High-Quality and Efficient Inverse Rendering for Geometry, Material, and Illumination Reconstruction

Yishuo Fei
Zhejiang University
China
22321111@zju.edu.cn

Chao Chen
Ningbo Museum
China
87339235@qq.com

Haipeng Liao
NingboTech University
China
lhp@nit.net.cn

Mo Chen
Zhejiang University
China
cmandhxj@zju.edu.cn

Yuhui Yang
Zhejiang University
China
zju_edu@zju.edu.cn

Dongming Lu
Zhejiang University
China
ldm@zju.edu.cn

Abstract

Inverse rendering is a challenging problem due to the inherent ambiguity in recovering 3D geometry, materials, and lighting from multi-view RGB images. Recent methods represent geometry and materials as neural networks, recovering them through an analysis-by-synthesis approach. However, accurately modeling shadows and inter-reflections remains a significant obstacle. In this paper, we propose a two-stage inverse rendering pipeline that effectively reconstructs geometry, materials, and illumination. First, we decompose the neural radiance field into diffuse and specular components and introduce an occlusion network to jointly improve surface reconstruction quality. We then use Monte Carlo-based path tracing to model shadows and inter-reflections. To enhance efficiency and quality, we employ separate hash-encoded MLPs for geometry and material representation. Experiments demonstrate that our method outperforms previous work on synthetic and real datasets, accurately recovering geometry, roughness, and high-quality albedo, while supporting realistic re-rendering and relighting.

***Keywords:** Inverse Rendering, Neural Radiance Decomposition, Monte Carlo Path Tracing, Hash-Encoded MLPs.*

1. Introduction

Inverse rendering, which involves recovering geometry, materials, and illumination from images, has been a long-standing issue in computer vision and graphics. This task has widespread applications in VR, AR, and visual effects production. Inverse rendering technology is crucial for generating high-quality visual content, enhancing the realism of environmental simulations, and improving the performance of machine vision systems.

Recent methods[1, 23, 32, 34, 35] represent geometry and the Spatially Varying Bidirectional Reflectance Distribution Function (SVBRDF) as neural networks, and recover them through an analysis-by-synthesis approach. Modeling shadows and inter-reflections presents a significant challenge. Previous methods have either neglected occlusion and indirect light[1, 2, 20, 32] or merely modeled visibility [34]. The absence of accurate occlusion and indirect lighting modeling results in shadows and indirect light being erroneously baked into the materials. MII [35] models both occlusion and indirect lighting using Spherical Gaussians (SGs), but it struggles to capture high-frequency details, resulting in blurry re-rendering results.

In this paper, we present a two-stage inverse rendering pipeline that effectively reconstructs geometry, material, and illumination. In the first stage, we employ a Signed Distance Function(SDF) to represent surface geometry and learn the object's geometry and neural radiance field from multi-view images. In the self-occluded regions of glossy objects, recovering geometry often results in errors such as depressions and unevenness. To address this, we decompose the neural radiance into diffuse and specular components and introduce an occlusion network to compute occlusion probabilities. In the second stage, we apply Monte Carlo-based path tracing to simulate shadows and inter-reflections. An MLP represents direct illumination, and the learned neural radiance field represents indirect illumination. We optimize the SVBRDF and direct illumination and refine geometry with the rendering equation. To streamline computational complexity, we employ a hash-encoded MLP to represent the SDF. Additionally, we utilize a separate hash-encoded MLP to capture high-frequency local details in the SVBRDF.

We evaluated our method on both synthetic and real datasets. The experiments demonstrate that our results surpass others in both quantitative and qualitative metrics. Our method efficiently recovers accurate geometry, roughness, and high-quality albedo within a shorter timeframe.

2. Background

2.1. Implicit Neural Representation

Implicit neural representations have greatly enhanced the performance of inverse rendering. IGR [6] can compute high-fidelity implicit neural surfaces from point cloud data, and the recovered zero-level set surface is smooth and natural. NeRF [18] encodes scenes into radiance and density fields using a multilayer perceptron, enabling realistic synthesis of new viewpoints and reconstructing scenes with differentiable volume rendering from a limited set of images. TensoRF [3] represents the radiance field as tensors, utilizing tensor decomposition techniques to compactly encode the scene. 3D-GS [11] further models the scene with 3D Gaussian splats, enabling real-time radiance field rendering. Additionally, surface-based methods such as IDR [30] and NeuS [27] utilize Signed Distance Functions (SDFs) to represent geometry, enhancing the quality of viewpoint generation precisely. NeuS2 [28] utilizes multi-resolution hash encoding to parameterize neural surface representation, significantly improving the training speed of NeuS [27]. However, these approaches primarily simulate surface emission and do not adequately address the decomposition of incoming radiance and material properties, thus limiting their effectiveness in scene editing.

2.2. Inverse Rendering with Implicit Neural Representation

Recent developments harness the fully differentiable capabilities of implicit neural representations, increasing the flexibility of capture settings and advancing the precision in modeling and optimizing geometry and materials. Ref-NeRF [25] decomposes a scene into separate geometry and appearance representations, boosting reconstruction performance for smooth objects but failing to accurately separate material properties. PhySG [32] decomposes scenes under arbitrary, unknown illumination, and NeRD [1] and NeuralPIL [2] process images under varying illumination, none of these methods account for occlusion and indirect lighting. NeRFactor [34] considers occlusion and direct lighting but lacks modeling for indirect lighting. MII [35] further models both occlusion and indirect light, encodes this data within MLPs, and uses Spherical Gaussians (SGs) [26] to approximate the rendering equation. However, it falls short of capturing high-frequency scene details. NeILF++ [31] represents the lighting of a static scene using a neural incident light field and an output neural radiance field, effectively disentangling geometry, material, and lighting. However, its training time remains lengthy. Some methods apply Monte Carlo sampling to model occlusion and indirect lighting. NeRV [23] limits its consideration to a single indirect bounce and depends on known environmental lighting. TensoIR [9] delivers superior re-rendering quality by employing light-intensity importance sampling for simulating incident light, albeit at the expense of compromising other components' reconstruction fidelity. TensoSDF [14], integrating a roughness-aware radiance and reflectance field, successfully reconstructs both diffuse and specular scenes, though its performance degrades for intermediate cases.

