Differentiable Linearly Transformed Cosines for Inverse Area Light Shading

ANONYMOUS AUTHOR(S)
SUBMISSION ID: 913

Fig. 1. Our inverse rendering system captures both the BRDF material and geometry of real-world objects using active area lighting. Leveraging differentiable LTC, the area lighting provides a broader range of BRDF samples while maintaining rendering efficiency similar to a single point light. This enables fast, high-fidelity inverse rendering, allowing for accurate renderings of novel views under various lighting conditions.

Fully decomposing material, geometry, and lighting is particularly challenging. We propose an inverse rendering method that uses active area lighting, where inverse area light shading is calculated using differential linearly transformed cosines (LTC). Area lighting with LTC enables efficient rendering without Monte Carlo integration while still providing a wider range of BRDF sampling per shot than point lighting, which makes material reconstruction more accurate. Additionally, we introduce a visibility weighting scheme for shadows as traditional LTC methods cannot handle these. We integrate our approach into both mesh and 3D Gaussian splatting pipelines, where it improves BRDF reconstruction in both cases with a PSNR improvement of >3 dB for relighting. Further, in a relighting experiment, we need 1/4 of the input photos for the same quality than using point lights. Our experiments show that our method outperforms state-of-the-art environment light and point light approaches, providing superior fidelity, stability in material and geometry reconstruction, within 15 minutes on average.

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

Additional Key Words and Phrases: Inverse Rendering, Point Light, Area Light, Linearly Transformed Cosines

ACM XXXX-XXXX/2025/4-ART
https://doi.org/10.1145/nnnnnnn.nnnnnnn

ACM Reference Format:
Anonymous Author(s). 2025. Differentiable Linearly Transformed Cosines for Inverse Area Light Shading. 1, 1 (April 2025), 11 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Within visual computing, inverse rendering is the process of recovering the geometry and material of the real world from many captured photographs by modeling light transport. Once recovered, we can render real-world scenes and objects from arbitrary viewpoints under novel lighting conditions, and edit physically-based material appearance properties. This makes it valuable for applications like game production and extended reality. Inverse rendering is challenging because the relationship between appearance and physical properties is under-constrained. Ambiguities often arise between material and lighting, where color variations caused by lighting can be mistakenly attributed to material albedo or specularity. This requires many constraints from many photographs to resolve.

One way to reduce optimization complexity is by reducing the degrees of freedom (DOF) in the lighting model. For example, using controlled point lights in a dark room can produce high quality results. Within this setting, our work investigates area lights as a setup for object capture in a dark room (Fig. 1), using an 'active' setting where the light is attached to the camera to induce different angular samplings of the surface BRDF. Per photo, point lights sample one angle per surface point, and previous methods have used (nearly) co-located point lights [Zhang et al. 2022a], separated point lights [Gao

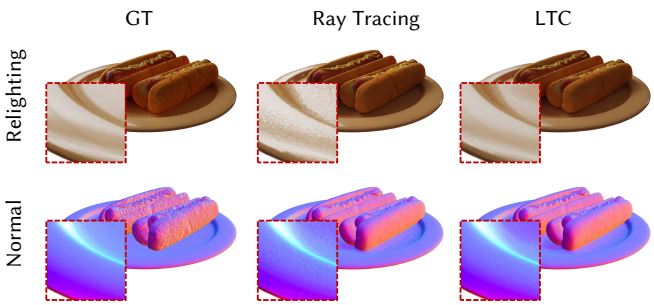

Fig. 2. **Ray tracing vs. LTC**. Ray tracing is noisy even when we employ multi-importance sampling [Veach 1998] and a bilateral denoiser. We increase the contrast of normal map visualization to highlight the noise.

et al. 2020; Kuang et al. 2024], polarimetric point lights [Hwang et al. 2022], or LED arrays with multiple point lights [Bi et al. 2024a; Ma et al. 2021a]. But area lights sample multiple angles per surface point, potentially allowing for higher quality or more efficient capture.

We propose an area light approach that offers three advantages:

(1) Efficient and high quality shading: The shading of a surface point under an area light can be approximated efficiently using Linearly Transformed Cosines (LTC) [Heitz et al. 2016]. Area lights are often rendered with expensive Monte Carlo (MC) integration via ray tracing, but LTC integration is closed form and so can improve reconstruction efficiency. Its approximation is sufficient to maintain high quality, it avoids the high memory usage associated with MC sampling during gradient calculations, and it eliminates the noise from insufficient MC sampling (Fig. 2). **Yaoan:** We need to remove memory usage according to reviews

(2) Higher BRDF sampling efficiency: Area lights give broader coverage of the 4-dimensional BRDF function space in a single capture than point lights. Specifically, for specular material regions, an area light from a fixed viewing direction emits rays that have a higher probability of producing non-zero specular reflections.

(3) Simplicity: Planar rectangular area lights with flat response are now cheap to buy, so they are as easy to set up and use as a point light. This makes it a practical choice for real-world applications.

To the best of our knowledge, no existing work uses differentiable LTC for inverse rendering even though it has wide potential. Our differentiable LTC method is implemented in CUDA and accommodates the optimization stability issue of nearly adjacent polygonal light vertices when clipping to the surface tangent plane. Since LTC can only represent shading and not shadowing, we use a limited amount of ray tracing via per-view area-guided visibility maps to add shadowing back in a computationally-efficient way.

To show the flexibility of our lighting method, we use it with two inverse renderers for object reconstruction: a mesh pipeline and a 3D Gaussian Splatting (3DGS) pipeline. For meshes, we extend NVDiffRec [Munkberg et al. 2022] to optimize a SVBRDF textured mesh. For 3DGS, we extend the recent relightable 3DGS method R3DG [Gao et al. 2023], where each Gaussian has SVBRDF parameters. First, we affix an LED area light to the camera and calibrate its relative pose (Fig. 1). Then, we capture a set of multi-view images and reconstruct an initial coarse geometry using off-the-shelf photogrammetry tools. Next, we use differentiable LTC to efficiently optimize the geometry and BRDF materials to match the input photographs. Our method shows superior SVBRDF reconstruction quality in both mesh and Gaussian pipelines to SOTA baselines.

Contributions.

- An efficient and stable differentiable LTC shading algorithm for area lights, with a visibility-aware weighting for shadows. Given initial geometry, our inverse rendering pipeline is 5× faster than compared neural inverse rendering methods that also use initial geometry. Our area light approach is more sample efficient than point lighting, achieving higher quality from fewer samples.
- An object reconstruction pipeline using inverse rendering from photographs that achieves high-fidelity physically-based material reconstruction and refined geometry reconstruction. We show this using both mesh-based and 3DGS-based pipelines, achieving competitive or better SVBRDF reconstruction results compared to existing point-light-based approaches.

Assumptions and limitations. Our work does not model shading effects due to interreflection, nor effects from transmissive materials like glass or highly specular mirror. Our final quality is dependent upon the initial geometry, and large defects in this geometry cannot be automatically refined correctly.

2 RELATED WORKS

Neural inverse rendering decomposes scene appearance into geometry, material, and lighting with neural networks from multiple observed images. The development of NeRF [Mildenhall et al. 2021] has spurred a series of neural inverse rendering methods based on implicit scene representations. Some methods constrain the lighting of input images to an environment map [Srinivasan et al. 2021; Zhang et al. 2021b], while others allow input images from varying lighting environments [Boss et al. 2021a,b; Yao et al. 2022].

