L} + \mathbf{R}^T \nabla_C \mathcal{L} + \left(\mathbf{R}^T\right)^2 \nabla_C \mathcal{L} + \dots \tag{18}$$

$\nabla_C \mathcal{L}$ is the dominant term in this formula: as in forward rendering, light bounces generally attenuate the magnitudes of radiances. In other words, $\nabla_E \mathcal{L}$ will be mostly determined by the first term $\nabla_C \mathcal{L}$. But by definition, $\nabla_C \mathcal{L}$ is given by the rasterizer, so it only contains information about visible triangles. So, with this pipeline, emissivity gradients will be nonzero primarily on visible triangles. Therefore, these visible colors will naturally absorb color errors as emissivity changes.

We propose a solution which effectively mitigates this effect by modifying our rendering pipeline. If we know that light sources are not visible, instead of computing the colors using $C = \mathbf{\Gamma}^{-1} E$, we add a light bounce: $C = \mathbf{R}\mathbf{\Gamma}^{-1} E$. Upon convergence, applying R again does not change the target radiance distribution on visible triangles, but this has the effect of masking out the direct emission term. Similarly to Eq. 8, we consider:

$$C = \mathbf{R}E + \mathbf{R}^2 E + \dots \tag{19}$$

The corresponding backward pass can be written as:

$$\nabla_E \mathcal{L} = \sum_{k \geq 1} (\mathbf{R}^\top)^k \nabla_C \mathcal{L}\,. \tag{20}$$

Because of the effect of removing unwanted emission terms, we call this feature *masking*. In our experiments (see Fig. 13), we observe that when enabled, this allows the optimization to modify significantly triangles that are not directly visible, bringing the convergence closer to the ground truth emission.

3.4. Other regularizations

To account for the ill-posedness of the lighting estimation problem, our image-based losses can be augmented with other regularization terms.

Sparsity. Emissivity distributions are usually sparse, because there is only a small fraction of triangles that act as light sources. This motivates the use of an average L1 loss to enforce sparsity, activated with an atan function:

$$\mathcal{L}_1(E) = \frac{1}{N} \sum_{i=1}^{N} \operatorname{atan}(||E_i||_1) \tag{21}$$

Mono-chromaticity. If we assume that light sources are close to white, we can use a light chromaticity prior (where norms are padded with a small value to avoid numerical issues). Denoting E_i the emissivity of triangle i:

$$\mathcal{L}_m(E) = \frac{1}{N} \sum_{i=1}^{N} \left\| E_i - (1,1,1)^\top \frac{\|E_i\|_2}{\sqrt{3}} \right\|_2. \tag{22}$$

Total variation. We also want to constrain the emissivity distribution on the mesh to only have a few nonzero connected parts, and to be constant in them. This can be enforced using a Total Variation

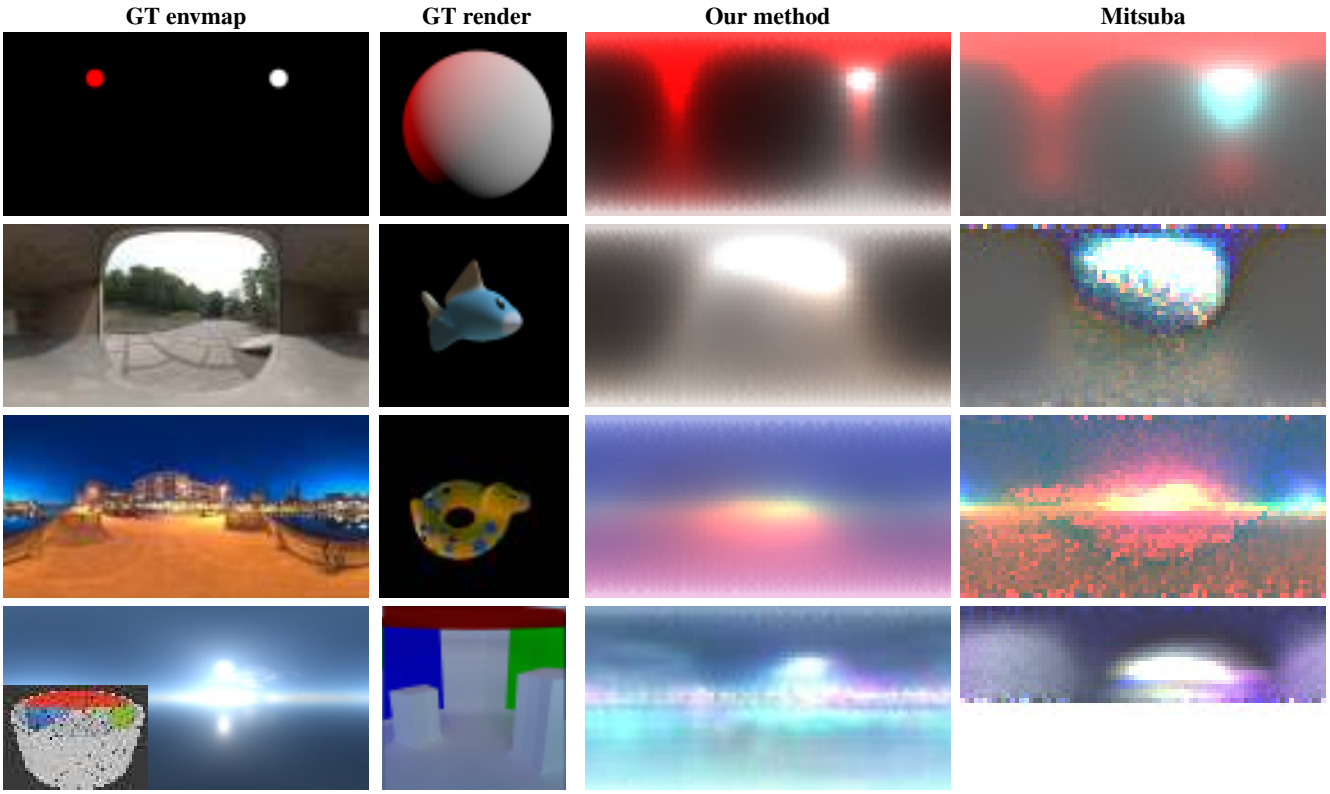

Figure 5: Environment map training. The first column shows the ground truth environment map used to illuminate the scene. The second column shows one of the reference images, rendered with our method. The third column shows the resulting environment map trained with our method. The forth column shows the resulting maps with Mitsuba. All optimized environment maps are shown at convergence. In the last row, a blender view of the scene is given on the left to help understand its composition. Note that the bottom half of the last Mitsuba environment map remains untouched, while ours presents non-physical emissivity due to antiradiance slowly altering the environment map, in the absence of information in this region.