3. Method

3.1. Overview

Given a collection of posed images of an object under static illumination, our goal is to decompose the geometry, SVBRDF, and illumination. We tackle the inverse rendering problem through an analysis-by-synthesis approach, taking into account indirect illumination and shadows. Figure 1 shows the forward rendering process of our method.

The geometry is represented by a zero-level set of a Signed Distance Function(SDF) as NeuS[27], parameterized by a hash-encoded MLP that maps a 3D location $\mathbf{x}$ to an SDF value s and a geometric feature vector $\mathbf{f}$ (Sec. 3.3). We decompose the neural radiance $L_o(\mathbf{x}, \omega_o)$ at a 3D point $\mathbf{x}$ along the direct ω_o into diffuse radiance $L_d(\mathbf{x})$ and specular radiance $L_s(\mathbf{x}, \omega_o)$ and utilize an occlusion component $\mathcal{O}$ to predict the occlusion probability of specular radiance (Sec. 3.4). The neural radiance results are compared with the captured images to optimize the geometry.

The SVBRDF is modeled by a hash-encoded MLP that predicts albedo $\mathbf{a}$ and roughness r (Sec. 3.5). To render a pixel, we apply sphere tracing on the hash-encoded SDF to locate the intersection point $\mathbf{x}$ where the camera ray meets the geometric surface. We sample incoming light rays using importance sampling and identify the secondary intersection $\mathbf{x}'$ through sphere tracing. We then compute the pixel color c using Monte Carlo estimation (Sec. 3.6). The rendering results are compared with the observed images to optimize SVBRDF and direct illumination.

3.2. Preliminaries

Multiresolution Hash Encoding. To address the slow training and evaluation speeds associated with MLPs, Instant-NGP[19] introduces a multiresolution hash encoding that reduces neural network complexity without compromising quality. Specifically, it employs a hierarchical set of hash tables that store trainable feature vectors, each layer corresponding to a different resolution. For a 3D point, it computes the multiresolution hash encoding by linearly interpolating and concatenating the learnable feature

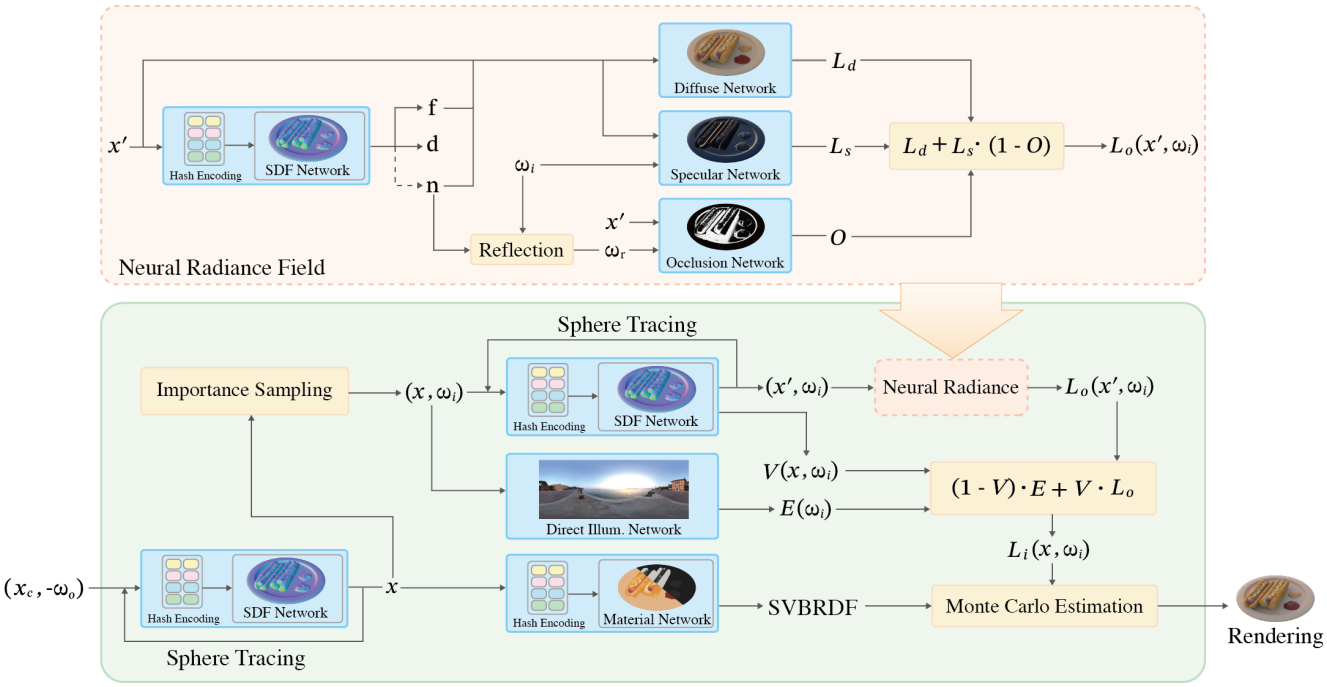

Figure 1. **Forward rendering.** For each pixel, we first compute the intersection $\mathbf{x}$ between the camera ray and the surface using sphere tracing and generate incoming light directions $(\mathbf{x}, \omega_i)$ through importance sampling. We then evaluate the type $V(\mathbf{x}, \omega_i)$ of the incoming light and determine the second intersection $\mathbf{x}'$ between the incoming ray and the surface. The indirect illumination $L_o(\mathbf{x}', \omega_i)$ is derived from the neural radiance field, combining diffuse radiance L_d, specular radiance L_s, and occlusion probability $\mathcal{O}$ as defined in Equation 4. We obtain the direct illumination $E(\omega_i)$ and the SVBRDF from the direct illumination and material network. Finally, we compute the incident radiance $L_i(\mathbf{x}, \omega_i)$ using Equation 11 and calculate the pixel color via Monte Carlo estimation.

vectors from neighboring voxels across layers. This approach significantly enhances training speed and captures high-frequency local details through a compact neural network augmented with multiresolution hash encoding.

