# Efficient Object Reconstruction with Differentiable Area Light Shading


YAOAN GAO, State Key Laboratory of CAD&CG, Zhejiang University, China
JIAMIN XU, Hangzhou Dianzi University, China
JAMES TOMPKIN, Brown University, USA
QI WANG, State Key Lab of CAD and CG, Zhejiang University, China
ZHENG DONG, Zhejiang Sci-Tech University, China and State Key Laboratory of CAD&CG, Zhejiang University, China
HUJUN BAO, State Key Lab of CAD and CG, Zhejiang University, China
YUJUN SHEN, Ant Group, China
HUAMIN WANG, Style3D Research, USA and The Ohio State University, USA
CHANGQING ZOU, State Key Laboratory of CAD&CG, Zhejiang University, China and Zhejiang Lab, China
WEIWEI XU*, State Key Laboratory of CAD&CG, Zhejiang University, China


Fig. 1. We investigate active area lighting in inverse rendering to capture the BRDF material and geometry of real-world objects (*left*). Area lighting can be efficient as it samples a broader range of BRDF angles, reducing the number of photos required thanks to better material roughness estimation. For rendering, we compare MC methods to differentiable linearly transformed cosines (LTC). While both can produce high-fidelity relighting (*right*), LTC can be faster.


In 3D object reconstruction from photographs, estimating material properties is challenging. We propose an inverse rendering method that uses active area lighting: as this provides a wider range of lighting angles per photo than point lighting, material reconstruction can be more accurate for the same number of photos. We compare area light shading with point lighting. With either mesh or 3D Gaussian splatting pipelines, area lighting can improve BRDF reconstruction and leads to +3 dB relighting PSNR over point lights, or need only 1/5 of the input photos for the same quality. We also compare area light shading with Monte Carlo ray tracing and with differential linearly transformed cosines (LTC) plus shadow visibility weighting. LTC can be faster, improving optimization times by 25%. In SOTA method-level comparisons, our approach improves material reconstruction, particularly for material roughness, leading to superior relighting quality.



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

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

**ACM Reference Format:**
Yaoan Gao, Jiamin Xu, James Tompkin, Qi Wang, Zheng Dong, Hujun Bao, Yujun Shen, Huamin Wang, Changqing Zou, and Weiwei Xu. 2025. Efficient Object Reconstruction with Differentiable Area Light Shading. In *SIGGRAPH Asia 2025 Conference Papers (SA Conference Papers '25), December 15–18, 2025, Hong Kong, Hong Kong*. ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3757377.3763865



*Corresponding author

Authors' Contact Information: Yaoan Gao, State Key Laboratory of CAD&CG, Zhejiang University, Hangzhou, China, yaoangao@zju.edu.cn; Jiamin Xu, Hangzhou Dianzi University, Hangzhou, China, superxjm@yeah.net; James Tompkin, Brown University, Providence, USA, james_tompkin@brown.edu; Qi Wang, State Key Lab of CAD and CG, Zhejiang University, Hangzhou, China, wqnina1995@gmail.com; Zheng Dong, Zhejiang Sci-Tech University, Hangzhou, China and State Key Laboratory of CAD&CG, Zhejiang University, Hangzhou, China, zhengdong@zju.edu.cn; Hujun Bao, State Key Lab of CAD and CG, Zhejiang University, Hangzhou, China, bao@cad.zju.edu.cn; Yujun Shen, Ant Group, Hangzhou, China, shenyujun0302@gmail.com; Huamin Wang, Style3D Research, Columbus, USA and The Ohio State University, Columbus, USA, wanghmin@gmail.com; Changqing Zou, State Key Laboratory of CAD&CG, Zhejiang University, Hangzhou, China and Zhejiang Lab, Hangzhou, China, aaronzou1125@gmail.com; Weiwei Xu, State Key Laboratory of CAD&CG, Zhejiang University, Hangzhou, China, xww@cad.zju.edu.cn.




*SA Conference Papers '25, Hong Kong, Hong Kong*



ACM ISBN 979-8-4007-2137-3/25/12
https://doi.org/10.1145/3757377.3763865

## 1 Introduction

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 underconstrained. Ambiguities often arise between material and lighting, where color variations caused by lighting can be mistakenly attributed to material albedo or roughness. 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 illuminate one angle per surface point, and previous methods have used (nearly) co-located point lights [Zhang et al. 2022a], separated point lights [Gao 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]. However, 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. Area lights illuminate multiple angles per surface point, giving broader coverage of the 4-dimensional BRDF function space in a single capture than point lights. For instance, 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. This potentially allows for higher quality material reconstructions or more efficient capture with fewer input photos for the same quality.

For area light inverse rendering, we also require differentiable shading, and so a second kind of efficiency—computational—should also be considered. Shading from area lights is often rendered using Monte Carlo (MC) integration with importance sampling. But, in the active lighting setting, most methods ignore indirect lighting from the object itself for the sake of efficiency. Given this relaxation, the shading of a surface point under an area light can also be approximated using Linearly Transformed Cosines (LTC) [Heitz et al. 2016], which has potential to be fast and noise free given its closed-form nature. No existing work has investigated differentiable LTC for multi-view object reconstruction. Our differentiable LTC is implemented in CUDA and accommodates an optimization stability issue caused by nearly adjacent polygonal light vertices when clipping to the surface tangent plane. Since LTC can only represent shading and not shadowing, we add shadowing back using limited ray tracing to compute per-view area-guided visibility maps. As these use a mostly-accurate initial geometry, they only rarely require updating by forward rendering, which is computationally efficient.

We integrate this approach into 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 multi-view images and reconstruct an initial coarse geometry using off-the-shelf photogrammetry tools. Next, we refine the geometry and optimize the BRDF materials to match the input photographs.

Area lighting shows improved SVBRDF reconstruction quality in both mesh and Gaussian pipelines, with particular improvements in material roughness estimation. For an equivalent quality of relighting for rendering captured objects in new scenes, the area light with LTC approach reduces the number of images needed to 1/5 of those needed with a point light approach. In comparison to MC, the closed-form integration of LTC with our area-guided visibility approach is 25% faster than MC at 16 SPP for comparable quality, and is 5× faster than other neural inverse rendering methods.