Accounting for global illumination, NeILF [Yao et al. 2022] introduces a neural incident light field to model the direct and indirect illumination of the scene. TensoIR [Jin et al. 2023] performs a secondary ray tracing to compute accurate visibility and indirect lighting, which enables accurate physically-based rendering. The aforementioned NeRF-based methods often struggle to reconstruct fine geometries. To address this problem, PhySG [Zhang et al. 2021a] utilizes a signed distance field (SDF) to represent scene geometry and employs sphere tracing to achieve precise ray-geometry intersections. However, this method does not take indirect illumination into account, leading to baked-in artifacts in the predicted materials. Lately, several SDF-based methods [Liu et al. 2023; Wu et al. 2023; Zhang et al. 2023b, 2022b] that model global illumination have been proposed to improve the accuracy of material-lighting decoupling.

The recent success of 3DGS in implicit scene representation has drawn great attention in the field of inverse rendering [Wu et al. 2024]. Deferred shading of rasterization is applied for realistic rendering and relighting. GS-IR [Liang et al. 2024] proposed a depth-based regularization term for normal estimation and a cube map-based baking strategy to model occlusion and indirect illumination. R3DG [Gao et al. 2023] employs a similar normal distillation strategy, but it associates a set of spherical harmonic (SH) coefficients with each Gaussian to represent indirect illumination. It uses

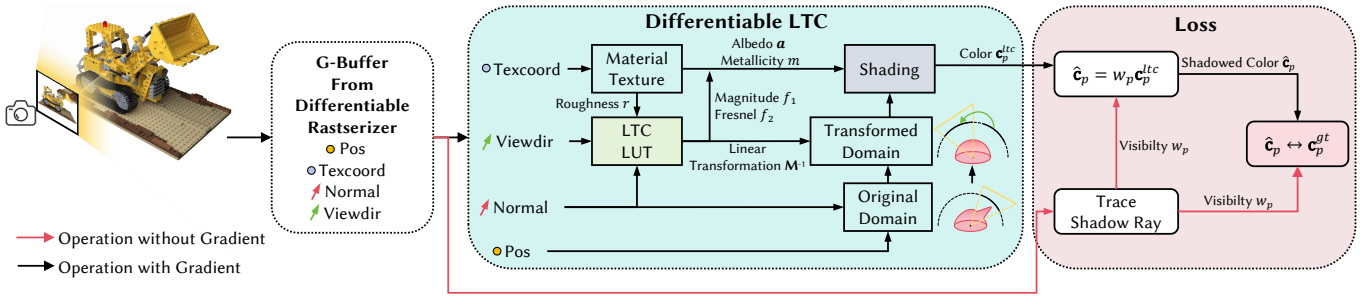

Fig. 3. **Our pipeline** reconstructs the geometry and BRDF materials of the target object using a set of images captured in a dark room. Built upon NVDiffrecMC [Hasselgren et al. 2022], our method first reconstructs the initial geometry. The mesh with textures is then differentiably rasterized to the G-buffer. For the shading points of each pixel, we apply differentiable LTC to efficiently compute its shading color. The original area light integration under complex BRDFs is transformed linearly to achieve a closed-form solution. Next, the LTC shading color is scaled using a per-pixel visibility map to account for shadow effects. Finally, the rendering loss is employed to end-to-end optimize the albedo, roughness, metallicity, and geometry.

physics-based rendering to compute the radiance of each Gaussian, ultimately blending them to produce the final rendering.

Different from these implicit scene representation methods, NVdiffrec [Munkberg et al. 2022] and NVdiffrec-MC [Hasselgren et al. 2022] directly optimize explicit meshes using a differentiable marching cube method DMTet [Shen et al. 2021], making them more compatible with modern graphics pipelines.

Inverse rendering with active lighting. Active lighting, which is used to reduce the dimensionality of the optimization space, is an effective way to alleviate the ambiguity in inverse rendering. Here, we focus on neural-based methods and will not introduce traditional active lighting methods [Gardner et al. 2003; Ghosh et al. 2009; Nam et al. 2018; Ren et al. 2011; Riviere et al. 2014; Schmitt et al. 2020; Wang et al. 2011; Zhou et al. 2013] in detail.

The point light assumption restricts the lighting distribution function to a Dirac delta function, allowing the integral of the rendering process to be computed with a single sample. It significantly simplifies the rendering, making it one of the most common active light sources. Bi et al. [2020b] optimize the geometry and SVBRDF of the object by training a depth estimation network and a reflectance estimation network. Subsequently, various methods combining point lighting with different scene representations have been proposed, such as NeRF-based [Bi et al. 2020a], SDF-based [Zeng et al. 2023; Zhang et al. 2022a], hybrid point-volumetric-based [Chung et al. 2024], 3DGS-based [Bi et al. 2024b] approaches. For avatar reconstruction, Lu et al. [2024] and Han et al. [2024] proposed methods to reconstruct the geometry and materials of the human body or face from images taken with a mobile phone and flashlight.

In addition to point lights, there are also more complex active light configurations, such as LED arrays in a light stage, which could enhance the quality of material decomposition [Kang et al. 2018, 2019]. Ma et al. [2021b] simplify the light-stage capturing setup by using only one camera and a collocated RGB LED array. Although the capture setup of this method is similar to ours, it sums many point light estimates rather than directly estimating flat area lighting, and empirically it requires high-precision initial geometry. Zhang et al. [2023a] estimate the SVBRDF of a planar sample from a single image captured under an RGB LCD area light and a camera without careful calibration. Although both this method and ours use LTC for physics-based rendering, this method is only applicable to planar objects and requires a large amount of training data.

3 METHOD

Our method reconstructs the geometry and BRDF materials of a target object using a set of photographs captured in a dark room with a camera equipped with a fixed planar area light source. We assume that the light source has a constant radiance, denoted by L, and that we know via calibration its relative pose with respect to the camera [Whelan et al. 2018]. We assume an initial coarse geometry.

For efficient inverse rendering, we introduce a differentiable LTC [Heitz et al. 2016] pipeline (Section 3.1) to quickly produce high-quality approximate shading on complex geometry and materials (Fig. 3). Since LTC ignores visibility and causes errors in shadow areas, we use an area-guided visibility scheme to overcome this limitation (Section 3.2). we integrate our differentiable LTC method into both mesh-based and 3DGS-based inverse rendering pipelines (Section 3.4) for fast and high-quality reconstruction.

Material representation. We follow previous work [Munkberg et al. 2022] in differentiable rendering and use the physically-based (PBR) material model from Disney [Burley and Studios 2012].This lets us easily import game assets and render our optimized models directly in existing engines without modifications. It is characterized by three key properties: albedo $\mathbf{a} \in [0,1]^3$, roughness $r \in [0,1]$, and metallicity $m \in [0,1]$. It combines two components: a diffuse term and an isotropic specular GGX lobe [Walter et al. 2007]:

$$\rho(\mathbf{x}, \omega_v, \omega_l) = \frac{(1-m)\mathbf{a}}{\pi} + \frac{D(r)F(m,\mathbf{a})G(r)}{4|\omega_v \cdot \mathbf{n}||\omega_l \cdot \mathbf{n}|}, \tag{1}$$

where ρ is the BRDF, $\mathbf{x}$ represents the shading point, ω_v and ω_l denote the view (surface-to-camera) and incident light (surface-to-light) directions, respectively, and $\mathbf{n}$ is the surface normal. D is the normal distribution function (NDF) that depends on roughness, which uses the GGX model [Trowbridge and Reitz 1975]. F is the Fresnel term and G models the shadowing effect between microfacets. Note that the simplified Disney BRDF does not account for tint, sheen, or subsurface effects.

3.1 Differentiable Linearly Transformed Cosines

In physically-based rendering, shading with a polygonal area light of constant radiance L requires computing the illumination integral over the spherical domain $\mathcal{P}$ covered by the area light:

$$\mathbf{c}_p(\mathbf{x}, \omega_v) = L \int_{\mathcal{P}} \rho(\mathbf{x}, \omega_v, \omega_l) \cos\theta_l \mathrm{d}\omega_l, \tag{2}$$

where $\mathbf{c}_p$ is the color of pixel p, $\mathbf{x}$ is the intersection point along the view direction ω_v, and $\cos\theta_l = \omega_l \cdot \mathbf{n}$ denotes the cosine of the angle between the incident light direction and the surface normal. Computing this integral typically requires Monte Carlo integration, which typically samples many rays to achieve an accurate result.