The Rendering Equation. For a surface point $\mathbf{x}$, we compute its color $\mathbf{c}$ using the rendering equation[10]:

$$\mathbf{c}(\mathbf{x}, \omega_o) = \int_{\Omega} L_i(\mathbf{x}, \omega_i) f_r(\mathbf{x}, \omega_i, \omega_o)(\omega_i, \mathbf{n})\, d\omega_i, \quad (1)$$

where $L_i(\mathbf{x}, \omega_i)$ denotes the incoming radiance at point $\mathbf{x}$ from direction ω_i and f_r is the BRDF function. The output color $\mathbf{c}(\mathbf{x}, \omega_o)$ represents the integral of reflected light over the hemisphere around the surface normal $\mathbf{n}$ in direction ω_o.

3.3. Hash-encoded SDF

Modeling geometric surfaces using neural network encoded Signed Distance Functions(SDF)[27, 30] has become a prevalent approach. The task requires a complex network architecture with significant depth and breadth to capture subtle local geometric variations. However, in our work, employing NeuS[27] for geometry reconstruction and sphere tracing[7, 15] for localizing surface points necessitates frequent access to the SDF network, exacerbating the computational complexity and training duration.

Instant-NGP[19] can reduce the computational complexity of its neural network by integrating a multiresolution hash table of trainable feature vectors, which enhances efficiency and maintains the capacity to capture high-frequency local details. Inspired by this, we parameterize a shallow SDF multilayer perceptron (MLP) with multiresolution hash encoding to accelerate the training process.

SDF Network. Each 3D coordinate $\mathbf{x}$ is encoded via multiresolution hash encoding $h_g(\mathbf{x})$, facilitated by a trainable hash table. Subsequently, the SDF network accepts $\mathbf{x}$ and $h_g(\mathbf{x})$ as inputs and outputs the SDF value $s \in \mathbb{R}$ and the geometric feature vector $\mathbf{f} \in \mathbb{R}^{15}$:

$$(s, \mathbf{f}) = G(h_g(\mathbf{x}), \mathbf{x}). \quad (2)$$

The normal at point $\mathbf{x}$ is computed as

$$\mathbf{n} = \frac{\partial s}{\partial \mathbf{x}}, \quad (3)$$

where $\frac{\partial s}{\partial \mathbf{x}}$ denotes the gradient of the SDF value with respect to $\mathbf{x}$. We approximate the gradient using the finite difference method, similar to [36].

3.4. Geometry Reconstruction

We follow the NeuS[27] strategy to represent object surfaces using Signed Distance Functions (SDF), but di-

verge by employing a hash-encoded SDF instead of an MLP-encoded version. In rendering glossy objects, self-occlusion creates pronounced brightness contrast across shadow edges in 2D images. During surface reconstruction, geometric inaccuracies such as depressions and unevenness often occur within occlusion boundary regions. To address this, we decompose the neural radiance into diffuse and specular components and introduce an occlusion network to predict occlusion probabilities.

Diffuse Network. Given a 3D coordinate $\mathbf{x}$, the diffuse radiance remains constant regardless of the viewing direction, ensuring uniqueness at each location. We map 3D points $\mathbf{x}$, 3D normals $\mathbf{n}$, and feature vectors $\mathbf{f}$ to determine the diffuse radiance L_d. The diffuse radiance is computed as $L_d(\mathbf{x}) = D(\mathbf{x}, \mathbf{n}, \mathbf{f})$.

Specular Network. The viewing direction strongly influences specular radiance. The specular network predicts the specular radiance for a 3D coordinate $\mathbf{x}$ and incoming viewing direction ω_o, assuming no occlusion is present. The predicted specular radiance maintains improved continuity across different viewing directions and typically does not exhibit discontinuities due to occlusion. Incoming view directions ω_o is encoded via spherical harmonics encoding $y(\omega_o)$. Subsequently, The model takes 3D points $\mathbf{x}$, $y(\omega_o)$, 3D normals $\mathbf{n}$ and feature vectors $\mathbf{f}$ as inputs to output specular radiance L_s. The specular radiance is computed as $L_s(\mathbf{x}, \omega_o) = S(\mathbf{x}, y(\omega_o), \mathbf{n}, \mathbf{f})$.

Occlusion Network. We find that geometric reconstruction errors in glossy objects due to occlusion are primarily caused by obstructions of specular radiance. We define a neural network to predict the occlusion probability of specular radiance, mapping 3D points $\mathbf{x}$ and reflective directions ω_r to occlusion probabilities $\mathcal{O}$. The occlusion probabilities is computed as $\mathcal{O}(\mathbf{x}, \omega_r)$, where ω_r is defined by $\omega_r = -2(\omega_o \cdot \mathbf{n})\mathbf{n} + \omega_o$, with ω_o representing the incoming view direction and $\mathbf{n}$ being 3D normals.

We observe that unconstrained occlusion probabilities, learned solely from rendering losses, lead to inconsistencies between predicted occlusion probabilities and reconstructed geometry. This results in incorrect decomposition of diffuse and specular radiance and subsequent geometric prediction errors, as illustrated in Figure 2. To address this, we follow the occlusion loss proposed in [16] to regulate the predicted occlusion probability.

Specifically, given a ray emitted from the sample point in the reflective direction, we determine the occlusion probability $\mathcal{O}_{march}$ by ray matching within the hash-encoded SDF. The occlusion loss is then computed as $\ell_{occ} = ||\mathcal{O}_{march} - \mathcal{O}||_1$. Since reflection occurs on object surfaces, we compute occlusion loss using sample points from rays emitted from the camera center where the SDF values are below the threshold $th = 0.001$.