*Assumptions and limitations.* Our work does not model shading due to interreflection, nor effects from transmissive materials like glass or highly specular mirror. Our final quality is dependent upon the initial geometry which we assume to be mostly accurate; 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, and NeRF [Mildenhall et al. 2021] has spurred many neural inverse rendering methods. Some methods constrain the lighting of input images to an environment map [Srinivasan et al. 2021; Zhang et al. 2021b], while others let input images varying in 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] uses a signed distance field (SDF) for scene geometry and uses sphere tracing for ray-geometry intersections. Other SDF-based methods [Liu et al. 2023; Wu et al. 2023; Zhang et al. 2023b, 2022b] have investigated global illumination models to improve the accuracy of material-lighting decoupling.

The recent success of 3DGS has drawn attention from the field of inverse rendering [Wu et al. 2024], with deferred shading of rasterization applied for realistic rendering and relighting. GS-IR [Liang et al. 2024] proposed a depth-based regularizer for normal estimation and a cube map-based baking strategy to model occlusion and indirect illumination. R3DG [Gao et al. 2023] uses a similar normal estimation strategy, but associates a set of SH coefficients with each Gaussian to represent indirect illumination, then uses physics-based rendering to compute and blend the radiance of each Gaussian.

Different from these methods, NVDiffrec [Munkberg et al. 2022] and NVDiffrecMC [Hasselgren et al. 2022] directly optimize a mesh using a differentiable marching cube method DMTet [Shen et al.

2021]. Meshes are easier to accelerate with GPUs for hardware-accelerated rasterization [Laine et al. 2020] and ray tracing, and fit into graphics production workflows. To address noise and aliasing in differentiable ray tracing, Zhou et al. [2021] introduce a rendering technique called vectorization, which analytically computes 2D point-to-region integrals. In contrast, our work investigates the practical trade-offs of using diffLTC for object reconstruction under active area lighting. Moreover, our visibility map estimation is more efficient than theirs, which, while accurate, incurs high computational cost due to convex region splitting and merging.

*Inverse rendering with known active lights.* This reduces the dimensionality of the optimization space and so helps to alleviate ambiguity. Here, we focus on neural-based methods and will not introduce traditional methods in detail [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].

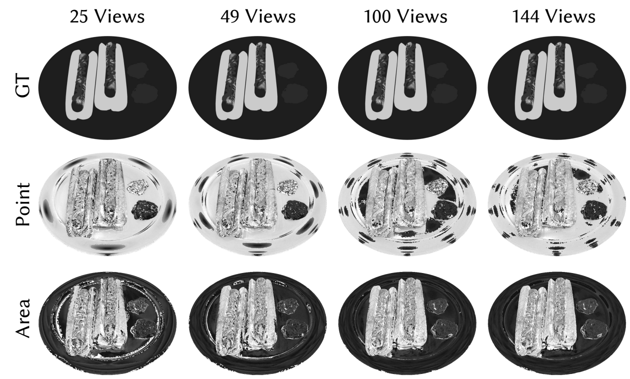
The point light assumption restricts the lighting distribution function to a Dirac delta function, allowing the integral of the rendering process to be computed with a single sample. This significantly simplifies the rendering, making it one of the most common active light sources. Bi et al. [2020b] optimize the geometry and SVBRDF of the object by training a depth estimation network and a reflectance estimation network. Subsequently, methods combine point lighting with different scene representations, 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], and 3DGS-based [Bi et al. 2024b]. 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.

Beyond point lights, other works have used 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. Zhang et al. [2023a] predict the SVBRDF of a planar sample from a single image captured under an RGB LCD area light and a camera without careful calibration. This method is only applicable to planar objects and requires a large amount of training data, whereas we consider reconstructing the material properties of an object from multi-view data and using direct inverse rendering rather than learning.

## 3 Efficient Area Light Object Reconstruction

We wish to reconstruct 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 $L$ and that we know via calibration its relative pose with respect to the camera [Whelan et al. 2018]. We make the common assumption for this problem that we are given an initial geometry that needs only minor refinement.

*Overall pipeline.* We follow the approach of NVDiffrecMC [Hasselgren et al. 2022], which is a mesh-based pipeline. We modify it to incorporate active point and area lighting via ray sampling and Monte Carlo integration, and to incorporate our differentiable LTC. Given the initial geometry, first we differentiably rasterize the mesh with textures into a G-buffer. Then, we shade each pixel from our area light—we will consider both MC and LTC assuming no interreflection. Finally, we use the rendering loss to end-to-end optimize the albedo, roughness, metallicity, and geometry (vertex positions and a normal map with normal offsets) iteratively via gradient descent. We defer the 3DGS-based inverse rendering pipeline for later (Section 4.2); the principles are the same.




| # Views | 25 | 49 | 100 | 144 |
|---|---|---|---|---|
| PSNR Point | 25.90 | 26.39 | 26.74 | 26.89 |
| PSNR Area | 27.86 | 29.08 | 29.04 | 29.17 |
| $r$ MAE Point | 0.1391 | 0.1356 | 0.1168 | 0.1245 |
| $r$ MAE Area | 0.0558 | 0.0436 | 0.0385 | 0.0354 |

Fig. 2. **Area lights give more accurate material roughness than point lights and need fewer views.** Reconstructed roughness $r$. Point lighting fails to reconstruct roughness where there is no specular hint. The table reports quantitative relighting PSNR and $r$ MAE on the 'Hotdog' synthetic object from uniform views. *Shaded with LTC; results similar for MC.*

*Material representation.* We follow previous work in inverse rendering for objects 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 $\rho$ is the BRDF, $\mathbf{x}$ represents the shading point, $\omega_v$ and $\omega_l$ 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 Area Light vs. Point Light for Efficient Object Capture

First, let us validate the benefit of using area lights over point lights in inverse rendering for efficient capture (Fig. 2). To have ground truth, we use the 'Hotdog' synthetic object [Mildenhall et al. 2021].

We use the same initial geometry, and vary the input with 25, 49, 100, 144, 225, and 1024 captures uniformly sampled over the hemisphere. We use relighting as our 'end goal' metric.