However, we can efficiently approximate this integral using linearly transformed cosines (LTC) [Heitz et al. 2016]. LTC exploits the fact that integrals of cosine distributions have closed-form solutions for polygonal light, and this distribution can be transformed to approximate BRDFs, which results in a fast integration method without relying on sampling and Monte Carlo integration. For a given view direction, if a linear transformation $\mathbf{M} \in \mathbb{R}^{3\times3}$ can be identified to map a cosine distribution to an approximation of the complex cosine-weighted BRDF $\rho(\mathbf{x}, \omega_v, \omega_l)\cos\theta_l$. Assuming isotropic BRDFs, and given that LTC are scale-invariant, each $\mathbf{M}$ has only 4 parameters [Heitz et al. 2016]. Its inverse transformation $\mathbf{M}^{-1}$ can then be used to simplify the complex integration in Eq. 2 to a straightforward cosine distribution within the transformed spherical domain $\mathcal{P}' = \mathbf{M}^{-1}\mathcal{P}$. This transformed integration $E(\mathcal{P}')$ admits a closed-form analytical solution [Heitz 2017; Lambert 1760], which equals the sum of several (signed) areas corresponding to the edges along the light source boundary:

$$\begin{aligned} E(\mathcal{P}') &= L \int_{\mathcal{P}'} \cos\theta_l \mathrm{d}\omega_l \\ &= \frac{L}{2\pi} \sum_{\langle \mathbf{p}_i, \mathbf{p}_j \rangle \in \partial\mathcal{P}'} \mathrm{acos}(\mathbf{p}_i \cdot \mathbf{p}_j) \left(\frac{\mathbf{p}_i \times \mathbf{p}_j}{||\mathbf{p}_i \times \mathbf{p}_j||} \cdot \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \right), \end{aligned} \tag{3}$$

where $\partial\mathcal{P}'$ denotes the boundary of $\mathcal{P}'$, and $\mathbf{p}_i$ and $\mathbf{p}_j$ are two consecutive vertices along the boundary.

To map the cosine-weighted BRDF to the cosine distribution, $\mathbf{M}^{-1}$ depends on both the material and the view direction. Following the approach of Heitz et al. [2016], for a fixed view direction, we precompute $\mathbf{M}$ for each specular term in the BRDF and view direction at fixed intervals and store the $\mathbf{M}^{-1}$ values. These values are interpolated during rendering. To reduce storage requirements, as in Heitz et al. [2016], the Fresnel term is separated from the BRDF integral and treated as an independent factor influencing the BRDF magnitude. This allows $\mathbf{M}^{-1} = \mathbf{M}^{-1}(r, \beta)$ to depend solely on the specular term's roughness r and the dot product of the view direction ω_v and the surface normal $\mathbf{n}$, denoted as β, enabling it to be efficiently stored in a 2D look-up table (LUT).

With linear transformations $\mathbf{M}^{-1}(r, \beta)$ and the closed-form integration in Eq. 3, the specular components of a shading point $\mathbf{p}$ can be calculated analytically as:

$$\mathbf{c}_p^s = F(r, \beta) E(\mathcal{P}'), \tag{4}$$

$$\mathcal{P}' = \left\{ \mathbf{M}^{-1}(r, \beta)\mathbf{p}_i | \mathbf{p}_i \in \mathcal{P} \right\}, \tag{5}$$

$$F(r, \beta) = F_0 f_1(r, \beta) + (1 - F_0) f_2(r, \beta), \tag{6}$$

where $F_0 = 0.04(1 - m) + m\mathbf{a}$ is the basic reflectance (the fraction of light that is reflected at normal incidence), and $F(r, \beta)$ compensates for energy loss caused by the shadowing term and the omission of the Fresnel term (including albedo and metallicity) in the fitting of $\mathbf{M}$. Specifically, f_1 is the magnitude of the BRDF, which is used as a scale factor to compensate for the energy loss due to the shadowing term, and f_2 is the factor for the Fresnel reflectance. f_1 and f_2 are also precomputed and can be stored together in a second LUT.

The diffuse component can be directly obtained by integrating over the cosine distribution without linear transformation:

$$\mathbf{c}_p^d = \frac{(1 - m)a}{\pi} E(\mathcal{P}). \tag{7}$$

The final rendering color using LTC is given by $\mathbf{c}_p^{ltc} = \mathbf{c}_p^s + \mathbf{c}_p^d$.

LTC in autodiff. The 2D LUT tables for $\mathbf{M}^{-1}$, f_0, f_1 enables us to compute the derivative of these quantities with respect to the roughness r and β through numerical gradients using forward differences. These derivatives can be used in the chain rule for the derivative computation in Eq. 4 to obtain the derivative of the specular components $\mathbf{c}_p^s$ with respect to r and β. Since the vertex positions are used to calculate the β, the derivatives of $\mathbf{c}_p^s$ can be back-propagated back to the vertex positions through β. The derivative chain can be analyzed for $\mathbf{c}_p^d$ in a similar way, and the derivatives to albedo a and metallicity m are obtained through the basic reflectance F_0 in the specular and diffuse components. We show the derivative chain for differentiable LTC in the supplementary material.

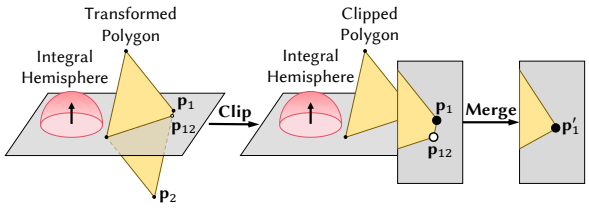

Fig. 4. **Light source clipping.** We must clip the transformed polygon to the surface tangent plane before projecting it onto the integral sphere. During clipping, it is possible that a new vertex P_{12} generated from clipping is close to another vertex P_1, which causes unstable inverse rendering. We merge such vertices to mitigate the problem.

Merging nearby light corners after clipping. During rendering, we clip each area light polygon to the point's tangent plane. Thus, it is possible for two clipped vertices to become close, as shown in Fig. 4. Two close vertices can cause problems when analytically integrating the cosine distribution over the domain of the clipped polygon using Eq. 3. First, in the forward process, the cross product in the denominator of Eq. 3 can be near zero. Second, in the backward process, the gradient of the arccos function near 1 approaches infinity, which can cause instability. To resolve this issue, we propose merging adjacent points $\mathbf{p}_i$ and $\mathbf{p}_j$ that are too close ($\mathbf{p}_i \cdot \mathbf{p}_j > 1 - 10^{-4}$) during both forward rendering and back-propagation. This causes only a minor

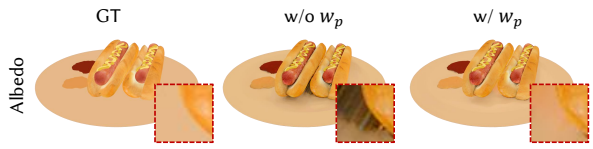

Fig. 5. **Visibility map w_p.** Without precomputed visibility, optimization bakes contact shadows into the albedo, which hurts relighting quality.

effect because the contribution from the small edge between close points to the overall rendering is small.