To synthesize appearances, we render images from the hash-encoded SDF using volume rendering[18]. For a camera ray $\mathbf{o} + t\omega_o$ emitted from the camera center $\mathbf{o}$ along the direction ω_o, we sample n points along the ray. The color rendered for the camera ray is computed as $\sum_i^n w_i L_o(\mathbf{x}_i, \omega_o)$, where w_i represents the weight for the i-th point $\mathbf{x}_i$, determined based on the SDF values, following the opaque density technique described in [27], and $L_o(\mathbf{x}_i, \omega_o)$ is the color of this point $\mathbf{x}_i$, computed as follows:

$$L_o(\mathbf{x}_i, \omega_o) = L_d(\mathbf{x}_i) + L_s(\mathbf{x}_i, \omega_o) \cdot (1 - \mathcal{O}(\mathbf{x}_i, \omega_r)). \quad (4)$$

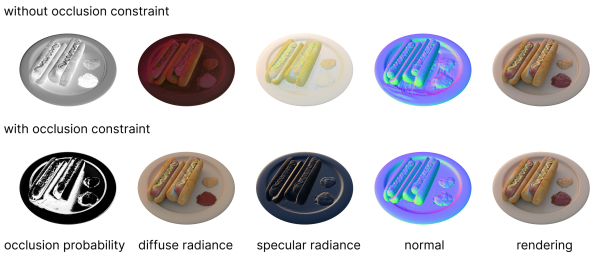

Figure 2. **Effects of occlusion constraint.** Without occlusion constraint, the predicted occlusion probabilities are inconsistent with the reconstructed geometry, leading to random decomposition of diffuse and specular radiance and subsequent errors in geometric reconstruction. With occlusion constraint, the reconstruction of all components is correct.

3.5. BRDF

We adopt the Cook-Torrance BRDF[4], parameterizing the BRDF of a surface point $\mathbf{x}$ with its diffuse albedo $\mathbf{a}(\mathbf{x}) \in [0, 1]^3$, roughness $r(\mathbf{x}) \in [0, 1]$, and incorporating a basic reflection ratio $F_0 = 0.04$ suitable for common dielectric surfaces.

Employing a material autoencoder to predict roughness and albedo at a surface point[5, 35] represents a viable option. However, this method struggles to capture high-frequency, local details due to excessive smoothing from its sparse latent space and insufficient network capacity.

Our solution represents the spatially varying bidirectional reflectance distribution function (SVBRDF) using a small multilayer perceptron (MLP) augmented by a multiresolution hash table of trainable feature vectors. For each 3D position $\mathbf{x}$, we map it to its multiresolution hash encoding $h_m(\mathbf{x})$. Our material MLP then takes the 3D position $\mathbf{x}$ along with its hash encoding $h_m(\mathbf{x})$ as input and outputs the diffuse albedo $\mathbf{a}$ and roughness r:

$$(\mathbf{a}, r) = M(h_m(\mathbf{x}), \mathbf{x}). \quad (5)$$

Due to the insufficient supervision of some surface points, directly applying this model leads to noisy predictions. To solve it, we introduce a prior smoothness based

on the assumption that spatially adjacent points have similar albedo and roughness. The loss function reflecting this smoothness prior is given by:

$$\ell_a = ||\mathbf{a}(\mathbf{x}) - \mathbf{a}(\mathbf{x}+\epsilon)||_1 \tag{6}$$

for albedo, and

$$\ell_r = ||r(\mathbf{x}) - r(\mathbf{x}+\epsilon)||_1. \tag{7}$$

for roughness, where ϵ denotes a random variable taken from a normal distribution characterized by a mean of zero and a variance of 0.01.

3.6. Rendering

We decompose the BRDF function f_r into a diffuse term f_d and a specular term f_s, represented by $f_r = f_d + f_s$. The diffuse term is computed as $f_d = \frac{\mathbf{a}}{\pi}$, and the specular term is computed as

$$f_s(\omega_i, \omega_o) = \frac{D(\mathbf{n}, \mathbf{h}; r)F(\omega_o, \mathbf{h})G(\mathbf{n}, \omega_o, \omega_i; r)}{4(\mathbf{n}\cdot\omega_o)(\mathbf{n}\cdot\omega_i)}, \tag{8}$$

where $\mathbf{h}$ is half-way vector, ω_o is the outgoing viewing direction, D is the normal distribution function, G is the geometry function and F is the Fresnel term.

We use Monte Carlo sampling to calculate diffuse and specular colors. To enhance the efficiency of Monte Carlo estimation, we adopt importance sampling techniques, including cosine sampling and GGX importance sampling[8].

Importance Sampling. Given a surface point, we use cosine sampling with N_d rays to compute diffuse color:

$$\mathbf{c}_{diffuse} = \frac{1}{N_d}\sum_{i}^{N_d} \mathbf{a} L_i(\mathbf{x}, \omega_i). \tag{9}$$

We observed that using only GGX-sampled rays to calculate specular color introduces erroneous noise in the albedo. Given a surface point, we sample N_d rays using cosine sampling and N_s rays using GGX importance sampling. For the specular color, we integrate these methods using multiple importance sampling[22, 24], summing over a total of $N_d + N_s$ rays. The specular color is computed as

$$\mathbf{c}_{specular} = \frac{1}{N_s + N_d}\sum_{i}^{N_s+N_d} \frac{L_i(\mathbf{x}, \omega_i) f_s(\omega_i, \omega_o)(\mathbf{n}\cdot\omega_i)}{P(\omega_i)}, \tag{10}$$

where i is the i-th sample ray, ω_i is the direction of the ray, and $P(\omega_i)$ is the combined sampling probability for i-th ray.