Figure 2 shows that as the number of views increases from 25 to 144, both point light and area light relighting results improve. However, even with just 25 views, an area light-based inverse rendering achieves significantly better relighting quality compared to point lighting with even 144 views. We can see that the reason for this is significantly improved material roughness $r$ reconstruction, which is overestimated from point lights and so causes specular reflections to disappear. In this particular scene, the plate is smooth and shiny. For point lighting, we see incorrect roughness on the plate except for regions with a specular hint; this hint appears only when the view angle aligns with the specular lobe. Thus, the key improvement from area lights is on specular objects, for which sampling efficiency matters. Point lights have fewer highlight hints due to their lower angular sampling efficiency.

### 3.2 Area Light Shading via MC and LTC

We assume the reader is familiar with Monte Carlo integration and is somewhat familiar with linearly transformed cosines (LTC) [Heitz et al. 2016]; for a review of LTC, please see our supplementary material. In physically-based rendering, as we assume no interreflection, 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:

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

where $\mathbf{c}_p$ is the color of pixel $p$, $\mathbf{x}$ is the intersection point along the view direction $\omega_v$, 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 samples many rays to achieve an accurate result. LTC efficiently approximates this integral by exploiting the fact that integrals of cosine distributions have closed-form solutions for a polygonal light, and this distribution can be linearly transformed via $\mathbf{M}$ to approximate isotropic BRDFs. This results in a fast integration method without relying on sampling and Monte Carlo integration, because the transformed integration $E(\mathcal{P}')$ admits a closed-form analytical solution [Heitz 2017; Lambert 1760]. It equals the sum of several (signed) areas corresponding to the edges along the light source boundary:

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

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.

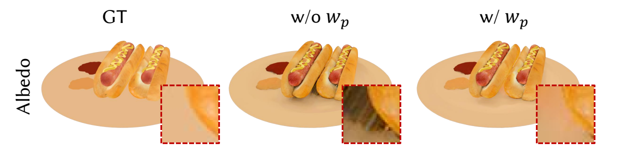



Fig. 3. **Visibility map $w_p$.** Without precomputed visibility, optimization bakes contact shadows into the albedo, which hurts relighting quality.

The rendered color is given by $\mathbf{c}_p^{ltc} = \mathbf{c}_p^s + \mathbf{c}_p^d$. The diffuse component can be directly obtained by integrating over the cosine distribution without linear transformation:

$$\mathbf{c}_p^d = \frac{(1-m)a}{\pi} E(\mathcal{P}) \, . \quad (4)$$

The specular component of a shaded point $\mathbf{p}$ requires the fitted linear transformations $\mathbf{M}^{-1}(r, \beta)$ for a BRDF to compute the closed-form integration in Eq. 3:

$$\mathbf{c}_p^s = F(r, \beta) E(\mathcal{P}'), \quad (5)$$

$$\mathcal{P}' = \left\{ \mathbf{M}^{-1}(r, \beta)\mathbf{p}_i \mid \mathbf{p}_i \in \mathcal{P} \right\}, \quad (6)$$

$$F(r, \beta) = F_0 f_1(r, \beta) + (1 - F_0) f_2(r, \beta), \quad (7)$$

where $r$ is the material roughness, $\beta$ is the dot product of the view direction $\omega_v$ and the surface normal $\mathbf{n}$, $F_0 == 0.04(1-m) + m\mathbf{a}$ is the basic reflectance, and $f_1(r, \beta), f_2(r, \beta)$ compensates for energy loss caused by the shadowing term and the omission of the Fresnel term (including albedo and metallicity) in the fitting of $\mathbf{M}$. $\mathbf{M}, f_1, f_2$ can be precomputed and stored as 2D look-up tables (LUTs).

*Shadowing via area-guided visibility.* The LTC-rendered color $\mathbf{c}_p^{ltc}$ ignores occlusion caused by geometry, so we use 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 calculated using LTC $\mathbf{c}_p^{ltc}$, multiplied by a visibility map $w_p$:

$$\begin{aligned} \hat{\mathbf{c}}_p(\mathbf{x}, \omega_v) &= L \int_{\mathcal{P}} V(\mathbf{x}, \omega_l) \rho(\mathbf{x}, \omega_v, \omega_l) \cos\theta_l \mathrm{d}\omega_l \quad (8) \\ &= w_p(\mathbf{x}, \omega_v) \mathbf{c}_p(\mathbf{x}, \omega_v) \approx w_p(\mathbf{x}, \omega_v) \mathbf{c}_p^{ltc}(\mathbf{x}, \omega_v), \quad (9) \end{aligned}$$

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

where $V(\mathbf{x}, \omega_l)$ is the visibility of incident light at the shading point $\mathbf{x}$. The visibility map reduces the weight of shadowed areas, which largely eliminates issues like baked shadows in albedo (Fig. 3).

*Cost analysis: Sampling and denoising.* We review the cost of rendering components within a forward and backward pass of MC and LTC by adapting NVDiffrecMC with our implementations (Tab. 1). Recall that we ignore global illumination and so use one bounce ray tracing. Recall also that NVDiffrecMC is not a pure MC ray tracer: the camera ray hit point is determined by a rasterizer (NVDiffrast [Laine et al. 2020]) and only the shading—the inner integral in the rendering equation—is computed using MC integration. In complex lighting environments, such as when using a passive environment map as NVDiffrec originally does, we sample and denoise for efficiency. NVDiffrecMC uses a bilateral denoiser that dominates forward and backpropagation time (Tab. 1). But, in a dark room with known area light position, no denoiser is required: Along with

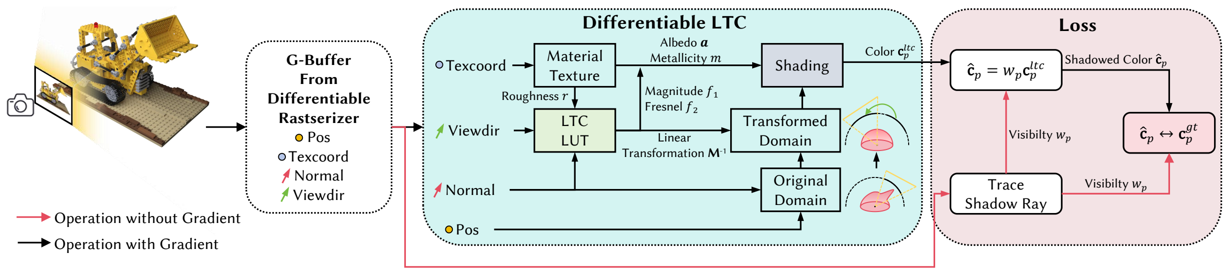



Fig. 4. **Object reconstruction with differentiable LTC shading.** We optimize object geometry and BRDF materials using a set of images captured in a dark room and an initial geometry, where the geometry is rasterized into a G-Buffer for efficiency following NVDiffrecMC [Hasselgren et al. 2022].