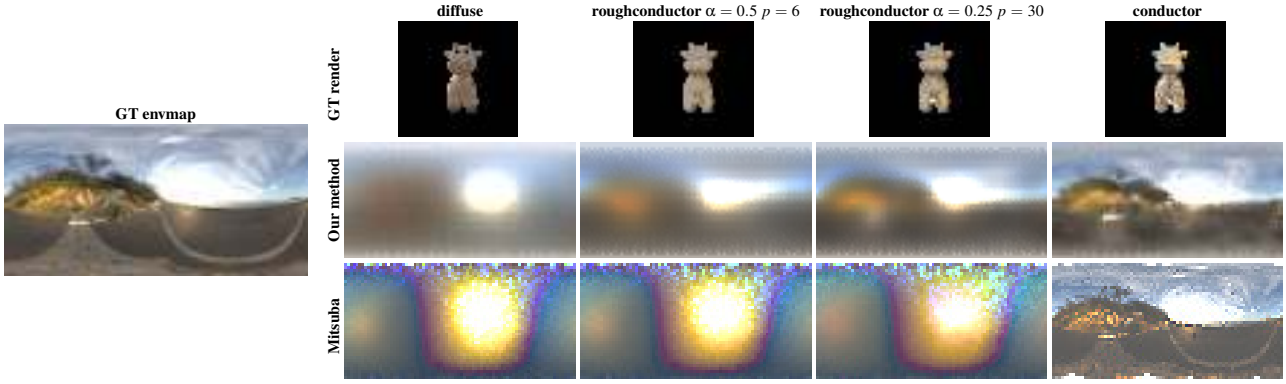

Figure 6: Environment map estimation at increasing surface specularity. We use 8 input images (one is shown here). We show the results for four material models (one per column). The first one is a simple Lambertian model. The next two are rough conductors with a Beckmann distribution controlled by the α roughness parameter for Mitsuba, or an equivalent Blinn-Phong model for our method (using a standard correspondence between α and the specular power p). The last one is a perfect mirror surface, modeled by a conductor material for Mitsuba. The environment map has a constant resolution of 3100 triangles (for our method) or 40x80 pixels (for Mitsuba). This highlights how surface glossiness helps in recovering higher frequencies in the environment map texture.

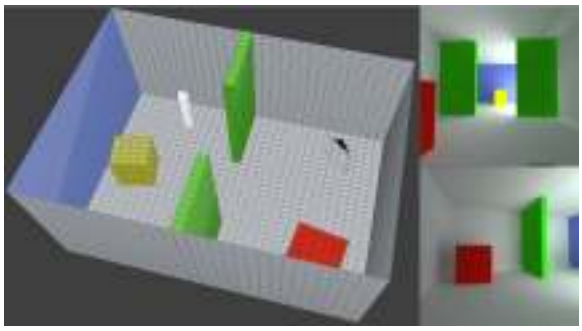

Figure 7: Synthetic scene for convergence study in Figures 11 and 13. Blender view of the scene without the roof (left) and two of the training images of the scene (right).

Scene	PSNR↑ Ours	PSNR↑ Mitsuba	SSIM↑ Ours	SSIM↑ Mitsuba
Sphere	**7.786**	5.296	**0.016**	0.005
Blub	12.119	**13.524**	0.557	**0.577**
Bob	**10.492**	10.434	0.461	**0.479**
Cornell	**9.592**	6.734	**0.780**	0.761
Spot (diffuse)	**14.419**	12.958	**0.615**	0.602
Spot ($\alpha = 0.5$)	**17.059**	13.382	**0.625**	0.607
Spot ($\alpha = 0.25$)	**18.254**	14.396	**0.630**	0.614
Spot (conductor)	**20.114**	20.053	0.630	**0.659**

Table 1: Comparison table between the trained environment maps (with our method and with Mitsuba) and the ground truth environment maps.

(TV) loss:

$$\mathcal{L}_{TV}(E) = \frac{1}{A} \sum_{\{F_1,F_2\}} \frac{a_{F_1} + a_{F_2}}{2} \|E_{F_1} - E_{F_2}\|_1 \,, \tag{23}$$

where the sum is indexed on the set of adjacent faces, a_F is the area of face F, and A is the sum of adjacent face areas.

4. Applications

We evaluate our inverse rendering method on several applications. Sections 4.1 and 4.2 present experiments using our method to optimize emissivity distributions. We show that our method can optimize environment maps or an emissivity distribution stored directly on the triangles of the mesh. Finally, Sec. 4.3 presents joint material parameters and emissivity optimization, with the downstream task of scene editing and relighting. In the following applications, to reduce the number of optimized variables, a single emissive color is optimized per emissive triangle, which is spread over the discretized hemisphere of directions.

4.1. Environment map estimation

We aim at retrieving an environment map from renderings of a scene, where the environment map is not directly visible. In order to integrate it into our framework, we represent the environment map as a tessellated sphere (3100 triangles in our experiments) of large radius encompassing the scene. We directly optimize the emissivities of these triangles in logarithmic space. Unless specified otherwise, the synthetic input images are generated with our renderer. To generate these images, each triangle of the tessellated sphere is given the radiance of a sampled HDR ground truth environment map [Hav]. If we denote $I = \{I_k\}$ the currently optimized images, $|I_k|$ their sizes, K the number of images, and $I^{in} = \{I_k^{in}\}$ the input images, we consider an average Euclidean distance as the loss function without additional regularization:

$$\mathcal{L}(I) := \frac{1}{K} \sum_{k=1}^{K} \frac{1}{|I_k|} \|I_k - I_k^{in}\|_2 \,. \tag{24}$$

We use an Adam optimizer with a learning rate of 0.08 in all our experiments.