The incoming radiance includes direct illumination from light sources and indirect illumination, reflecting multiple times off object surfaces before reaching a shading point. We use sphere tracing[7, 15] to identify incoming radiance types and determine the second intersection $\mathbf{x}'$ that reflects indirect illumination. For a surface point $\mathbf{x}$, the incoming radiance is computed by

$$L_i(\mathbf{x}, \omega_i) = (1 - V(\mathbf{x}, \omega_i))E(\omega_i) + V(\mathbf{x}, \omega_i)L_o(\mathbf{x}', \omega_i), \tag{11}$$

where $E(\omega_i)$ is the direct radiance along direction ω_i, $L_o(\mathbf{x}', \omega_i)$ represents the indirect radiance along the direction ω_i, and $V(\mathbf{x}, \omega_i)$ indicates the type of illumination in direction ω_i.

Traditional lighting representation methods, such as Spherical Harmonics (SH) and Spherical Gaussians (SG), typically rely on predefined basis functions and fixed model structures, which limit their adaptability to different scenes. In contrast, Multi-Layer Perceptrons (MLPs) leverage data-driven learning to adaptively optimize their structure and parameters, effectively capturing the diversity and complexity of lighting. Therefore, we use MLPs to model direct illumination. For indirect illumination, theoretically, we should recursively compute the outgoing radiance at position $\mathbf{x}'$ along direction ω_i using path tracing. However, due to the computational complexity, we cache the indirect illumination using the neural radiance derived from geometry reconstruction, similar to [35], as described in Equation 4.

We find that directly using random variables in importance sampling can lead to errors in BRDF calculations. To solve this problem, we employ Fibonacci sphere sampling[17] on the hemisphere to obtain all samples. Additionally, for each surface point, we introduce a random offset to the azimuthal angle during the sampling process, enhancing the relighting quality.

3.7. Training

We employ a two-stage training process to optimize geometry, SVBRDF, and direct illumination. In the first stage, we optimize the hash-encoded SDF G, diffuse radiance D, specular radiance S and occlusion probability $\mathcal{O}$. The loss in the first stage is:

$$\ell_{geo} = \ell_{color} + \lambda_{reg}\ell_{reg} + \lambda_{mask}\ell_{mask} + \lambda_{occ}\ell_{occ}, \tag{12}$$

where ℓ_{color} is a color loss, ℓ_{reg} is an Eikonal term and ℓ_{mask} is a mask loss, each similar to that used in [27]. In the second stage, we optimize albedo $\mathbf{a}$, roughness r and direct illumination E while fine-tuning the geometry. The loss in the second stage is:

$$\ell_{render} = \ell_{recon} + \ell_{geo} + \lambda_a\ell_a + \lambda_r\ell_r, \tag{13}$$

where ℓ_{recon} is the reconstruction loss between the renderings and the captured images. In our experiments, we adjust the weights as follows: λ_{reg} from 0.4 to 0.1 and λ_{mask} from 1.0 to 0.1 over 300 epochs, λ_{occ} from 0.1 to 0.005 over 200 epochs. We set the constants λ_a and λ_r at 0.01 and 0.04 respectively.

Figure 3. **Comparisons with previous work.** We show the predicted normal, aligned albedo, roughness, and the environment map of MII [35], TensoIR [9], TensoSDF [14], and our method on two scenes. Additionally, we compare renderings from a novel view under the original lighting and renderings with both novel views and novel lighting.

We configure the experiments with 128 rays for both N_d and N_s. The MLP architectures for SDF, material, and occlusion each include 2 layers with 64 hidden units. The direct illumination MLP contains 4 layers with 256 hidden units. The diffuse and specular MLP contains 3 layers with 64 hidden units. We apply hash encoding with a multiresolution hash grid, featuring 16 levels of resolution ranging from 16 to 2048, to 3D locations and use spherical harmonics encoding of degree 5 for 3D directions. Our model is implemented in PyTorch[21] and optimized with Adam[12] at a learning rate of $5e^{-4}$. The first stage runs for 1000 epochs and the second for 200 epochs on a single V100 GPU, each epoch processing 1024 pixels, taking about 2 and 1 hours respectively. To ensure stability, the geometry is fixed for the first 50 epochs of the second stage.

4. Experiments

4.1. Synthetic Data

We evaluate our method on the synthetic scenes in [35]. Each scene contains 100 training images rendered under the natural environment map with corresponding foreground masks, and 200 test images with albedo and roughness maps to evaluate the inverse rendering ability. For relighting performance, each scene contains additional 200 images rendered under two different environment maps respectively. The image resolution is set to 800×800.

4.2. Baseline Comparisons

We take MII[35], TensoIR[9] and TensoSDF[14] as baselines for quantitative comparison on the synthetic datasets. Both methods decompose scenes into ge-

Method	Normal	Roughness	Light	Albedo			Aligned Albedo			View Synthsis			Relighting			Runtime
	MAE° ↓	MSE ↓	MSE ↓	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓	hrs ↓
MII[35]	3.52	0.0408	0.0149	26.3102	0.9362	0.0599	28.1918	0.9389	0.0605	31.7689	0.9547	0.0667	32.0602	0.9561	0.0664	12
TensoIR*[9]	4.19	0.0661	0.0899	23.9417	0.9369	0.0680	28.5685	0.9421	0.0555	**35.8839**	**0.9777**	0.0369	26.3639	0.9434	0.0660	4
TensoSDF*[14]	3.64	0.2384	-	19.8022	0.9035	0.0987	23.4351	0.9236	0.0894	32.7417	0.9661	0.0522	20.2758	0.9266	0.0733	15
Ours	**2.83**	0.0200	**0.0117**	**27.9586**	**0.9566**	**0.0421**	**30.4865**	**0.9570**	**0.0419**	34.0569	0.9760	**0.0346**	**33.3362**	**0.9655**	**0.0484**	**3**
w/o occ. & ind. illum.	-	0.0434	0.0198	24.3919	0.9435	0.0541	27.2887	0.9454	0.0517	32.7657	0.9727	0.0382	31.3157	0.9628	0.0512	-
w/o ind. illum.	-	**0.0172**	0.0178	26.6950	0.9512	0.0550	28.9566	0.9525	0.0516	33.4462	0.9748	0.0360	32.3108	0.9643	0.0512	-

Table 1. **Quantitative evaluations.** We report the mean performance across test images from all four synthetic scenes. We modified TensoIR[9] and TensoSDF[14] to utilize albedo instead of aligned albedo during relighting, consistent with MII[35], to ensure identical experimental conditions. The findings indicate that our approach outperforms others in normal, albedo estimation, light estimation, and relighting. Some metrics in view synthesis perform less effectively compared to TensoIR[9], and the roughness estimation of our full model is slightly inferior to the baseline models.