Table 1. Time (ms) for one iteration with a batch of 8 800×800 images from 'Hotdog' object in NVDiffrecMC, including with our LTC shading modifications.

| Method | Forward Rendering | | | | | | | Back Propagation | | Total |
|---|---|---|---|---|---|---|---|---|---|---|
| | Build BVH | Evaluate normal | Rasterize | Shade | Denoise | Composite | Total | Denoise | Total | |
| LTC | 0.0 | 12 | 1.5 | 2.1 | 0 | 27 | 50 | 0 | 77 | 173 |
| MC 1 SPP | 2.4 | 12 | 1.5 | 2.4 | 100 | 27 | 150 | 117 | 197 | 390 |
| MC 16 SPP | 2.4 | 12 | 1.5 | 26.0 | 100 | 27 | 175 | 117 | 302 | 514 |
| MC 1,024 SPP | 2.4 | 12 | 1.5 | 1685.0 | 100 | 27 | 2143 | 117 | 7163 | 9326 |

speeding execution, removing the denoiser *improves* quality at 16 and 1,024 SPP. There are two reasons: First, next-event estimation is more efficient under area lighting than under environment lighting, and the noise is substantially suppressed by importance sampling. Second, a denoiser inevitably smooths details by accident. As such, we do not use a denoiser for NVDiffrecMC with area lighting via ray sampling and MC integration. LTC shading is noise-free via its closed-form integration, reducing total backpropagation time from 295 ms to 77 ms with our CUDA implementation.

*Cost analysis: Visibility maps.* Soft shadows from area lights require accurate visibility estimation. Computing visibility maps by ray tracing $w_\mathrm{p}$ for each pixel in each of 200 input views using 1,024 rays $\omega_l$ on the spherical domain $\mathcal{P}$ takes 15 seconds. Recomputing this at each optimization iteration for the images in the batch is prohibitively slow. Consider that, in the active lighting setting, the shadow/visibility maps $w_p$ depend upon the initial geometry and the camera's view. As the initial geometry is assumed to be largely accurate, and as LTC separates visibility map computation from shading, then efficiency can be improved if we simply do not update the maps often. The trade-off is a slight quality loss. If we ray trace visibility at the beginning of optimization and update it every 1,000 iterations then, on the 'Hotdog' object (random 200 views), reconstruction takes a quarter of the time for -0.5 db in relighting quality (32.56 db to 31.95 db), compared to computing the visibility map at every iteration.

Since our precomputed visibility maps depends on initial geometry, we evaluate the effect of initial geometry error and update frequency. We added perturbations to the initial geometry using the displace modifier in Blender and varied this interval. 1) Perturbing the initial geometry degrades final quality as the added perturbation was not fully ameliorated through our optimization. 2) Visibility update frequently has the expected slight impact on quality.

We could also use the same area-guided visibility trick for MC. In this case, we only intersect light rays with the light itself, and instead weight the output with the pre-computed visibility map. This obviates the need for frequent BVH updates and shadow tests, which would incur per-iteration costs of 2.4 ms and 7 ms, respectively, at 16 SPP. As shown in Tab. 1, the impact on overall runtime is negligible. However, this method results in a slight reduction in relighting quality, with a drop of 0.8 dB (from 31.53 dB to 30.69 dB). The use of a visibility map introduces bias no matter how it is computed as it is an approximation. In our experiments, computing per-ray visibility at every iteration with 16 SPP and updating visibility maps every 1,000 iterations at 1,024 SPP results in comparable runtimes. Consequently, applying the visibility trick to 16-SPP MC does not lead to a meaningful speedup.

## 4 Optimization Details

*Differentiable LTC (Fig. 4).* 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 $\beta$ through numerical gradients using forward differences. These derivatives can be used in the chain rule for the derivative computation in Eq. 5 to obtain the derivative of the specular components $\mathbf{c}_p^s$ with respect to $r$ and $\beta$. Since the vertex positions are used to calculate the $\beta$, the derivatives of $\mathbf{c}_p^s$ can be back-propagated back to the vertex positions through $\beta$. 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.

*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. 5. 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

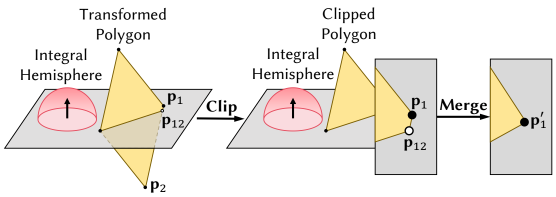



Fig. 5. **Light source clipping.** In LTC, we clip the transformed polygon to the surface tangent plane before projecting it onto the integral sphere. When a new vertex $P_{12}$ generated from clipping is too close to another vertex $P_1$, it causes unstable optimization. We merge such vertices to mitigate this.

Table 2. **Ablation of initial geometry and visilibily map update frequency**. We show relighting PSNR on 'hotdog' object.

| Update every $n$ iteration | Initial geometry without perturbation | Initial geometry with 5% perturbation | Runtime |
|---|---|---|---|
| 1 | 32.56 | 31.55 | 40mins |
| 100 | 32.33 | 31.15 | 18mins |
| 1000 | 31.95 | 30.56 | 11mins |
| 3000 | 31.80 | 30.27 | 10mins |

forward rendering and back-propagation. This causes only a minor effect because the contribution from the small edge between close points to the overall rendering is small.

### 4.1 Mesh-based optimization pipeline

We use a triangular 3D mesh with textures to represent object geometry, albedo, roughness, and metallicity maps. We optimize the vertex positions, normal map as offset from the geometry normal, material albedo $\mathbf{a}$, roughness $r$, metallicity $m$, and radiance $L$ of the area light. For initialization, we set albedo to 1.0, metallicity to 0.0, and we randomly initialize roughness and the light radiance.