We compare our method to an equivalent optimization pipeline in Mitsuba 3 [JSRV22]. For fair comparison, images used as input for Mitsuba were also generated in Mitsuba, and the number of pixels in the trained environment map is the same as the number of its triangles with our method. For both our method and Mitsuba, the optimized environment map is initialized to a uniform gray radiance. We also use an Adam optimizer, with a learning rate of 0.0007 in all our experiments.

Results. Results are shown in Fig. 5 for diffuse objects and Fig. 6 for glossy objects. A result for a real scene is given in Figures 9 and 10, and shows that we can integrate virtual 3D objects (a bunny and a monkey) in a photograph with consistent lighting. Quantitative comparisons are given in Table 1, with PSNR and SSIM scores computed between the ground truth environment maps and the optimized ones. Figure 5 shows that with diffuse materials, we can only aim at reconstructing a low-frequency approximation. Still, our method outperforms inverse path-tracing consistently. Fig. 6 illustrates the influence of material glossiness, from completely diffuse to perfect mirror, on the quality of the environment map reconstruction. As expected, the reconstruction quality increases with material specularity for both methods. For a perfect mirror, however, we can see that Mitsuba achieves a sharper result, albeit noisier. Both methods are limited by the resolution of the reconstructed environment map, but our method is also limited by the resolution of the mesh and by the number of hemispherical bins in our implementation. However, this experiment shows that this only becomes limiting for highly specular materials.

4.2. Emissivity optimization

When light sources are part of the geometry of the scene, we optimize the emissivity distribution over the triangles of the mesh directly instead of an environment map. It allows us to handle scene were the light does not come from the environment, but from objects inside the scene. We apply our framework to retrieve emissivity distributions on a synthetic scene, where we compare our performance to path-tracing, and on a real-world scene, as shown on Fig. 10. We use a learning rate of 0.02 in our method, and 0.0003 with Mitsuba.

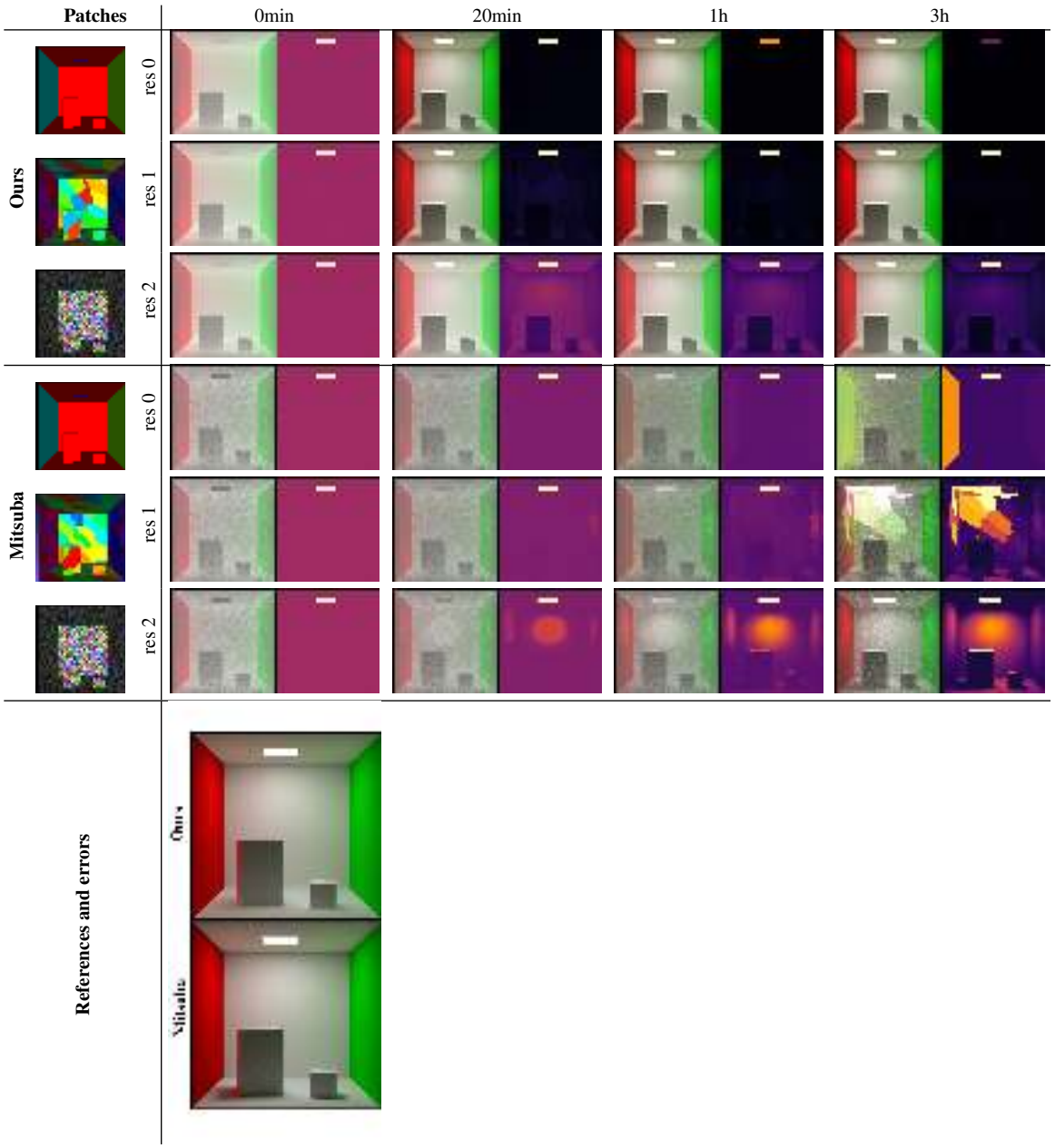

Figure 8: Comparison of emissivity distributions convergence between our method and Mitsuba, with different mesh resolutions. For each resolution, each cluster of triangles shares the same set of variables. The coarsest resolution, *res 0*, contains 4 clusters. The second one, *res 1* contains 80 clusters, and *res 2* is the finest resolution and each triangle is optimized independently (around 7.5k triangles). The first column visualizes clusters with false colors. The next columns show the results of the optimization at the beginning, after 20min of optimization, 1h and 3h. We show the rendered image from one of the training viewpoint, and a visualization of the error from the same viewpoint, with a logarithmic scale. One of the eight input images is shown in the last row, along with the errors of the trained distributions with respect to the ground truth distribution. Images in the first column show the patch segmentation in false colors for each resolution.