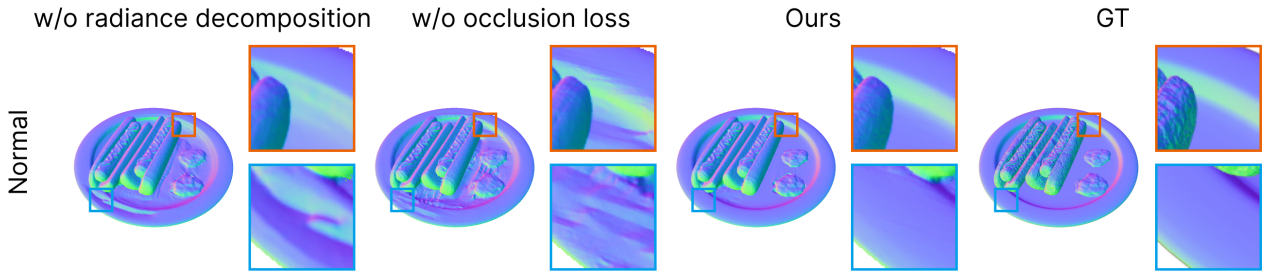

Figure 4. **Ablations on geometry reconstruction.** Please refer to Section 4.3 for detailed descriptions.

ometry, materials, and illumination. We adopt Peak Signal-to-Noise Ratio(PSNR), Structural Similarity Index Measure(SSIM)[29], and Learned Perceptual Image Patch Similarity(LPIPS)[33] to evaluate four key aspects: novel view synthesis, albedo, aligned albedo, and relighting performance. Additionally, we adopt the Mean Squared Error(MSE) to evaluate roughness and illumination.

MII[35] trains geometry, SVBRDF, and environmental lighting in three stages, using Spherical Gaussians(SGs)[26] to represent direct and indirect light. In the second stage, it samples visibility and indirect light at the surface point from the radiance field pre-trained in the first stage, catching each into a separate neural network. Table 1 and Figure 3 demonstrate that our method is quantitatively and qualitatively superior to MII. One limitation of MII is its use of Spherical Gaussians (SGs) to represent the visibility and indirect illumination of surface points, which are ineffective at capturing high-frequency variations. Additionally, MII caches visibility and indirect illumination for only a subset of surface points in the radiance field. This strategy fails to account for the significant differences in visibility and indirect illumination between adjacent surface points. As a result, indirect light is incorrectly baked into the albedo, and occluded areas exhibit exaggerated roughness.

TensoIR[9] jointly recovers geometry, SVBRDF, environmental lighting, and the radiance field, modeling both shadows and indirect illumination. Table 1 shows that our results are superior to TensoIR's except for view synthesis. A potential explanation is that TensoIR employs lighting-intensity importance sampling to simulate incoming light rays, simplifying the interaction between light and objects in the scene. This reduces the problem's ambiguity, achieving a local optimum for view synthesis, but compromises the reconstruction quality of other components. Figure 3 illustrates that TensoIR's albedo estimation introduces erroneous brightness, and the lighting estimation significantly deviates from the ground truth. Moreover, the roughness is inaccurately recovered, with some regions predicted in stack contrast to the ground truth.

TensoSDF [14] integrates a roughness-aware radiance and reflectance field with a tensor-based SDF representation, enabling the reconstruction of scenes with arbitrary smoothness. As shown in Table 1 and Figure 3, our method surpasses TensoSDF both quantitatively and qualitatively. We identify the core strength of TensoSDF in its roughness-aware fusion of radiance and reflectance fields. However, for materials with moderate roughness, the surface characteristics often exhibit a blend of smooth and rough properties, introducing complexity that challenges the model's ability to capture fine details accurately. Figure 3 further highlights that TensoSDF produces significant errors in albedo prediction, deviating substantially from the ground truth. While its predicted roughness is relatively high, it is more reasonable compared to TensoIR. However, the inaccurate albedo reconstruction leads to excessive brightness

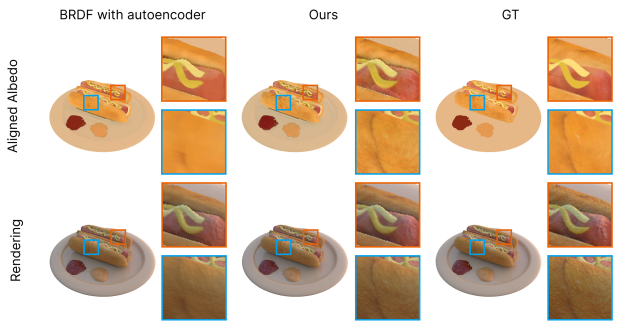

Figure 5. **Ablations on BRDF representation.** Please refer to Section 4.3 for detailed descriptions.

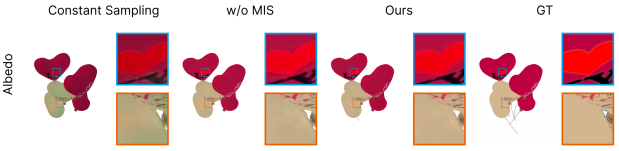

Figure 6. **Ablations on light sampling.** Please refer to Section 4.3 for detailed descriptions.

in the relighting results.