*Losses.* We define a loss $\mathcal{L}$ as the weighted combination of the image loss and a set of regularizer terms:

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

where $\lambda$ 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:

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

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 the Disney Principled BRDF, we add a binary loss encouraging metallicity to be either 0 or 1 for pure surfaces such as pure metal, pure stone, or pure plastic:

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

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

Next, we apply smoothness priors 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 $L_{\text{nc}}$ to enforce normal consistency across randomly sampled adjacent triangles.

### 4.2 3DGS-based optimization pipeline

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) pipeline [Gao et al. 2023]. Specifically, we shade each Gaussian under area lighting with LTC and derive the pixel color by splatting the Gaussian points onto the screen.

## 5 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) from TensoIR [Jin et al. 2023] and five scenes (coffee, helmet, musclecar, teapot and toaster) from Shiny Blender Dataset [Verbin et al. 2022] to demonstrate the benefit of area lights on specular objects. We use Blender's Cycles renderer 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.

Our real captured data consists of eight objects with varying materials (Fig.1 and Fig.8). We capture with 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].

*Geometry.* Like our comparison methods, our pipeline relies on initial geometry. Neural implicit methods like NeuS [Wang et al. 2021] and TensoSDF [Li et al. 2024] can produce satisfactory geometry with an active lighting setup, so we use the geometry reconstruction stage of TensoSDF [Li et al. 2024] for initial geometry. For NVDiffrecMC [Hasselgren et al. 2022], we replace its original DMTet-based geometry initialization [Hasselgren et al. 2022] with the better TensoSDF geometry. For the SOTA method-level comparisons, as DPIR [Chung et al. 2024] uses points as primitives, we use its original point cloud initialization based on the visual hull. IRON [Zhang et al. 2022a] bases its neural geometry on NeuS [Wang et al. 2021]. After initial 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

Table 3. **Quantitative comparisons show area light improves NVS and relighting on synthetic data.** '+ D' is with MC denoiser, showing a drop in reconstruction quality. We scale each RGB channel of albedo, novel view synthesis, and relighting results by an optimal global scalar, as NVDiffrecMC [Hasselgren et al. 2022]. We evaluate roughness on 5 / 9 objects (hotdog, armadillo, ficus, musclecar and teapot), which have well-defined roughness in the principled BSDF. The other 4 have materials that are not parameterized with roughness, such as mixed BSDF. We evaluate normal in world space by rasterizing it into image space for each test view and then normalizing it. For runtime, we use a single RTX3090 GPU and do not include the time spent on geometry initialization. Since the material model in IRON is referenced from Mitsuba [Jakob et al. 2022], we render its reconstruction using that. 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. DPIR does not parameterize roughness. DPIR has no open-source relighting code. As the points in DPIR are not only at the surface, but also inside the object, we do not report its chamfer distance.

| Method | Light model | Novel View Synthesis | | | Relighting | | | Albedo | | | Roughness | Normal | Chamfer Dist | Runtime |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | PSNR ↑ | SSIM ↑ | LPIPS ↓ | PSNR ↑ | SSIM ↑ | LPIPS ↓ | PSNR ↑ | SSIM ↑ | LPIPS ↓ | MAE ↓ | MAE ↓ | mm ↓ | |
| NVDiffrecMC [Hasselgren et al. 2022] | Static env. map | 29.24 | 0.9348 | 0.09069 | 26.52 | 0.9204 | 0.09775 | 27.22 | 0.9417 | 0.06774 | 0.05543 | 0.02552 | 0.04623 | 65mins |
| | Active point | 29.48 | 0.9565 | 0.04245 | 26.61 | 0.9228 | 0.07874 | 27.90 | 0.9226 | 0.07059 | 0.07170 | 0.01673 | 0.04586 | 12mins |
| | Active area (16 SPP+D) | 29.05 | 0.9565 | 0.04225 | 27.56 | 0.9288 | 0.07053 | 28.32 | 0.9236 | 0.07362 | 0.02588 | 0.01777 | 0.04589 | 34mins |
| NVDiffrecMC (Our modification) | Active area (16 SPP) | 29.19 | 0.9565 | 0.04027 | 27.77 | 0.9350 | 0.06528 | 28.35 | 0.9201 | 0.07485 | 0.03929 | 0.01755 | 0.04587 | 16mins |
| | Active area (1024 SPP) | 29.11 | 0.9571 | 0.03890 | 27.97 | 0.9334 | 0.06186 | 28.88 | 0.9380 | 0.06398 | 0.02461 | 0.01708 | 0.04587 | 350mins |
| | Active area (LTC) | 29.37 | 0.9568 | 0.03924 | 28.29 | 0.9373 | 0.05791 | 29.10 | 0.9421 | 0.06229 | 0.02477 | 0.01704 | 0.04589 | 12mins |
| TensoSDF [Li et al. 2024] | Static field MLP | 29.68 | 0.9538 | 0.05242 | 25.66 | 0.9277 | 0.07526 | 25.28 | 0.9245 | 0.08663 | 0.03524 | 0.01833 | 0.04732 | 396mins |
| IRON [Zhang et al. 2022a] | Active point | 22.59 | 0.9188 | 0.09783 | 23.23 | 0.8937 | 0.1053 | 24.24 | 0.9000 | 0.09728 | 0.05333 | 0.02935 | 0.07609 | 571mins |
| DPIR [Chung et al. 2024] | Active point | 25.16 | 0.9386 | 0.08627 | - | - | - | 27.50 | 0.9214 | 0.08945 | - | 0.06866 | - | 223mins |

Laplacian smoothing [Vollmer et al. 1999]. Next, we generate a $1024 \times 1024$ texture map for both the material and normal maps. Objects with highly metallic areas (Bust, Luckycat) or metallic *and* low reflectance areas (Bottle cap) are known to be 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.

The quality of the initial geometry is shown in Table 3 as the TensoSDF row. The refined geometry after optimization produces a slight improvement only; this is expected as the initial geometry is already mostly accurate. As such, we will not discuss it further.

*Training details.* We jointly optimize geometry, material, and lighting. We initialize white albedo, random roughness, zero metallicity and zero normal map perturbation. We use Adam [Kingma 2014] to optimize in PyTorch [Paszke et al. 2019] with peak learning rates of $1 \times 10^{-6}$ for vertex positions, 0.01 for albedo, roughness, metallicity, and normal maps, and 0.03 for area light radiance. We use 100 warm-up iterations, linearly increasing the learning rate to its peak. After that, we exponentially decay the learning rate to 10% of the peak value. Training consists of 3 k iterations, with a batch size of 8 images, and uses an RTX 3090 GPU. We ignore gradients from the visibility maps: in our multiview setting (100+ views), we found no stability issues caused by this (unlike in a single view setting). For the 3DGS-based pipeline, we use the same training configuration as in R3DG [Gao et al. 2023].