Figure 9: Trained environment map on a real scene. The geometry was reconstructed using Colmap [SF16] using around 50 photographs (24 are shown here). Two lights are used to illuminated the scene: a yellow-white, and a magenta light, which appear on the resulting environment map.

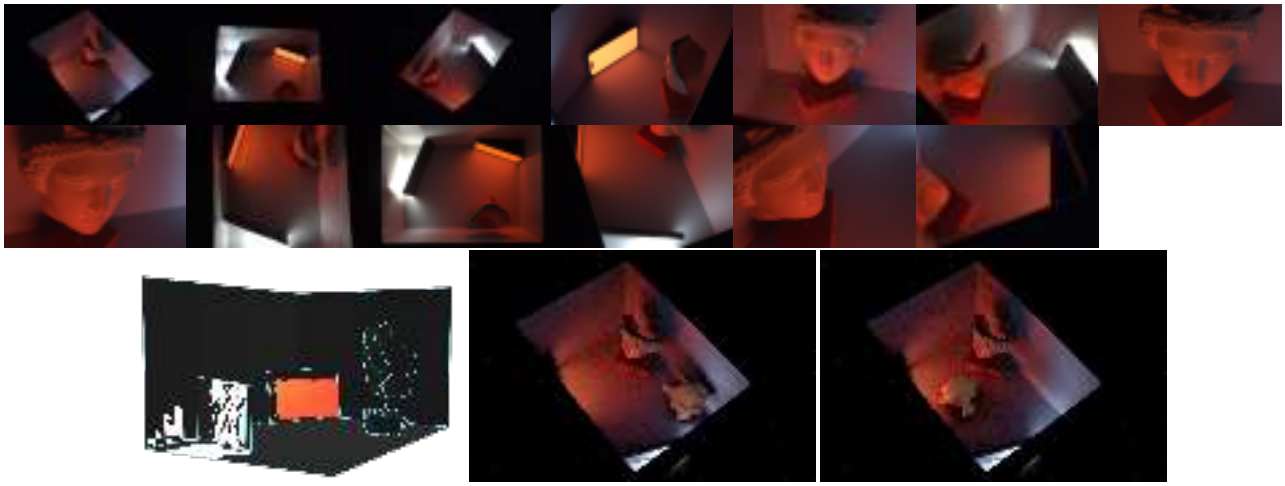

Figure 10: Trained emissivity distribution on a real scene. The geometry was reconstructed using Colmap [SF16] using 13 photographs. Two phones are used to light the scene (one white, and one red), and are included in the geometry of the scene. The emissivity distribution is trained on the whole mesh. Note that geometry reconstruction is inexact, and can lead to small errors in the optimized emissivity distribution.

Impact of the mesh resolution. In Fig. 8, we study the speed of convergence and the accuracy with respect to the number of variables. We decompose a Cornell Box into groups of triangles known to have the same material by clustering them by geodesic distance. We then optimize per-cluster emissivity. In this set of experiments, we only use the image-based loss. We use a learning rate of 0.02 with our method, and 0.0003 with Mitsuba. As expected, convergence is faster with a smaller number of clusters, because there are fewer variables to optimize. We can see that the Mitsuba-based optimization converges significantly slower than our optimization, independently of the number of variables. On the images, it appears that this is due to the stochastic nature of Mitsuba's optimization, with noisy estimates slowing down convergence. On the other hand, increasing the learning rate degrades the quality of the solution. This experiment highlights the stability of our method compared to differentiable path-tracing.

Impact of the masking feature and regularizations. We assess our masking feature (Sec. 3.3) to account for invisible light sources, on a synthetic scene represented in Fig. 7, in which the light source is hidden from the camera.

From input images, we try to recover the emissivity distribution, even though the light source is not visible, and the materials are diffuse. We run the optimization with and without the modification presented in section 3.3, as well as with and without the regulariza-