3.2 Area-guided visibility

While LTC can efficiently and accurately approximate the effects of complex materials and lighting, the LTC-rendered color $\mathbf{c}_p^{ltc}$ ignores occlusion caused by geometry, resulting in biased reconstruction outcomes such as shadows being baked into the albedo. Although Ambient Occlusion (AO) can approximate these shadows, it assumes uniform light intensity in all directions, leading to significant errors under our area light setup. To address this, we propose an area-guided visibility weighting inspired by Heitz et al. [2018]. A shadowed color $\hat{\mathbf{c}}_p$ can be decomposed into a color without visibility, which can be approximately calculated using LTC $\mathbf{c}_p^{ltc}$, multiplied by a visibility map w_p as following:

$$\hat{\mathbf{c}}_p(\mathbf{x}, \omega_v) = L \int_{\mathcal{P}} V(\mathbf{x}, \omega_l) \rho(\mathbf{x}, \omega_v, \omega_l) \cos\theta_l \mathrm{d}\omega_l \tag{8}$$

$$= w_p(\mathbf{x}, \omega_v) \mathbf{c}_p(\mathbf{x}, \omega_v) \approx w_p(\mathbf{x}, \omega_v) \mathbf{c}_p^{ltc}(\mathbf{x}, \omega_v), \tag{9}$$

$$w_p(\mathbf{x}, \omega_v) = \frac{\int_{\mathcal{P}} V(\mathbf{x}, \omega_l) \rho(\mathbf{x}, \omega_v, \omega_l) \cos\theta_l \mathrm{d}\omega_l}{\int_{\mathcal{P}} \rho(\mathbf{x}, \omega_v, \omega_l) \cos\theta_l \mathrm{d}\omega_l}, \tag{10}$$

where $V(\mathbf{x}, \omega_l)$ represents the visibility of incident light at the shading point $\mathbf{x}$. The visibility map reduces the weight of shadowed areas, making them contribute less to the geometry and material estimation of certain points and largely eliminating issues like baked shadows in albedo (Fig. 5).

The visibility map w_p can be solved numerically using ray tracing and Monte Carlo integration. This may initially seem counter to our goals of using LTC—why not just use ray tracing for everything? But, we can use the visibility map solely as a weighting factor without autodifferentiation, allowing for efficient implementation using NVIDIA OptiX. For the given geometry and material, we compute $w_\mathbf{p}$ for each pixel in the training view using 1,024 sampled incident rays ω_l on the spherical domain $\mathcal{P}$. This takes 15 seconds for 200 input images. Further, empirically, we need only update the visibility map every 1,000 iterations to produce good results, which significantly reduces the computational cost. Even though the visibility map is out of date, it remains helpful during optimization because, as we use active lighting and so as the shadow varies with each input view, there are often notable areas of shadow that must be efficiently accommodated.

3.3 LTC in a mesh-based optimization

We use a triangular 3D mesh with textures to represent object geometry, albedo, roughness, and metallicity maps. Once reconstructed, a mesh with textures is easy to edit within existing digital content creation tools. Using meshes lets us exploit hardware-accelerated differentiable rasterization [Laine et al. 2020] and hardware-accelerated ray-tracing for efficient shadow rays (Fig. 3.2).

We optimize the vertex positions, normal map as offset from the geometry normal, material parameters (albedo $\mathbf{a}$, roughness r, metallicity m), and radiance L of the rectangular area light. For initialization, we set albedo to be 1.0, metallicity to be 0.0, and we randomly initialize roughness and the light radiance.

Losses. To optimize the geometry and material parameters, we define a loss $\mathcal{L}$ as weighted combination of the image loss and a set of regularizer terms:

$$\mathcal{L} = \mathcal{L}_{\text{render}} + \lambda_{\text{arm}}\mathcal{L}_{\text{arm}} + \lambda_{\text{b}}\mathcal{L}_{\text{b}} + \lambda_{\text{n}}\mathcal{L}_{\text{n}} + \lambda_{\text{nm}}\mathcal{L}_{\text{nm}} + \lambda_{\text{nc}}\mathcal{L}_{\text{nc}}, \tag{11}$$

where λ is the weight of each term. We set $\lambda_{\text{arm}} = 0.1$, $\lambda_{\text{n}} = 0.025$, $\lambda_{\text{nm}} = 1.0$ and $\lambda_{\text{nc}} = \lambda_{\text{b}} = 0.1$.

The rendering loss $\mathcal{L}_{\text{render}}$ measure the difference between captured pixel colors $\mathbf{c}_p^{gt}$ and rendered pixel colors $\hat{\mathbf{c}}_p$. Given the error introduced by visibility approximation and significant indirect lighting in the shadowed regions, we modulate the color loss with the visibility map w_p to reduce the gradient from these regions:

$$\mathcal{L}_{\text{render}} = \begin{cases} 0, & \text{if } \mathbf{c}_p^{gt} \le \hat{\mathbf{c}}_p \text{ and } \mathbf{c}_p^{gt} \text{ is overexposed,} \\ \sum_p w_p ||\mathbf{c}_p^{gt} - \hat{\mathbf{c}}_p||_2^2, & \text{otherwise.} \end{cases} \tag{12}$$

To handle overexposure, where colors are clamped to the maximum value of the image format, we detect overexposed pixels by checking if their color values reach the format's maximum (e.g., 255).

To enhance the ill-posed roughness-metallicity optimization, we introduce a metallicity binary loss. Inspired by a suggestion from Unreal [Epic Games 2024] for Disney principled BRDF, we add binary loss encouraging metallicity to be either 0 or 1 for pure surfaces such as pure metal, pure stone, pure plastic, and so on.

$$\mathcal{L}_{\text{b}} = \sum_p m_p \cdot (1 - m_p), \tag{13}$$

where m_p is the screen-space metallicity of pixel p.

Next, we apply smoothness priors in NVdiffrec-MC [Hasselgren et al. 2022] for albedo, specular, metallicity ($\mathcal{L}_{\text{arm}}$), surface normal ($\mathcal{L}_{\text{n}}$), and normal map textures ($\mathcal{L}_{\text{nm}}$). Additionally, we apply the smoothness prior to L_{nc} to enforce normal consistency across randomly sampled adjacent triangles.

3.4 LTC in a 3DGS-based optimization

Optimizing mesh geometry without a plausible initialization can be challenging due to issues with changing topology and avoiding self-intersections. In contrast, 3DGS [Kerbl et al. 2023] uses 3D Gaussian primitives as its primary rendering entity, which can be optimized from random initialization. As a proof of concept, we integrate our area light inverse rendering using LTC into the relightable 3D Gaussian (R3DG) [Gao et al. 2023] pipeline. Specifically, we shade each Gaussian under area lighting with LTC and derive the pixel color by splatting the Gaussian points onto the screen.

4 EXPERIMENTS

Datasets. We use a synthetic dataset and a real captured dataset. The synthetic data consists of four synthetic scenes (ficus, lego,

armadillo, and hotdog) as used in TensoIR [Jin et al. 2023]. We use Blender's Cycles renderer [Community 2018] to produce 200 training images and 200 testing images. The poses for the training and testing views are identical to those in the NeRF-synthetic dataset [Mildenhall et al. 2021]. For evaluation, we render the training and testing data under point, area, and environment lighting.

As shown in Fig.1, our real captured data consists of eight objects with varying materials. The capture system uses a consumer-grade DSLR camera paired with an LED area light. We set the color temperature of the area light to 5000K, which is typically considered neutral white light. The relative pose between the camera and the area light is calibrated by attaching an AprilTag[Olson 2011] to the LED light and taking photos in front of a plane mirror [Whelan et al. 2018]. The size of the area light follows the specifications of our lighting device ($15cm \times 10.6cm$). Camera poses are computed using RealityCapture [CapturingReality 2016]. For detailed information on the size of each object and the number of captured images, please refer to the supplementary material.