*Metrics.* For quantitative comparisons on the BRDF estimation and relighting results (Tab. 3), 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], using masks to assess only the object region. 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. We report runtime of all methods *after* geometry initialization.

## 5.1 Comparisons

We use two kinds of comparisons. The first is to methods with the same loss built upon the same codebase of NVDiffrecMC [Hasselgren et al. 2022]. These are above the line in Table 3, and directly show the impact of area lights and LTC. The second is to broader SOTA method-level approaches, shown below the line. These methods use different codebases and use the losses presented by the original authors. Even using the same loss in these codebases would not be fair as other implementation differences will impact the comparisons.

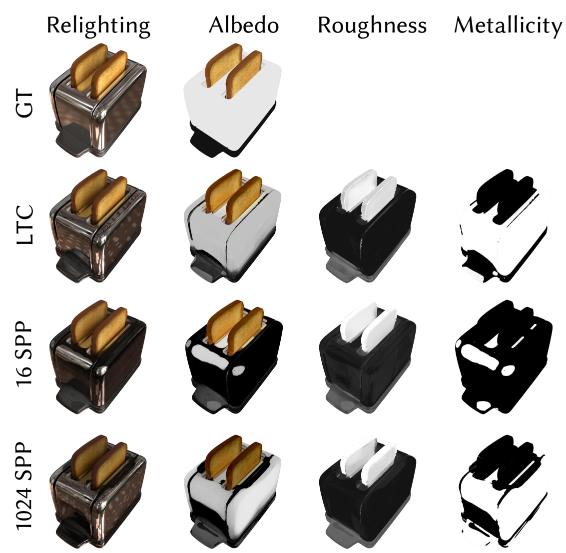



Fig. 6. **Ray tracing vs. LTC**. Area lighting with ray sampling and Monte Carlo integration at low SPP suffers from instable gradient and fails at highly specular objects.

For NVDiffrecMC-based comparisons, as expected, both MC and LTC area lights produce plausible materials. For MC with 16 SPP, the relighting results are 0.5 db less than LTC quality and LTC is 25% faster on average. MC at 16 SPP suffers from noise in both rendering and gradients, which can occasionally cause failures (Fig. 6). As SPP increases to 1,024, MC approaches LTC quality. Although memory consumption in MC backpropagation can be adequately mitigated by path replay [Vicini et al. 2021], training time increases drastically:

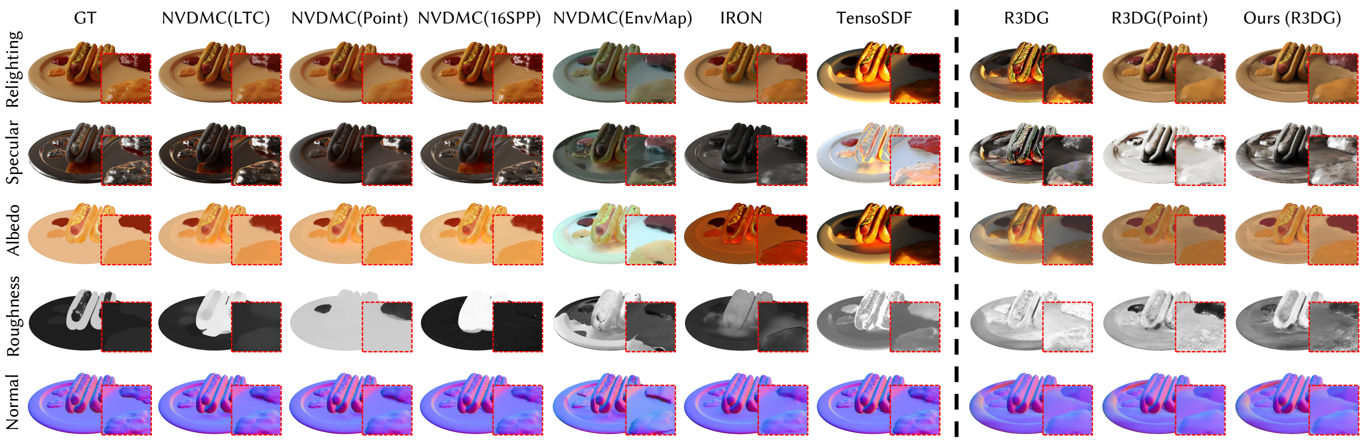



Fig. 7. **Relighting is more accurate with area lights, especially in specular regions.** We scale each RGB channel of relighting and albedo results by an optimal global scalar, as NVDiffrecMC [Hasselgren et al. 2022]. We increase the brightness of specular component for better visualization. To visualize world-space normal, we linearly scale it from [-1,1] to [0,1]. NVDMC (Origin) refers to original NVDiffrecMC that uses static environmental lighting. NVDMC (Point) refers to NVDiffrecMC modified for point lighting while NVDMC (16SPP) refers to NVDiffrecMC with area lighting via ray sampling and Monte Carlo integration under 16 SPP. Since the material model in IRON is referenced from Mitsuba [Jakob et al. 2022], we render its reconstruction using that. 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. As they cannot be rendered with Blender, we use their provided code. DPIR has not open-sourced their relighting code.

1,024 SPP is 29× slower than LTC (Tab. 3). Adding the denoiser back to try to speed this up does not reach the same quality, as discussed; e.g., 16 SPP with denoising is worse than 16 SPP without.

As broader method-level comparison of SOTA approaches, we compare to three alternative inverse rendering methods not based on NVDiffrecMC: IRON [Zhang et al. 2022a] and DPIR [Chung et al. 2024], which use active point lighting, and TensoSDF [Li et al. 2024], which assume static environmental lighting. For the active lighting setting, Table 3 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. Our inverse rendering method is also an order of magnitude faster than these neural methods.