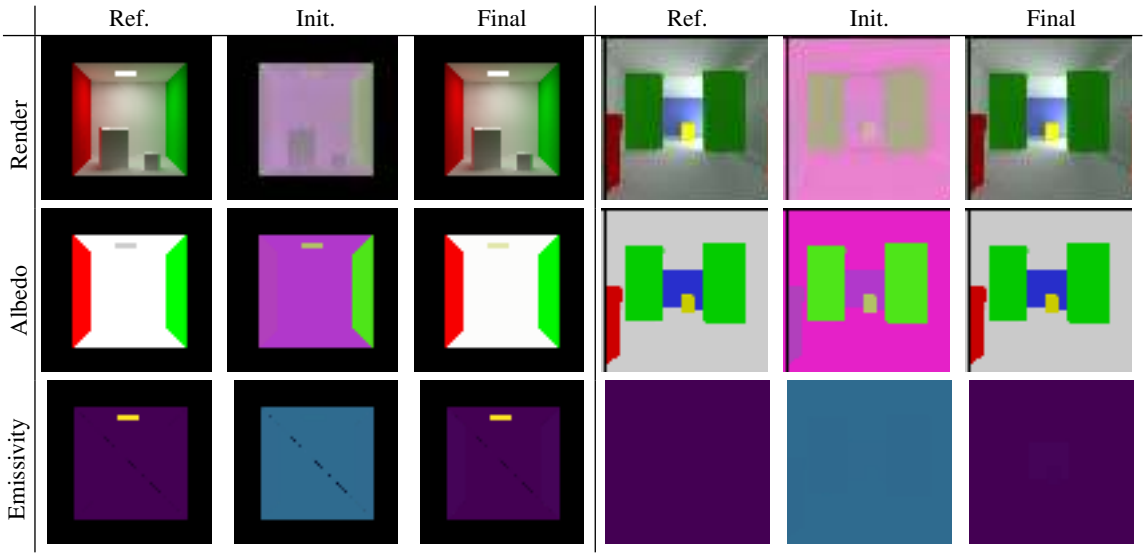

Figure 11: Our method is able to perform joint estimation of albedo and emissivity. For the right scene, we used regularization $(\lambda, \lambda_m) = (1, 0.1)$ to disambiguate the solution.

tion losses of Sec. 3.4. The total loss with regularizations is:

$$\mathcal{L}_{\text{reg}}(I,E) := \lambda\mathcal{L}(I) + \lambda_1\mathcal{L}_1(E) + \lambda_m\mathcal{L}_m(E) + \lambda_{TV}\mathcal{L}_{TV}(E) \quad (25)$$

with $(\lambda, \lambda_1, \lambda_m, \lambda_{TV}) = (10.0, 0.5, 0.001, 0.03)$.

Results are shown in Fig. 13. Since this problem is ambiguous and the space of solutions is high-dimensional (Sec. 3.3), we do not aim at converging exactly to the ground truth solution. Instead, we show that without masking, the optimization is stuck in a subset of incorrect solutions, but enabling it allows one to reach solutions that are much closer to the ground truth. Without masking, we can see that only visible triangles become emissive during the optimization, as described in Sec. 3.3: the optimization converges to a distribution close to the first light bounce in the scene. However, with masking enabled, hidden triangles are able to become emissive. The additional regularization terms further improve the solution.

4.3. Joint optimization of material and emissivity

Then, we use our method to address a more difficult problem: the joint estimation of emissivity and material parameters (here, albedo) which is known to be a highly ambiguous problem. In this section, we also group triangles into clusters known to have the same parameters, and optimize only one emissivity and one albedo color per cluster. We only use an image-based loss, without regularization. Results for two scenes are shown in Fig. 11. On the second scene, we also used our masking feature, in order to show its behavior when jointly optimizing materials and lighting. We can see that our method converges to the correct distribution, even with important indirect lighting effects.

4.4. Position gradients

Finally, we also show that our method is capable of providing gradients with respect to positions, as detailed in Section 3.2. As geometry optimization is a difficult task which requires special care and optimization techniques in inverse rendering [NJJ21], we restrict ourselves to the simpler task of optimizing the position of an object in a scene, in order to highlight the contribution of our differentiable pipeline without any additional optimization component. In these experiments, we optimize the position of an object by comparing one image of the scene to a ground truth image, in order to recover the position of this object used in the reference image. However, in order to showcase the stability of our method, the object is not directly visible from the camera: the gradients only come from indirect information, like colors cast on the walls.

5. Implementation and performance

Implementation. We implemented our method using CUDA 12.6 kernels to compute forward and backward passes, with a PyTorch [PGM*19] binding wrapping them as a custom operator. At the end of the optimization pipeline, we use PyTorch3D [RRN*20] as our differentiable rasterizer (except in Section 4.4, where nvdiffrast [LHK*20] is used). The complete rendering pipeline is compatible with PyTorch, and we use the Adam [KB14] optimizer. In order to train emissivity and albedo distributions with Mitsuba, we used its face attribute feature. All of our time and memory experiments were run on a personal computer with a NVIDIA Tesla V100. The number of bins per triangle controls the directional resolution of our light simulation. In all of our experiments, we used 130 bins per hemisphere, linearly dividing the angular domain. In our experiments with Mitsuba, we also use the Adam [KB14] optimizer, and the Path Replay Backpropagation integrator [VSJ21]. We use 5 samples per pixel during the optimization.

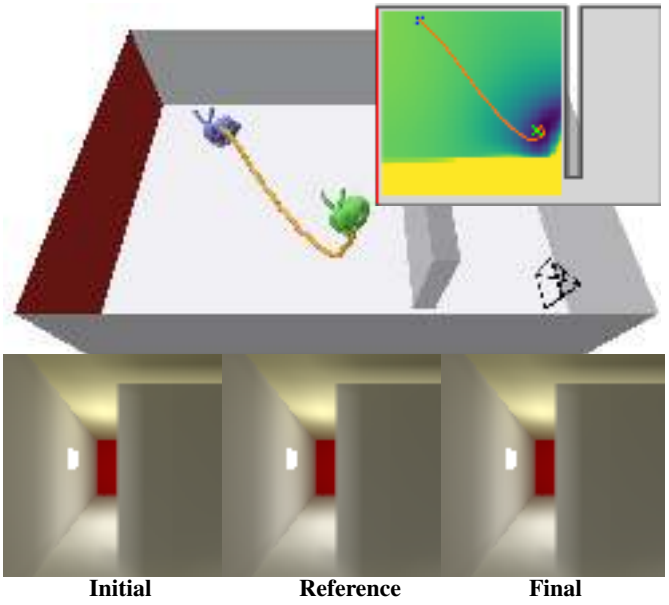

Figure 12: Optimizing the position of a bunny shape in the scene by applying our method to a single view. The optimization of the position can be seen on the top image, with the initial position in blue, final position in green, and trained position with the yellow path. The position of the camera is also shown. In the top-right corner, the landscape of the image loss is shown (at z=0), alongside with the optimized translation. In the bottom row, the initial, final, and reference renderings used to train the position are shown (500 iterations, around one hour on our machine).