Geometry initialization. Our pipeline relies on initial geometry, as do comparison methods such as TensoSDF [Li et al. 2024], NVDiffrecMC [Hasselgren et al. 2022], IRON [Zhang et al. 2022a], and DPIR [Chung et al. 2024] which require either initial geometry or a geometry reconstruction stage. Although assuming constant lighting, we find that neural implicit methods like NeuS [Wang et al. 2021] and TensoSDF [Li et al. 2024] can produce satisfactory geometry in most cases under our active and dynamic lighting setup. Consequently, we use the first stage (geometry reconstruction stage) of TensoSDF [Li et al. 2024] to reconstruct the initial geometry.

After geometry reconstruction, we extract the mesh and simplify it to 100–300k faces using Quadric Error Metric (QEM) simplification [Garland and Heckbert 1997], then apply Laplacian smoothing [Vollmer et al. 1999]. Next, we generate a 1024×1024 texture map in Blender [Community 2018] for both the material and normal maps. Certain objects with highly metallic areas (Bust, Luckycat) or metallic *and* low reflectance areas (Bottle cap) are difficult; for these, to define the initial geometry, we capture a set of images under fixed environment lighting (no active area lighting); this is a limitation.

Training details. During training, we use Adam [Kingma 2014] to optimize in PyTorch [Paszke et al. 2019] with peak learning rates of 1×10^{-6}, 0.01 for the albedo, roughness, and metallically texture maps, and 0.03 for the radiance of the area light. We set 100 warm-up iterations, linearly increasing the learning rate to its peak. After that, the learning rate is exponentially decayed to 10% of the peak value during optimization. The training consists of 3k iterations, with a batch size of 8 images, and takes an average of 11 minutes on one RTX 3090 GPU. For the 3D Gaussian-based pipeline, we use the same training configuration as in R3DG [Gao et al. 2023].

Metrics. For quantitative comparisons on the BRDF estimation, novel view synthesis, and relighting results (Tab. 1), we use Peak Signal-to-Noise Ratio (PSNR), Structural Similarity (SSIM) [Wang et al. 2004], and Learned Perceptual Image Patch Similarity (LPIPS) [Zhang et al. 2018] as metrics. To assess geometry quality, we compare the estimated normal maps with the ground truth using Mean Absolute Error (MAE) and compute Chamfer distances between the reconstructed and ground truth meshes.

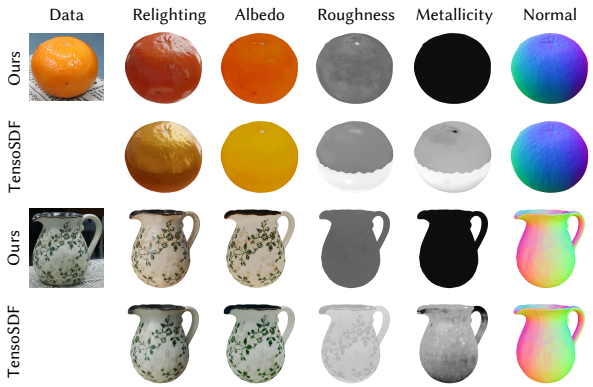

Fig. 6. Real objects results. Compared to natural environment capture in TensoSDF, our area light approach more accurately recovers PBR materials.

4.1 Comparison with the state-of-the-art

We compare our approach with four inverse rendering methods: IRON [Zhang et al. 2022a] and DPIR [Chung et al. 2024], which use active point lighting, and TensoSDF [Li et al. 2024] and NVDiffrecMC [Hasselgren et al. 2022], which assume static environmental lighting. As NVDiffrecMC [Hasselgren et al. 2022] uses ray sampling and Monte Carlo integration, we also modify it to incorporate active point and area lighting without using our LTC for a more comprehensive comparison. For area lighting without LTC, we train with 16 samples per pixel (SPP), which is the default setting in NVDiffrecMC. Increasing the SPP significantly increases the training time.

For the active lighting setting, Table 1 shows that our method significantly outperforms IRON [Zhang et al. 2022a] and DPIR [Chung et al. 2024] in terms of geometry, material, and the ability to relight under novel environmental conditions. Additionally, the training of our method is substantially faster by one order of magnitude.

When compared to NVDiffrecMC [Hasselgren et al. 2022] for point lighting, geometry and albedo are similar. For albedo, since point lighting does not suffer from shadow or visibility estimation bias, its albedo is slightly better in certain shadowed areas. However, our LTC approach produces better results for relighting thanks to more accurate roughness r: NVDiffrecMC causes specular areas to appear diffuse under relighting conditions (Fig. 7).

For NVDiffrecMC with area lighting via ray sampling and Monte Carlo integration, NVDiffrecMC produces more plausible material properties with 16 SPP and its relighting results are also better, albeit with some noise due to inefficient sampling (Fig. 2). At 16 SPP, ray tracing is 3× slower than our approach. Increasing the number of ray samples in the Monte Carlo integration can reduce noise but significantly increases training time and memory consumption.

Evaluation on real objects. We evaluate our method on real objects and compare it with TensoSDF [Li et al. 2024]. In this more challenging setting, performance is generally worse and takes longer. For real captured objects (Fig. 6), we see that TensoSDF fails to properly decompose roughness and metallicity. Additionally, for the albedo, since the color temperature of the environment lighting is not calibrated, the color temperature is baked into the albedo. For example, the albedo of the vase object exhibits cool tones, which are influenced by the environment lighting. See Fig. 11 for more results.

Method	Light model	Normal	Chamfer Dist	Albedo			Novel View Synthesis			Relighting			Runtime
		MAE ↓	mm ↓	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓	with Init.
TensoSDF [Li et al. 2024]	Static field MLP	0.028	0.0238	26.36	0.908	0.091	29.14	0.948	0.052	24.38	0.917	0.072	396mins
NVDiffrecMC [Hasselgren et al. 2022]	Static env. map	0.031	0.0235	29.40	0.949	0.056	30.80	0.960	0.045	27.78	0.935	0.061	66mins
IRON [Zhang et al. 2022a]	Active point	0.037	0.0822	21.12	0.902	0.112	22.57	0.906	0.109	23.23	0.894	0.105	571mins
DPIR [Chung et al. 2024]	Active point	0.051	0.0527	23.34	0.930	0.087	23.89	0.926	0.086	-	-	-	223mins
NVDiffrecMC [Hasselgren et al. 2022]	Active point	0.022	0.0236	31.71	0.964	0.035	30.56	0.952	0.041	29.54	0.950	0.044	13mins
	Active area (16SPP)	0.024	0.0236	31.17	0.952	0.045	30.70	0.958	0.035	29.52	0.952	0.042	29mins
Ours	Active area (LTC)	0.023	0.0235	31.48	0.958	0.041	30.96	0.961	0.033	30.40	0.965	0.037	13mins

Table 1. **Quantitative comparisons show improved NVS and relighting on synthetic data.** We evaluate normal in world space by rasterizing it into image space for each test view and then normalizing it. We scale each RGB channel of novel view synthesis and relighting results by an optimal global scalar, as per NVDiffrecMC [Hasselgren et al. 2022]. We use Mitsuba [Jakob 2010] following IRON [Zhang et al. 2022a] for PBR relighting under a novel environment map, while we use Blender Cycles [Community 2018] for mesh-based methods. DPIR uses points as its primitive, each with a basis BRDF attached [Lawrence et al. 2006]; we use their built-in pipeline to render test data. Unfortunately, DPIR has not open-sourced their relighting code. For runtime, we use a single RTX3090 GPU and do not include the time spent on geometry initialization.

Method	Novel View Synthesis			Relighting		
	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓
R3DG [Gao et al. 2023]	30.16	0.9606	0.03714	24.22	0.8950	0.07537
- with point	30.06	0.9480	0.04523	27.49	0.9222	0.06296
- with our area LTC	31.00	0.9549	0.03854	28.18	0.9291	0.05612

Table 2. **Evaluation on 3DGS-based pipeline.** We show the effectiveness of our LTC rendering loss under area lighting by replacing the environment lighting in R3DG [Gao et al. 2023] with our LTC-based area lighting.