*Evaluation on real objects.* We compare results to TensoSDF [Li et al. 2024]. As we might expect in this more challenging setting, performance decreases for all methods than on synthetic objects. For real objects (Fig. 8), we see that TensoSDF fails to properly decompose albedo, roughness, and metallicity; our approach fares better. Additionally, for the albedo, since the color temperature of the environment lighting is not calibrated, it is baked into the albedo in TensoSDF. For example, the albedo of the vase object exhibits cool tones, which are influenced by the environment lighting. Figure 12 shows all eight objects reconstructed by our mesh+LTC approach.

*Evaluation on 3DGS-based pipeline.* We use relightable 3D Gaussian (R3DG) [Gao et al. 2023] to evaluate our differentiable-LTC-based area lighting. We modify the original environment lighting model in R3DG to incorporate point lighting and area lighting. As R3DG does not support metallicity, we do not include comparison on Shiny Blender Dataset [Verbin et al. 2022] with mostly metal objects. We observe that our LTC area light approach better estimates complex materials, yielding better relighting performance (Table 4).

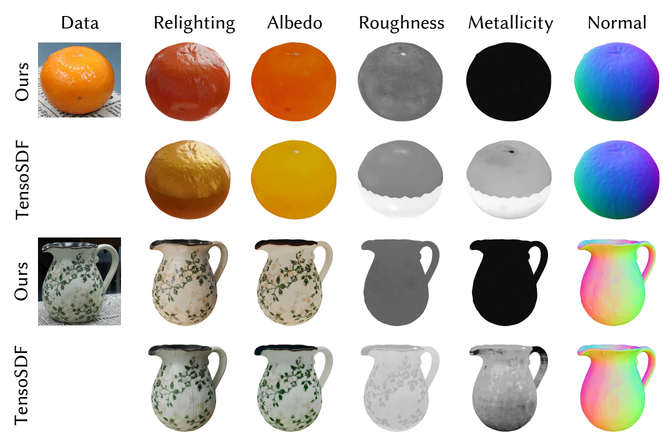



Fig. 8. **Real objects results.** Data column shows the objects under environment lighting. Since we do not capture this environment lighting, we relight the objects with another HDR environment map and show the relighting results. Compared to natural environment capture in TensoSDF, our area light approach more accurately recovers PBR materials.

Table 4. **Evaluation on 3DGS-based pipeline.** We replace the environment lighting in R3DG [Gao et al. 2023] with LTC shaded area lighting.

| 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 5. **Ablation of area light size**. Evaluated in our NVDiffrecMC LTC pipeline. We observe similar results with 16 SPP ray tracing, showing that the quality decrease is due to the area light size rather than LTC.

| Side Length (meters) | Albedo | Normal | Relighting | | |
|---|---|---|---|---|---|
| | PSNR ↑ | MAE ↓ | PSNR ↑ | SSIM ↑ | LPIPS ↓ |
| 1 | 30.28 | 0.02242 | 30.40 | 0.9543 | 0.04767 |
| 2 | 30.42 | 0.02182 | 31.90 | 0.9610 | 0.04028 |
| 4 (ours) | 30.32 | 0.02216 | 31.95 | 0.9617 | 0.04144 |
| 8 | 29.50 | 0.02365 | 29.41 | 0.9513 | 0.05598 |

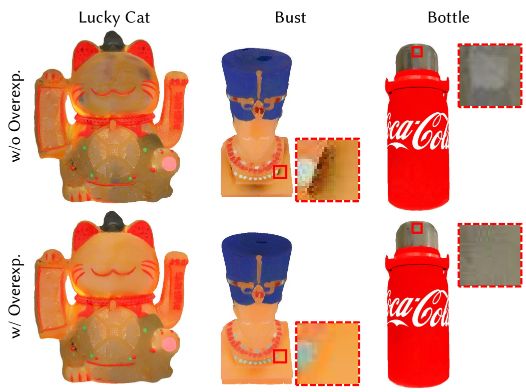



Fig. 9. **Ablations on overexposure loss.** For better visualization, we increase the contrast in bottle.

## 5.2 Ablations

*Area light size.* We examine the impact of area light size using the 'Hotdog' synthetic object. The area light is square. The object diameter is 2 m. The camera is 4 m away. The ratio of area light to object size is similar to our real world examples where the LED panel is 10 cm, the object diameters are 10–35 cm, and the camera is 35–45 cm away. When the area light is too large or too small, quality suffers (Tab. 5). When using a small area light, the issue is similar to point lighting: specular areas appear diffuse under relighting conditions because point lighting may miss highlight hints due to lower angular sampling efficiency. When using a large area light, we lack contrast between regions with and without highlights, decreasing quality. A best practice would choose an area light size and capture distance based on the angular coverage of the BRDF lobe, geometric surface detail, and light brightness (potential overexposure). In practice, we fix the area light size and set its distance to roughly the diagonal length of the object's bounding box.

*Overexposure detection in $\mathcal{L}_{\text{render}}$.* As shown in Fig. 9, 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.

*Metallicity binary loss.* Without the metallicity binary loss, our system sometimes fails to optimize the metallicity correctly. For example, in Fig. 10, 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.

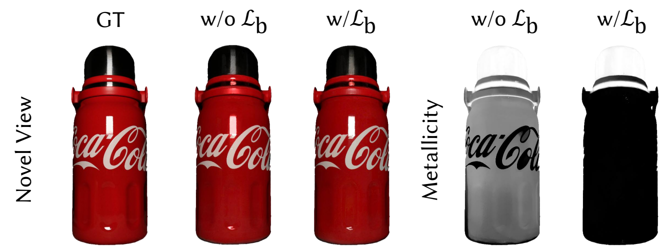



Fig. 10. Ablations on metallicity binary loss.

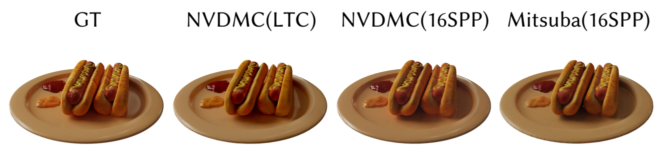



Fig. 11. **Relighting results using a different backbone render pipeline.** We obtain comparable results in NVDiffrecMC [Hasselgren et al. 2022] and Mitsuba [Jakob et al. 2022] pipeline.

*Mitsuba backbone.* We also show a preliminary experiment comparing NVDiffRecMC to Mitsuba as the rendering backbone for ray tracing (Fig. 11). Both approaches produce similar relighting quality. We provide experimental details in the supplemental material.