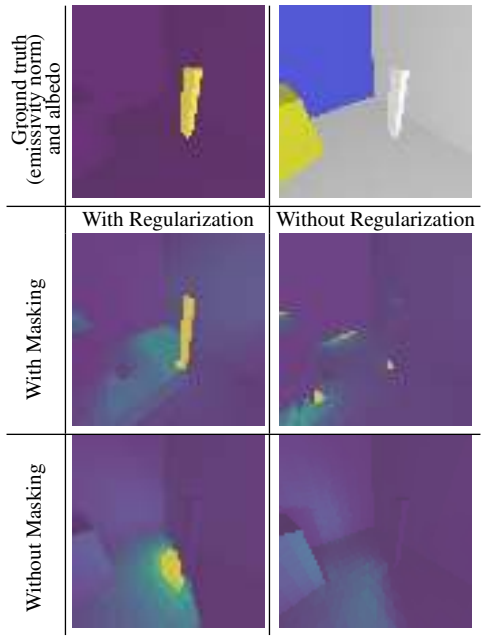

Figure 13: Illustration of our masking strategy to retrieve invisible light sources. As shown on Fig. 7, in this scene, the light source is the white pole that we can see here, but which is invisible on the input images. Except for the image of the albedo (on top right), the images in this figure show emissivity norms. The first row shows the ground truth emissivity and albedo. The second row shows the optimized emissivity distribution with our masking feature (and with or without regularization losses), and the third gives the optimized emissivity distribution without our masking feature. On these results, we can see that without masking, the optimization is unable to assign high emissivities to invisible triangles. This problem is mitigated with our masking feature, and regularization can guide the optimization towards a reasonable solution.

Execution time. We show that our method has little time dependence in the number of input images for a given scene, unlike Mitsuba. As expected, we can see in Fig. 14-(left) that Mitsuba exhibits a linear time dependence in the number of images to render, because it has to simulate light transport independently for every render. On the other hand, the execution time of our method only marginally depends on the number of images to render because the light simulation is done only once, and only a fast rasterization step has to be performed for every image.

In the supplementary materials, we study the memory usage and speed of our method, in Appendix F. It illustrates numerically the expected behavior of our method: at a fixed directional resolution (130 bins per hemisphere in our case), the VRAM usage of our method is linear in the mesh size (around 1.8GB for every 50k triangles). Time complexity, however, remains quadratic. This could be mitigated with acceleration techniques like hierarchical radiosity [SAG94, SDS95] but we leave this implementation for future work.

Figure 14: Comparison of the total execution time (in seconds) for our method (blue) and Mitsuba (orange) with respect to the number of images to render. Times were measured on the spot scene with three light bounces, Mitsuba uses the PRB integrator, and the number of spp for Mitsuba (128) is chosen so that both methods have approximately the same execution time for the first point of the plot.

6. Discussion and conclusion

In this article, we present a novel differentiable rendering approach based on radiosity algorithms. This method, complementary to inverse path-tracing, addresses the fundamental drawback of noisy

Monte Carlo estimates. We show that our method generally outperforms differentiable path-tracing on several inverse rendering tasks. Our approach inherits some limitations specific to radiosity algorithms: the computation time strongly depends on the discretization of the scene. Reaching higher frequencies requires finer resolution, both for the triangle mesh and for the discretization of the hemisphere, which impacts hard shadows and highly specular materials. Additionally, radiosity approaches are less suited for advanced illumination phenomena, like caustics or subsurface scattering.

In the future, our implementation would benefit from efficient radiosity techniques (like clustering or hierarchical models, higher order finite elements or interpolation schemes), and from more sophisticated material models which remain out of the scope of this paper. In addition, it would be interesting to apply our antiradiance model to shape optimization.

7. Acknowledgments

This project was provided with computer and storage resources by GENCI at IDRIS thanks to the grant 2026-AD010616148R1 on the supercomputer Jean Zay's V100 partition. It was also provided with HPC resources from the CC-IN2P3. This work is partially supported by the French National Research Agency within the PEPR ICCARE program (ANR-23-PEIC-0001).

References

[ALKN19] AZINOVIĆ D., LI T.-M., KAPLANYAN A., NIESSNER M.: Inverse path tracing for joint material and lighting estimation, 2019. Version Number: 1. doi:10.48550/ARXIV.1903.07145. 2

[BCNR10] BAI J., CHANDRAKER M., NG T.-T., RAMAMOORTHI R.: A dual theory of inverse and forward light transport. ECCV'10, Springer-Verlag, p. 294–307. 3

[BKPB17] BONNEEL N., KOVACS B., PARIS S., BALA K.: Intrinsic decompositions for image editing. *Computer Graphics Forum 36*, 2 (May 2017), 593–609. doi:10.1111/cgf.13149. 3

[Bun05] BUNNELL M.: Dynamic ambient occlusion and indirect lighting. URL: https://api.semanticscholar.org/CorpusID:61207984. 2

[CG85] COHEN M. F., GREENBERG D. P.: The hemi-cube: a radiosity solution for complex environments. *SIGGRAPH Comput. Graph. 19*, 3 (July 1985), 31–40. Place: New York, NY, USA Publisher: Association for Computing Machinery. doi:10.1145/325165.325171. 2

[CGL*19] CHEN W., GAO J., LING H., SMITH E. J., LEHTINEN J., JACOBSON A., FIDLER S.: Learning to predict 3D objects with an interpolation-based differentiable renderer, 2019. Version Number: 2. doi:10.48550/ARXIV.1908.01210. 2

[Con02] CONTENSIN M.: Inverse lighting problem in radiosity. *Inverse Problems in Engineering 10*, 2 (Jan. 2002), 131–152. doi:10.1080/10682760290004311. 2

[CRA11] CARROLL R., RAMAMOORTHI R., AGRAWALA M.: Illumination decomposition for material recoloring with consistent interreflections. In *ACM SIGGRAPH 2011 papers*. 2011, pp. 1–10. 3

[CW93] COHEN M. F., WALLACE J. R.: *Radiosity and realistic image synthesis*. Morgan Kaufmann, 1993. 2