Runtime with geometry initialization. We calculate the runtime of our method and comparison methods after geometry initialization to report efficiency. For NVDiffrecMC [Hasselgren et al. 2022], we replace its original DMTet-based geometry initialization [Hasselgren et al. 2022] with TensoSDF [Li et al. 2024], which offers better geometry reconstruction. For DPIR [Chung et al. 2024], we use its original point cloud initialization based on the visual hull.

Our inverse rendering method is an order of magnitude faster than neural inverse rendering methods such as TensoSDF [Li et al. 2024], IRON [Zhang et al. 2022a], and DPIR [Chung et al. 2024] (Tab. 1). Compared to NVDiffrecMC [Hasselgren et al. 2022] using 16 SPP, our method is 2–3 times faster on average in both static environment lighting and active area lighting. This improvement is attributed to the closed-form integration of our differentiable LTC and an efficient CUDA implementation, enabling our method to match the speed of NVDiffrecMC [Hasselgren et al. 2022] with point lighting (1 SPP) while delivering superior relighting performance.

Evaluation on 3DGS-based pipeline. We present comparisons using the 3DGS-based pipeline. We employ relightable 3D Gaussian (R3DG) [Gao et al. 2023] to evaluate the benefits of our differentiable-LTC-based area lighting. As shown in Table 2, we modify the original environment lighting model in R3DG to incorporate point lighting and area lighting. For area lighting, we use our differentiable LTC rendering. From the comparison results, we observe that the active area lighting and differentiable LTC rendering better estimate complex materials, so yielding better relighting performance.

4.2 Ablation studies

Number of views. We perform an ablation study on the number of views using the 'Hotdog' synthetic object. Additionally, we reduce the area of area lighting to a point to demonstrate the higher BRDF sampling efficiency of our active area lighting. The ablation study is conducted using the same initial geometry, with 25, 49, 100, 144, 225, and 1024 uniformly sampled views for inverse rendering under both point lighting and area lighting. The relighting results are presented in Table 3 and Fig. 8. We remove the regularizer $\mathcal{L}_{\text{arm}}$ to eliminate any interference from the smoothing term.

# Views	Point			Area		
	PSNR ↑	SSIM ↑	LPIPS ↓	SSIM ↑	PSNR ↑	LPIPS ↓
25	25.90	0.9302	0.07161	27.86	0.9359	0.05880
49	26.39	0.9293	0.07391	29.08	0.9474	0.04829
100	26.74	0.9292	0.07874	29.04	0.9466	0.04759
144	26.89	0.9288	0.08049	29.17	0.9495	0.04746
225	26.04	0.9322	0.08113	28.91	0.9474	0.04847
1024	26.09	0.9323	0.08120	29.31	0.9512	0.04607

Table 3. **Ablations on # views under point and area lighting.** Relighting quality compared to ground truth on the 'Hotdog' synthetic object.

Table 3 shows that as the number of views increases from 25 to 144, both point lighting and area lighting results improve. As the number of views increases further to 225 and 1024, the relighting results converge. However, even with just 25 views, our area light-based inverse rendering achieves significantly better relighting quality compared to point lighting with even 225 or 1024 views (Fig. 8).

Overexposure detection in $\mathcal{L}_{\text{render}}$. As shown in Fig. 10, removing overexposure detection from the rendering loss leads to a decrease in the accuracy of albedo reconstruction for real captured objects, particularly in areas with overexposed regions across most views. Since the overexposed pixels are clamped to 255, which is below their true value, the predicted albedo tends to darken in these highlight areas to fit the clamped values.

Area-guided visibility. We evaluate the effect of our visibility weight by removing it from both the final rendering color and the rendering loss. As shown in Fig. 5, without the visibility weight, the shadow under the hotdog is baked into the albedo, causing the albedo in that region to appear darker than it should be.

Metallicity binary loss. Without the metallicity binary loss, our system sometimes fails to optimize the metallicity correctly. For example, in Fig. 9, the body of the bottle is made of high-gloss plastic, but without the metallicity binary loss, our method incorrectly classifies it as metal with high metallicity.

5 CONCLUSION

Our method introduces an efficient inverse rendering pipeline for area lights using differentiable Linearly Transformed Cosines, which

enables closed-form shading integration. In an object reconstruction pipeline, this approach improves BRDF sampling efficiency, enhancing material estimation accuracy while reducing the number of views and computational cost. We integrate this into mesh-based and 3DGS-based pipelines, showing improvements in both for BRDFs in synthetic and real datasets.

REFERENCES

Sai Bi, Zexiang Xu, Kalyan Sunkavalli, Miloš Hašan, Yannick Hold-Geoffroy, David Kriegman, and Ravi Ramamoorthi. 2020a. Deep reflectance volumes: Relightable reconstructions from multi-view photometric images. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part III 16*. Springer, 294–311.

Sai Bi, Zexiang Xu, Kalyan Sunkavalli, David Kriegman, and Ravi Ramamoorthi. 2020b. Deep 3d capture: Geometry and reflectance from sparse multi-view images. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 5960–5969.

Zoubin Bi, Yixin Zeng, Chong Zeng, Fan Pei, Xiang Feng, Kun Zhou, and Hongzhi Wu. 2024a. Gs3: Efficient relighting with triple gaussian splatting. In *SIGGRAPH Asia 2024 Conference Papers*. 1–12.

Zoubin Bi, Yixin Zeng, Chong Zeng, Fan Pei, Xiang Feng, Kun Zhou, and Hongzhi Wu. 2024b. GS^3: Efficient Relighting with Triple Gaussian Splatting. In *SIGGRAPH Asia 2024 Conference Papers*.

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

Mark Boss, Varun Jampani, Raphael Braun, Ce Liu, Jonathan Barron, and Hendrik Lensch. 2021b. Neural-pil: Neural pre-integrated lighting for reflectance decomposition. *Advances in Neural Information Processing Systems* 34 (2021), 10691–10704.

Brent Burley and Walt Disney Animation Studios. 2012. Physically-based shading at disney. In *Acm Siggraph*, Vol. 2012. vol. 2012, 1–7.

CapturingReality. 2016. Reality capture, http://capturingreality.com.

Hoon-Gyu Chung, Seokjun Choi, and Seung-Hwan Baek. 2024. Differentiable Point-based Inverse Rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 4399–4409.

Blender Online Community. 2018. *Blender - a 3D modelling and rendering package*. Blender Foundation, Stichting Blender Foundation, Amsterdam. http://www.blender.org

Epic Games. 2024. *Physically Based Materials in Unreal Engine*. https://dev.epicgames.com/documentation/en-us/unreal-engine/physically-based-materials-in-unreal-engine#metallic

Duan Gao, Guojun Chen, Yue Dong, Pieter Peers, Kun Xu, and Xin Tong. 2020. Deferred neural lighting: free-viewpoint relighting from unstructured photographs. *ACM Transactions on Graphics (TOG)* 39, 6 (2020), 1–15.

Jian Gao, Chun Gu, Youtian Lin, Hao Zhu, Xun Cao, Li Zhang, and Yao Yao. 2023. Relightable 3D Gaussian: Real-time Point Cloud Relighting with BRDF Decomposition and Ray Tracing. *arXiv:2311.16043* (2023).

Andrew Gardner, Chris Tchou, Tim Hawkins, and Paul Debevec. 2003. Linear light source reflectometry. *ACM Transactions on Graphics (TOG)* 22, 3 (2003), 749–758.

Michael Garland and Paul S Heckbert. 1997. Surface simplification using quadric error metrics. In *Proceedings of the 24th annual conference on Computer graphics and interactive techniques*. 209–216.