4.3. Ablation Studies

Ablations on geometry reconstruction. We conduct ablation studies excluding radiance decomposition and occlusion loss and compare the results in Figure 4. Without radiance decomposition, the recovered surface in occluded areas exhibits inaccuracies, including depressions and unevenness. Without occlusion loss, the reconstructed geometry introduces significant noise due to erroneous estimates of occlusion probabilities. In contrast, our method recovers accurate geometric shapes, precisely reconstructing surfaces in occluded regions.

Ablations on BRDF representation. We ablate the BRDF model using an autoencoder and compare our method in Figure 5. Representing albedo with a hash-encoded MLP enables the network to capture high-frequency and accurate details. In contrast, using an autoencoder for albedo representation leads the network to predict a smoother albedo, resulting in renderings with blurred local details and slightly inferior quality.

Ablations on light sampling. We conduct an ablation study on the effectiveness of the light sampling for albedo reconstruction in Figure 6. "w/o MIS" indicates that multiple importance sampling is not used. When using uniform sampling, the reconstructed albedo appears darker and deviates significantly from the ground truth. Without MIS, more noise is introduced at the texture boundaries of the albedo, whereas our method achieves better albedo reconstruction results.

Ablations on indirect illumination. We ablate the effects of indirect light and visibility in Figure 7 and Table 1. Without modeling visibility, shadows are incorrectly baked into the predicted albedo, and the roughness in occluded areas is significantly exaggerated. Without modeling indirect light, the effects of indirect light are baked into the albedo, leading to erroneous brightness. Table 1 shows that the "w/o ind." scenario is slightly superior in the roughness metric, which we attribute to noise from sampled incoming light rays.

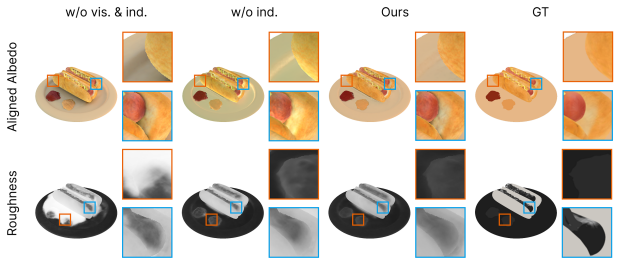

Figure 7. **Ablations on indirect illumination.** Please refer to Section 4.3 for detailed descriptions.

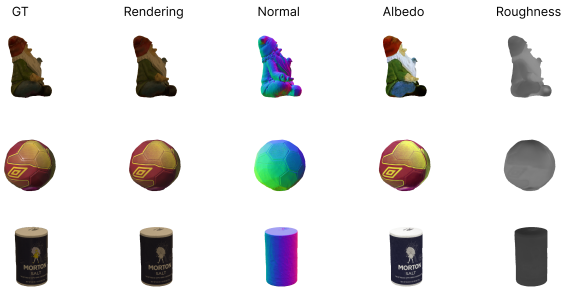

Figure 8. **Results on real captures.** Our method reasonably estimates the geometry and material of real-world objects.

4.4. Results on Real Captures.

We evaluated our method on three real-world datasets from Stanford-ORB[13], each containing 60 training and 10 test images. The images were captured in indoor environments or outdoor scenes with occlusions, resulting in non-ideal ambient lighting conditions. In Figure 8, we show each scene's re-rendering, geometry, and material results. Our method successfully predicts plausible geometry and material properties.

5. Conclusion

Our paper presents a two-stage inverse rendering pipeline that effectively reconstructs geometry, material, and illumination. Our method separates neural radiance into diffuse, specular, and occlusion components, improving surface reconstruction quality. We use Monte Carlo-based path tracing to model shadows and inter-reflections. To enhance efficiency and quality, we employ two hash-

encoded MLPs to represent geometry and material, respectively. Experimental results demonstrate that our approach accurately recovers geometry, roughness, and high-quality albedo, while supporting realistic re-rendering and relighting.

Our method has the following limitations. First, when using sphere tracing to locate surface points, the surface geometry cannot be jointly optimized with material and illumination. Second, to reduce ambiguity in the inverse rendering problem, we assume $F_0 = 0.04$ in the Fresnel term, which is suitable for common dielectric surfaces. We aim to address these issues in future work.

Acknowledgement

This study was funded by the Ningbo Key Research and Development Program (grant number: 2023Z141) and the Ningbo Major Science and Technology Project for Key Tasks (grant number: 2022Z167).

References

[1] M. Boss, R. Braun, V. Jampani, J. T. Barron, C. Liu, and H. P. A. Lensch. Nerd: Neural reflectance decomposition from image collections. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 12684–12694, 2020. 1, 2

[2] M. Boss, V. Jampani, R. Braun, C. Liu, J. Barron, and H. Lensch. Neural-pil: Neural pre-integrated lighting for reflectance decomposition. *Advances in Neural Information Processing Systems*, 34:10691–10704, 2021. 1, 2

[3] A. Chen, Z. Xu, A. Geiger, J. Yu, and H. Su. Tensorf: Tensorial radiance fields. In *European conference on computer vision*, pages 333–350, 2022. 2

[4] R. L. Cook and K. E. Torrance. A reflectance model for computer graphics. *ACM Transactions on Graphics (ToG)*, 1:7–24, 1982. 4

[5] Y. Deng, X. Li, S. Liu, and M.-H. Yang. Physics-based indirect illumination for inverse rendering. In *International Conference on 3D Vision (3DV)*, pages 1249–1258, 2024. 4

[6] A. Gropp, L. Yariv, N. Haim, M. Atzmon, and Y. Lipman. Implicit geometric regularization for learning shapes. In *37th International Conference on Machine Learning: ICML 2020, Online, 13-18 July 2020, Part 5 of 15*, 2021. 2

[7] J. C. Hart. Sphere tracing: A geometric method for the antialiased ray tracing of implicit surfaces. *The Visual Computer*, 12:527–545, 1996. 3, 5