[CYB*24] CHANG W., YANG X., BELHE Y., RAMAMOORTHI R., LI T.-M.: Spatiotemporal bilateral gradient filtering for inverse rendering. In *SIGGRAPH Asia 2024 Conference Papers* (New York, NY, USA, 2024), SA '24, Association for Computing Machinery. URL: https://doi.org/10.1145/3680528.3687606, doi:10.1145/3680528.3687606. 3

[DSDD07] DACHSBACHER C., STAMMINGER M., DRETTAKIS G., DURAND F.: Implicit visibility and antiradiance for interactive global illumination. *ACM Trans. Graph. 26*, 3 (July 2007), 61–es. Place: New York, NY, USA Publisher: Association for Computing Machinery. doi:10.1145/1276377.1276453. 2, 3, 4

[FLZ*25] FREUDE C., LIPP L., ZEZULKA M., RIST F., WIMMER M., HAHN D.: Inverse simulation of radiative thermal transport. *Computer Graphics Forum 44*, 2 (2025), e70048. doi:https://doi.org/10.1111/cgf.70048. 2

[GGL*23] GAO J., GU C., LIN Y., ZHU H., CAO X., ZHANG L., YAO Y.: Relightable 3d gaussian: Real-time point cloud relighting with brdf decomposition and ray tracing. *arXiv:2311.16043* (2023). 3

[GTGB84] GORAL C. M., TORRANCE K. E., GREENBERG D. P., BATTAILE B.: Modeling the interaction of light between diffuse surfaces. *ACM SIGGRAPH Computer Graphics 18*, 3 (July 1984), 213–222. doi:10.1145/964965.808601. 2

[Hav] HAVEN P.: Hdris• poly haven. *URL: https://polyhaven. com/hdris*. 8

[HCZ21] HADADAN S., CHEN S., ZWICKER M.: Neural radiosity. *ACM Transactions on Graphics 40*, 6 (Dec. 2021), 1–11. doi:10.1145/3478513.3480569. 3

[HLN*23] HADADAN S., LIN G., NOVÁK J., ROUSSELLE F., ZWICKER M.: Inverse global illumination using a neural radiometric prior, 2023. arXiv:2305.02192. 3

[HZ22] HADADAN S., ZWICKER M.: Differentiable neural radiosity, 2022. URL: https://arxiv.org/abs/2201.13190. 3

[JLX*24] JIN H., LIU I., XU P., ZHANG X., HAN S., BI S., ZHOU X., XU Z., SU H.: Tensoir: Tensorial inverse rendering, 2024. URL: https://arxiv.org/abs/2304.12461. 3

[JSL*25] JIANG K., SUN J.-M., LI Z., WANG D., LI T.-M., RAMAMOORTHI R.: Differentiable light transport with gaussian surfels via adapted radiosity for efficient relighting and geometry reconstruction. *ACM Trans. Graph. 44*, 6 (Dec. 2025). URL: https://doi.org/10.1145/3763305, doi:10.1145/3763305. 3

[JSRV22] JAKOB W., SPEIERER S., ROUSSEL N., VICINI D.: Dr.Jit: A just-in-time compiler for differentiable rendering. *Transactions on Graphics (Proceedings of SIGGRAPH) 41*, 4 (July 2022). doi:10.1145/3528223.3530099. 2, 8

[Kaj86] KAJIYA J. T.: The rendering equation. In *Proceedings of the 13th annual conference on Computer graphics and interactive techniques* (1986), pp. 143–150. 2

[KB14] KINGMA D. P., BA J.: Adam: A method for stochastic optimization, 2014. Version Number: 9. doi:10.48550/ARXIV.1412.6980. 11

[KBM*20] KATO H., BEKER D., MORARIU M., ANDO T., MATSUOKA T., KEHL W., GAIDON A.: Differentiable Rendering: A Survey, 2020. Version Number: 2. doi:10.48550/ARXIV.2006.12057. 2

[KKLD23] KERBL B., KOPANAS G., LEIMKÜHLER T., DRETTAKIS G.: 3D Gaussian splatting for real-time radiance field rendering, 2023. _eprint: 2308.04079. URL: https://arxiv.org/abs/2308.04079. 3

[LADL18] LI T.-M., AITTALA M., DURAND F., LEHTINEN J.: Differentiable Monte Carlo ray tracing through edge sampling. *ACM Trans. Graph. (Proc. SIGGRAPH Asia) 37*, 6 (2018), 222:1–222:11. 2

[LB14] LOPER M. M., BLACK M. J.: OpenDR: An approximate differentiable renderer. In *Computer Vision – ECCV 2014* (Cham, 2014), Fleet D., Pajdla T., Schiele B., Tuytelaars T., (Eds.), Springer International Publishing, pp. 154–169. 2

[LHJ19] LOUBET G., HOLZSCHUCH N., JAKOB W.: Reparameterizing discontinuous integrands for differentiable rendering. *Transactions on Graphics (Proceedings of SIGGRAPH Asia) 38*, 6 (Dec. 2019). doi:10.1145/3355089.3356510. 2

Lowres version

[LHK*20] LAINE S., HELLSTEN J., KARRAS T., SEOL Y., LEHTINEN J., AILA T.: Modular primitives for high-performance differentiable rendering. *ACM Transactions on Graphics 39*, 6 (2020). 2, 11

[MST*20] MILDENHALL B., SRINIVASAN P. P., TANCIK M., BARRON J. T., RAMAMOORTHI R., NG R.: NeRF: Representing scenes as neural radiance fields for view synthesis. *CoRR abs/2003.08934* (2020). arXiv: 2003.08934. URL: https://arxiv.org/abs/2003.08934. 3

[NDDJK21] NIMIER-DAVID M., DONG Z., JAKOB W., KAPLANYAN A.: Material and Lighting Reconstruction for Complex Indoor Scenes with Texture-space Differentiable Rendering. In *Eurographics Symposium on Rendering - DL-only Track* (2021), Bousseau A., McGuire M., (Eds.), The Eurographics Association. doi:10.2312/sr.20211292. 6