Abhijeet Ghosh, Tongbo Chen, Pieter Peers, Cyrus A Wilson, and Paul Debevec. 2009. Estimating specular roughness and anisotropy from second order spherical gradient illumination. In *Computer Graphics Forum*, Vol. 28. Wiley Online Library, 1161–1170.

Yuxuan Han, Junfeng Lyu, and Feng Xu. 2024. High-Quality Facial Geometry and Appearance Capture at Home. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 697–707.

Jon Hasselgren, Nikolai Hofmann, and Jacob Munkberg. 2022. Shape, light, and material decomposition from images using monte carlo rendering and denoising. *Advances in Neural Information Processing Systems* 35 (2022), 22856–22869.

Eric Heitz. 2017. *Geometric derivation of the irradiance of polygonal lights*. Ph. D. Dissertation. Unity Technologies.

Eric Heitz, Jonathan Dupuy, Stephen Hill, and David Neubelt. 2016. Real-time polygonal-light shading with linearly transformed cosines. *ACM Transactions on Graphics (TOG)* 35, 4 (2016), 1–8.

Eric Heitz, Stephen Hill, and Morgan McGuire. 2018. Combining analytic direct illumination and stochastic shadows. In *Proceedings of the ACM SIGGRAPH symposium on interactive 3D graphics and games*. 1–11.

Inseung Hwang, Daniel S Jeon, Adolfo Munoz, Diego Gutierrez, Xin Tong, and Min H Kim. 2022. Sparse ellipsometry: portable acquisition of polarimetric SVBRDF and shape with unstructured flash photography. *ACM Transactions on Graphics (TOG)* 41, 4 (2022), 1–14.

Wenzel Jakob. 2010. Mitsuba renderer. [Online]. https://www.mitsuba-renderer.org/.

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

Kaizhang Kang, Zimin Chen, Jiaping Wang, Kun Zhou, and Hongzhi Wu. 2018. Efficient reflectance capture using an autoencoder. *ACM Trans. Graph.* 37, 4 (2018), 127.

Kaizhang Kang, Cihui Xie, Chengan He, Mingqi Yi, Minyi Gu, Zimin Chen, Kun Zhou, and Hongzhi Wu. 2019. Learning efficient illumination multiplexing for joint capture of reflectance and shape. *ACM Trans. Graph.* 38, 6 (2019), 165–1.

Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 2023. 3D Gaussian Splatting for Real-Time Radiance Field Rendering. *ACM Trans. Graph.* 42, 4 (2023), 139–1.

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

Zhiyi Kuang, Yanchao Yang, Siyan Dong, Jiayue Ma, Hongbo Fu, and Youyi Zheng. 2024. OLAT Gaussians for Generic Relightable Appearance Acquisition. In *SIGGRAPH Asia 2024 Conference Papers*. 1–11.

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

Jean-Henri Lambert. 1760. *Photometria sive de mensura et gradibus luminis, colorum et umbrae*. Sumptibus viduae Eberhardi Klett, typis Christophori Petri Detleffsen.

Jason Lawrence, Aner Ben-Artzi, Christopher DeCoro, Wojciech Matusik, Hanspeter Pfister, Ravi Ramamoorthi, and Szymon Rusinkiewicz. 2006. Inverse shade trees for non-parametric material representation and editing. *ACM Transactions on Graphics (ToG)* 25, 3 (2006), 735–745.

Jia Li, Lu Wang, Lei Zhang, and Beibei Wang. 2024. TensoSDF: Roughness-aware Tensorial Representation for Robust Geometry and Material Reconstruction. *ACM Transactions on Graphics (Proceedings of SIGGRAPH 2024)* 43, 4 (2024), 150:1–13.

Zhihao Liang, Qi Zhang, Ying Feng, Ying Shan, and Kui Jia. 2024. Gs-ir: 3d gaussian splatting for inverse rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 21644–21653.

Yuan Liu, Peng Wang, Cheng Lin, Xiaoxiao Long, Jiepeng Wang, Lingjie Liu, Taku Komura, and Wenping Wang. 2023. Nero: Neural geometry and brdf reconstruction of reflective objects from multiview images. *ACM Transactions on Graphics (TOG)* 42, 4 (2023), 1–22.

Jiawei Lu, Tianjia Shao, He Wang, Yong-Liang Yang, Yin Yang, and Kun Zhou. 2024. Relightable Detailed Human Reconstruction from Sparse Flashlight Images. *IEEE Transactions on Visualization and Computer Graphics* (2024).

Xiaohe Ma, Kaizhang Kang, Ruisheng Zhu, Hongzhi Wu, and Kun Zhou. 2021a. Free-form scanning of non-planar appearance with neural trace photography. *ACM Transactions on Graphics (TOG)* 40, 4 (2021), 1–13.

Xiaohe Ma, Kaizhang Kang, Ruisheng Zhu, Hongzhi Wu, and Kun Zhou. 2021b. Free-form scanning of non-planar appearance with neural trace photography. *ACM Transactions on Graphics (TOG)* 40, 4 (2021), 1–13.

Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. 2021. Nerf: Representing scenes as neural radiance fields for view synthesis. *Commun. ACM* 65, 1 (2021), 99–106.

Jacob Munkberg, Jon Hasselgren, Tianchang Shen, Jun Gao, Wenzheng Chen, Alex Evans, Thomas Müller, and Sanja Fidler. 2022. Extracting Triangular 3D Models, Materials, and Lighting From Images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 8280–8290.

Giljoo Nam, Joo Ho Lee, Diego Gutierrez, and Min H Kim. 2018. Practical svbrdf acquisition of 3d objects with unstructured flash photography. *ACM Transactions on Graphics (ToG)* 37, 6 (2018), 1–12.

Edwin Olson. 2011. AprilTag: A robust and flexible visual fiducial system. In *2011 IEEE international conference on robotics and automation*. IEEE, 3400–3407.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. 2019. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems* 32 (2019).

Peiran Ren, Jiaping Wang, John Snyder, Xin Tong, and Baining Guo. 2011. Pocket reflectometry. *ACM Transactions on Graphics (TOG)* 30, 4 (2011), 1–10.

Jérémy Riviere, Pieter Peers, and Abhijeet Ghosh. 2014. Mobile surface reflectometry. In *ACM SIGGRAPH 2014 Posters*. 1–1.

Carolin Schmitt, Simon Donne, Gernot Riegler, Vladlen Koltun, and Andreas Geiger. 2020. On joint estimation of pose, geometry and svbrdf from a handheld scanner. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 3493–3503.

Tianchang Shen, Jun Gao, Kangxue Yin, Ming-Yu Liu, and Sanja Fidler. 2021. Deep marching tetrahedra: a hybrid representation for high-resolution 3d shape synthesis. *Advances in Neural Information Processing Systems* 34 (2021), 6087–6101.

Pratul P Srinivasan, Boyang Deng, Xiuming Zhang, Matthew Tancik, Ben Mildenhall, and Jonathan T Barron. 2021. Nerv: Neural reflectance and visibility fields for relighting and view synthesis. In *Proceedings of the IEEE/CVF Conference on Computer*

Vision and Pattern Recognition. 7495–7504.

TS Trowbridge and Karl P Reitz. 1975. Average irregularity representation of a rough surface for ray reflection. *JOSA* 65, 5 (1975), 531–536.

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

Jörg Vollmer, Robert Mencl, and Heinrich Mueller. 1999. Improved laplacian smoothing of noisy surface meshes. In *Computer graphics forum*, Vol. 18. Wiley Online Library, 131–138.

Bruce Walter, Stephen R Marschner, Hongsong Li, and Kenneth E Torrance. 2007. Microfacet Models for Refraction through Rough Surfaces. *Rendering techniques* 2007 (2007), 18th.