[8] E. Heitz. Sampling the ggx distribution of visible normals. *Journal of Computer Graphics Techniques (JCGT)*, 7:1–13, 2018. 5

[9] H. Jin, I. Liu, P. Xu, X. Zhang, S. Han, S. Bi, X. Zhou, Z. Xu, and H. Su. Tensoir: Tensorial inverse rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 165–174, 2023. 2, 6, 7

[10] J. T. Kajiya. The rendering equation. In *Proceedings of the 13th annual conference on Computer graphics and interactive techniques*, pages 143–150, 1986. 3

[11] B. Kerbl, G. Kopanas, T. Leimkühler, and G. Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Trans. Graph.*, 42(4):139–1, 2023. 2

[12] D. P. Kingma. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 6

[13] Z. Kuang, Y. Zhang, H.-X. Yu, S. Agarwala, E. Wu, J. Wu, et al. Stanford-orb: a real-world 3d object inverse rendering benchmark. *Advances in Neural Information Processing Systems*, 36, 2024. 8

[14] J. Li, L. Wang, L. Zhang, and B. Wang. Tensosdf: Roughness-aware tensorial representation for robust geometry and material reconstruction. 2024. 2, 6, 7

[15] S. Liu, Y. Zhang, S. Peng, B. Shi, M. Pollefeys, and Z. Cui. Dist: Rendering deep implicit signed distance function with differentiable sphere tracing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2019–2028, 2020. 3, 5

[16] Y. Liu, P. Wang, C. Lin, X. Long, J. Wang, L. Liu, T. Komura, and W. Wang. Nero: Neural geometry and brdf reconstruction of reflective objects from multiview images. *ACM Transactions on Graphics (TOG)*, 42:1–22, 2023. 4

[17] R. Marques, C. Bouville, K. Bouatouch, and L. P. Santos. Spherical fibonacci point sets for qmc estimates of illumination integrals. In *Efficient Quadrature Rules for Illumination Integrals: From Quasi Monte Carlo to Bayesian Monte Carlo*, pages 5–29. Springer, 2015. 5

[18] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *ECCV*, 2020. 2, 4

[19] T. Müller, A. Evans, C. Schied, and A. Keller. Instant neural graphics primitives with a multiresolution hash encoding. *ACM transactions on graphics (TOG)*, 41:1–15, 2022. 2, 3

[20] J. Munkberg, J. Hasselgren, T. Shen, J. Gao, W. Chen, A. Evans, T. Müller, and S. Fidler. Extracting triangular 3d models, materials, and lighting from images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8280–8290, 2022. 1

[21] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019. 6

[22] M. Pharr, W. Jakob, and G. Humphreys. *Physically based rendering: From theory to implementation*. 2023. 5

[23] P. P. Srinivasan, B. Deng, X. Zhang, M. Tancik, B. Mildenhall, and J. T. Barron. Nerv: Neural reflectance and visibility fields for relighting and view synthesis. *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7491–7500, 2021. 1, 2

[24] E. Veach and L. J. Guibas. Optimally combining sampling techniques for monte carlo rendering. In *Proceedings of the 22nd annual conference on Computer graphics and interactive techniques*, pages 419–428, 1995. 5

[25] D. Verbin, P. Hedman, B. Mildenhall, T. Zickler, J. T. Barron, and P. P. Srinivasan. Ref-nerf: Structured view-dependent appearance for neural radiance fields. *arXiv e-prints*, 2021. 2

[26] J. Wang, P. Ren, M. Gong, J. Snyder, and B. Guo. All-frequency rendering of dynamic, spatially-varying reflectance. In *ACM SIGGRAPH Asia 2009 papers*, pages 1–10. 2009. 2, 7

[27] P. Wang, L. Liu, Y. Liu, C. Theobalt, T. Komura, and W. Wang. Neus: Learning neural implicit surfaces by volume rendering for multi-view reconstruction. *arXiv preprint arXiv:2106.10689*, 2021. 2, 3, 4, 5

[28] Y. Wang, Q. Han, M. Habermann, K. Daniilidis, C. Theobalt, and L. Liu. Neus2: Fast learning of neural implicit surfaces for multi-view reconstruction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023. 2

[29] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 13:600–612, 2004. 7

[30] L. Yariv, Y. Kasten, D. Moran, M. Galun, M. Atzmon, B. Ronen, and Y. Lipman. Multiview neural surface reconstruction by disentangling geometry and appearance. *Advances in Neural Information Processing Systems*, 33:2492–2502, 2020. 2, 3

[31] J. Zhang, Y. Yao, S. Li, J. Liu, T. Fang, D. Mckinnon, Y. Tsin, and L. Quan. Neilf++: Inter-reflectable light fields for geometry and material estimation. *ArXiv*, abs/2303.17147, 2023. 2

[32] K. Zhang, F. Luan, Q. Wang, K. Bala, and N. Snavely. Physg: Inverse rendering with spherical gaussians for physics-based material editing and relighting. In *The IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5453–5462, 2021. 1, 2

[33] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 586–595, 2018. 7

[34] X. Zhang, P. P. Srinivasan, B. Deng, P. Debevec, W. T. Freeman, and J. T. Barron. NeRFactor: Neural Factorization of Shape and Reflectance Under an Unknown Illumination. *ACM Transactions on Graphics (ToG)*, 40:1–18, 2021. 1, 2

[35] Y. Zhang, J. Sun, X. He, H. Fu, R. Jia, and X. Zhou. Modeling indirect illumination for inverse rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18643–18652, 2022. 1, 2, 4, 5, 6, 7

[36] F. Zhao, Y. Jiang, K. Yao, J. Zhang, L. Wang, H. Dai, Y. Zhong, Y. Zhang, M. Wu, L. Xu, et al. Human performance modeling and rendering via neural animated mesh. *ACM Transactions on Graphics (TOG)*, 41:1–17, 2022. 3