[NDSRJ20] NIMIER-DAVID M., SPEIERER S., RUIZ B., JAKOB W.: 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. 2, 6

[NDVZJ19] NIMIER-DAVID M., VICINI D., ZELTNER T., JAKOB W.: Mitsuba 2: a retargetable forward and inverse renderer. *ACM Trans. Graph. 38*, 6 (Nov. 2019). Place: New York, NY, USA Publisher: Association for Computing Machinery. doi:10.1145/3355089.3356498. 2

[NJJ21] NICOLET B., JACOBSON A., JAKOB W.: Large steps in inverse rendering of geometry. *ACM Transactions on Graphics (Proceedings of SIGGRAPH Asia) 40*, 6 (Dec. 2021). doi:10.1145/3478513.3480501. 3, 11

[NRN*23] NICOLET B., ROUSSELLE F., NOVÁK J., KELLER A., JAKOB W., MÜLLER T.: Recursive control variates for inverse rendering. *Transactions on Graphics (Proceedings of SIGGRAPH) 42*, 4 (Aug. 2023). doi:10.1145/3592139. 2

[Pel99] PELLEGRINI M.: Rendering equation revisited: how to avoid explicit visibility computations. In *Proceedings of the Tenth Annual ACM-SIAM Symposium on Discrete Algorithms* (USA, 1999), SODA '99, Society for Industrial and Applied Mathematics, p. 725–733. 2

[PGM*19] PASZKE A., GROSS S., MASSA F., LERER A., BRADBURY J., CHANAN G., KILLEEN T., LIN Z., GIMELSHEIN N., ANTIGA L., DESMAISON A., KÖPF A., YANG E., DEVITO Z., RAISON M., TEJANI A., CHILAMKURTHY S., STEINER B., FANG L., BAI J., CHINTALA S.: PyTorch: An imperative style, high-performance deep learning library, 2019. Version Number: 1. doi:10.48550/ARXIV.1912.01703. 11

[RRN*20] RAVI N., REIZENSTEIN J., NOVOTNY D., GORDON T., LO W.-Y., JOHNSON J., GKIOXARI G.: Accelerating 3D deep learning with PyTorch3D. *arXiv:2007.08501* (2020). 11

[SAG94] SMITS B., ARVO J., GREENBERG D.: A clustering algorithm for radiosity in complex environments. In *Proceedings of the 21st Annual Conference on Computer Graphics and Interactive Techniques* (New York, NY, USA, 1994), SIGGRAPH '94, Association for Computing Machinery, pp. 435–442. doi:10.1145/192161.192277. 2, 12

[SDS95] SILLION F., DRETTAKIS G., SOLER C.: A Clustering Algorithm for Radiance Calculation In General Environments. *Eurographics Workshop on Rendering Techniques* (July 1995). ISBN: 978-3-211-82733-8. doi:10.1007/978-3-7091-9430-0_19. 2, 12

[SDZ*20] SRINIVASAN P. P., DENG B., ZHANG X., TANCIK M., MILDENHALL B., BARRON J. T.: Nerv: Neural reflectance and visibility fields for relighting and view synthesis, 2020. URL: https://arxiv.org/abs/2012.03927. 3

[SF16] SCHÖNBERGER J. L., FRAHM J.-M.: Structure-from-motion revisited. In *Conference on Computer Vision and Pattern Recognition (CVPR)* (2016). 10

[SMK05] SEITZ S. M., MATSUSHITA Y., KUTULAKOS K. N.: A theory of inverse light transport. In *Proceedings of the Tenth IEEE International Conference on Computer Vision - Volume 2* (USA, 2005), ICCV '05, IEEE Computer Society, p. 1440–1447. URL: https://doi.org/10.1109/ICCV.2005.25, doi:10.1109/ICCV.2005.25. 3

[SP89] SILLION F., PUECH C.: A general two-pass method integrating specular and diffuse reflection. In *Proceedings of the 16th Annual Conference on Computer Graphics and Interactive Techniques* (New York, NY, USA, 1989), SIGGRAPH '89, Association for Computing Machinery, pp. 335–344. doi:10.1145/74333.74368. 2

[SWW*24] SHI Y., WU Y., WU C., LIU X., ZHAO C., FENG H., ZHANG J., ZHOU B., DING E., WANG J.: Gir: 3d gaussian inverse rendering for relightable scene factorization, 2024. URL: https://arxiv.org/abs/2312.05133. 3

[VSJ21] VICINI D., SPEIERER S., JAKOB W.: 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. 3, 5, 11

[WCG87] WALLACE J. R., COHEN M. F., GREENBERG D. P.: A two-pass solution to the rendering equation: A synthesis of ray tracing and radiosity methods. *SIGGRAPH Comput. Graph. 21*, 4 (Aug. 1987), 311–320. Place: New York, NY, USA Publisher: Association for Computing Machinery. doi:10.1145/37402.37438. 2

[YCB*23] YE W., CHEN S., BAO C., BAO H., POLLEFEYS M., CUI Z., ZHANG G.: IntrinsicNeRF: Learning intrinsic neural radiance fields for editable novel view synthesis. In *Proceedings of the IEEE/CVF International Conference on Computer Vision* (2023), pp. 339–351. 3

[YDMH99] YU Y., DEBEVEC P., MALIK J., HAWKINS T.: Inverse global illumination: recovering reflectance models of real scenes from photographs. In *Proceedings of the 26th annual conference on Computer graphics and interactive techniques - SIGGRAPH '99* (1999), ACM Press, pp. 215–224. doi:10.1145/311535.311559. 2

[ZCC16] ZHANG E., COHEN M. F., CURLESS B.: Emptying, refurnishing, and relighting indoor spaces. *ACM Transactions on Graphics 35*, 6 (Nov. 2016), 1–14. doi:10.1145/2980179.2982432. 2

[ZRJ23] ZHANG Z., ROUSSEL N., JAKOB W.: Projective sampling for differentiable rendering of geometry. *Transactions on Graphics (Proceedings of SIGGRAPH Asia) 42*, 6 (Dec. 2023). doi:10.1145/3618385. 2

Lowres version