Chun-Po Wang, Noah Snavely, and Steve Marschner. 2011. Estimating dual-scale properties of glossy surfaces from step-edge lighting. In *Proceedings of the 2011 SIGGRAPH Asia Conference*. 1–12.

Peng Wang, Lingjie Liu, Yuan Liu, Christian Theobalt, Taku Komura, and Wenping Wang. 2021. NeuS: Learning Neural Implicit Surfaces by Volume Rendering for Multi-view Reconstruction. *NeurIPS* (2021).

Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. 2004. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing* 13, 4 (2004), 600–612.

Thomas Whelan, Michael Goesele, Steven J Lovegrove, Julian Straub, Simon Green, Richard Szeliski, Steven Butterfield, Shobhit Verma, Richard A Newcombe, M Goesele, et al. 2018. Reconstructing scenes with mirror and glass surfaces. *ACM Trans. Graph.* 37, 4 (2018), 102.

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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 4295–4304.

Tong Wu, Jia-Mu Sun, Yu-Kun Lai, Yuewen Ma, Leif Kobbelt, and Lin Gao. 2024. DeferredGS: Decoupled and Editable Gaussian Splatting with Deferred Shading. *arXiv preprint arXiv:2404.09412* (2024).

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*. Springer, 700–716.

Chong Zeng, Guojun Chen, Yue Dong, Pieter Peers, Hongzhi Wu, and Xin Tong. 2023. Relighting neural radiance fields with shadow and highlight hints. In *ACM SIGGRAPH 2023 Conference Proceedings*. 1–11.

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 *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 3601–3610.

Kai Zhang, Fujun Luan, Zhengqi Li, and Noah Snavely. 2022a. Iron: Inverse rendering by optimizing neural sdfs and materials from photometric images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 5565–5574.

Kai Zhang, Fujun Luan, Qianqian Wang, Kavita Bala, and Noah Snavely. 2021a. Physg: Inverse rendering with spherical gaussians for physics-based material editing and relighting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 5453–5462.

Lianghao Zhang, Fangzhou Gao, Li Wang, Minjing Yu, Jiamin Cheng, and Jiawan Zhang. 2023a. Deep SVBRDF Estimation from Single Image under Learned Planar Lighting. In *ACM SIGGRAPH 2023 Conference Proceedings*. 1–11.

Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. 2018. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 586–595.

Xiuming Zhang, Pratul P Srinivasan, Boyang Deng, Paul Debevec, William T Freeman, and Jonathan T Barron. 2021b. Nerfactor: Neural factorization of shape and reflectance under an unknown illumination. *ACM Transactions on Graphics (ToG)* 40, 6 (2021), 1–18.

Yuanqing Zhang, Jiaming Sun, Xingyi He, Huan Fu, Rongfei Jia, and Xiaowei Zhou. 2022b. Modeling indirect illumination for inverse rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 18643–18652.

Zhenglong Zhou, Zhe Wu, and Ping Tan. 2013. Multi-view photometric stereo with spatially varying isotropic materials. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 1482–1489.

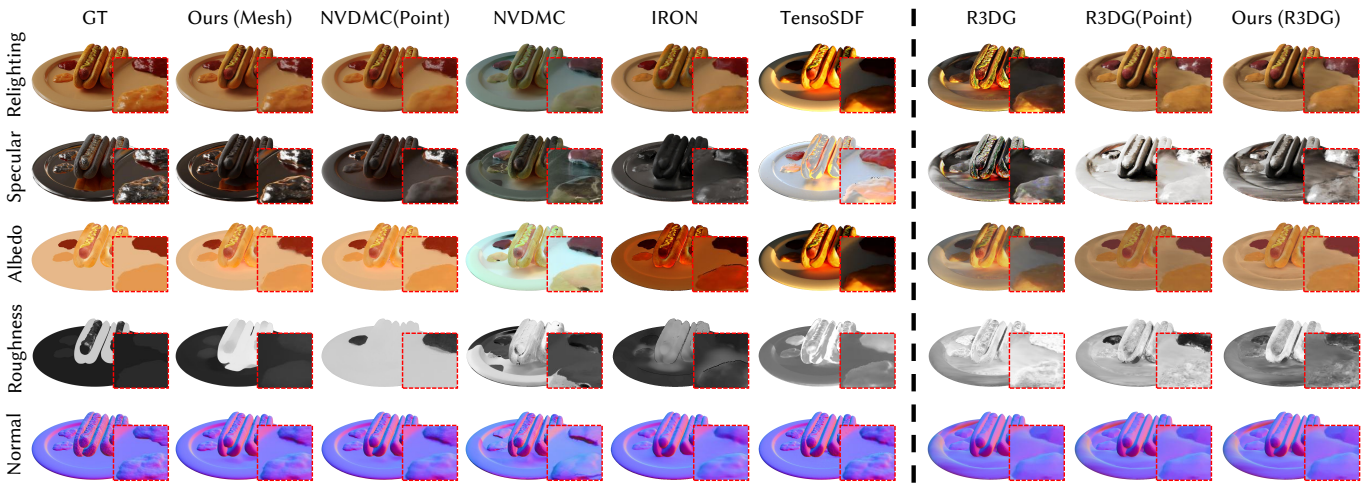

Fig. 7. Comparison of generalization to novel natural environment lighting. Note that our pipeline matches ground truth better for specular highlight regions. We increase the brightness of specular component for better visualization. To visualize world-space normal, we linearly scale it from [-1,1] to [0,1]. We render TensoSDF, NVDiffrecMC-based [Hasselgren et al. 2022] methods and ours using Blender [Community 2018]. Since the material model in IRON is referenced from Mitsuba [Jakob 2010], we render its reconstruction under environment illumination using Mitsuba [Jakob 2010]. R3DG [Gao et al. 2023] leverages 3D Gaussian with material while DPIR [Chung et al. 2024] employs point as geometry primitive and basis BRDF [Lawrence et al. 2006] attached. Both of them are not directly renderable by mesh-based rendering engine. Therefore, we employ their built-in pipeline to render test data. Unfortunately, DPIR has not open-sourced their relighting code. Hence we do not include their relighting results.

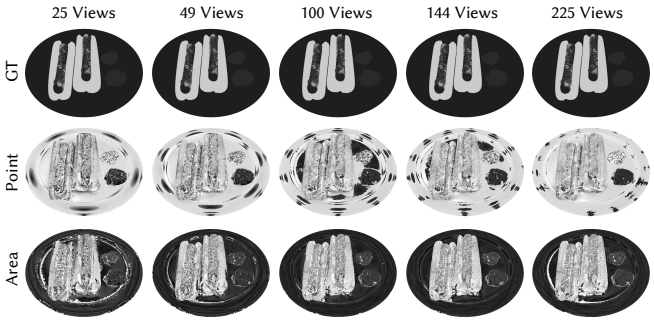

Fig. 8. Ablations on number of views. We show the reconstructed roughness. Note that point light fails to reconstruct roughness for regions where there is no specular hint.

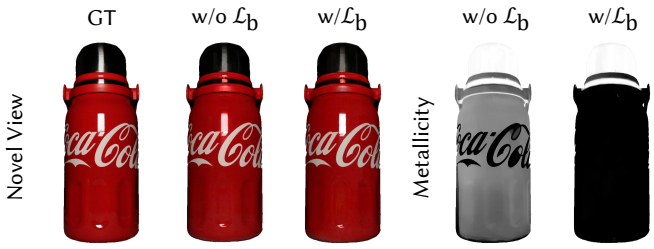

Fig. 9. Ablations on binary loss.

Fig. 10. Ablations on overexposure loss. For better visualization, we increase the contrast in bottle.

Fig. 11. Our reconstruction results on real captured objects. Zoom in to see the details.