## 6 Conclusion

For sample efficiency, area lighting can help to improve object reconstruction material quality over point lights, particularly for material roughness, and this is what enables it to reduce the number of views necessary significantly for a similar quality. For computational efficiency, we have introduced a differentiable linearly transformed cosines approach that can reduce optimization time by 25%, with insight into trade-offs in visibility computation of time and quality.

Looking forward, one opportunity for inverse LTC shading is efficient pattern lights, e.g., textured or colored. These have potential to improve normal estimation, but MC sampling here would only increase noise. By partitioning the pattern into polygons, LTC can still compute shading efficiently and with low noise.

## Acknowledgments

We thank the anonymous reviewers for their professional and constructive comments. We also thank Xiuchao Wu for his insightful discussions and Pinxuan Dai for his help in capturing real objects. Weiwei Xu is partially supported by the National Key R&D Program of China under Grant No. 2024YFE0216600, NSFC grant No. 62421003. Jiamin Xu is partially supported by NSFC grant No. 62302134 and ZJNSF grant No. LQ24F020031. James Tompkin is partially supported by the US NSF CNS-2038897 and by CAREER IIS-2144956. This paper is supported by Ant Group and Information Technology Center and State Key Lab of CAD&CG, Zhejiang University.

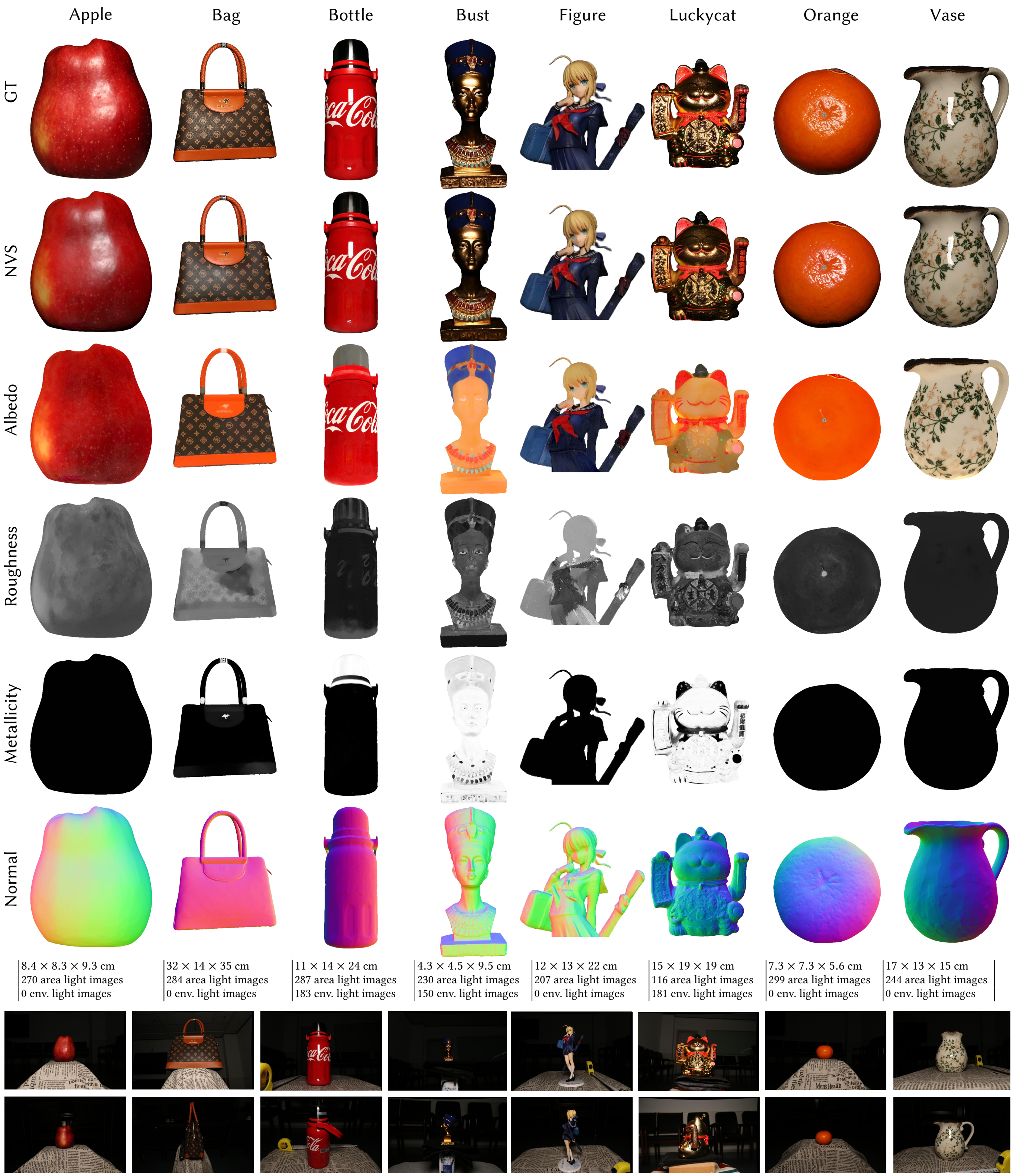



Fig. 12. **Reconstruction results on real captured objects with area lights and LTC**. Example input photographs are at the bottom, along with object size and input photo number. As discussed in the main body text, highly metallic objects are extremely challenging and so have additional photos under environment lighting *just* for initial geometry reconstruction. Please zoom in to see the details.

## 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.

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

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

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.

Dor Verbin, Peter Hedman, Ben Mildenhall, Todd Zickler, Jonathan T Barron, and Pratul P Srinivasan. 2022. Ref-nerf: Structured view-dependent appearance for neural radiance fields. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 5481–5490.

Delio Vicini, Sébastien Speierer, and Wenzel Jakob. 2021. Path replay backpropagation: Differentiating light paths using constant memory and linear time. *ACM Transactions on Graphics (TOG)* 40, 4 (2021), 1–14.

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.

Yang Zhou, Lifan Wu, Ravi Ramamoorthi, and Ling-Qi Yan. 2021. Vectorization for fast, analytic, and differentiable visibility. *ACM Transactions on Graphics (TOG)* 40, 3 (2021), 1–21.

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.