RTR-GS: 3D Gaussian Splatting for Inverse Rendering with Radiance Transfer and Reflection

Anonymous Author(s)
Submission Id: 2556

Figure 1: We propose RTR-GS, a framework for geometry-light-material decomposition from multi-view images. Our method significantly enhances normal estimation and visual realism for reflective surfaces compared to GS-IR [31] and GShader [22]. Additionally, we achieve material and lighting decomposition while accounting for secondary lighting effects through physically-based deferred rendering. The material components include albedo, metallic, and roughness. This high-quality decomposition enables realistic relighting and material editing.

ABSTRACT

3D Gaussian Splatting (3DGS) has demonstrated impressive capabilities in novel view synthesis. However, rendering reflective objects remains a significant challenge, particularly in inverse rendering and relighting. We introduce RTR-GS, a novel inverse rendering framework capable of robustly rendering objects with arbitrary reflectance properties, decomposing BRDF and lighting, and delivering credible relighting results. Given a collection of multi-view images, our method effectively recovers geometric structure through a hybrid rendering model that combines forward rendering for radiance transfer with deferred rendering for reflections. This approach successfully separates high-frequency and low-frequency appearances, mitigating floating artifacts caused by spherical harmonic overfitting when handling high-frequency details. We further refine BRDF and lighting decomposition using an additional physically-based deferred rendering branch. Experimental results show that our method enhances novel view synthesis, normal estimation, decomposition, and relighting while maintaining efficient training inference process.

CCS CONCEPTS

• **Computing methodologies** → **Rasterization**; • **Point-based models**; • **Machine learning approaches**; • **Rendering**;

KEYWORDS

Novel view synthesis, Gaussian Splatting, Relighting

1 INTRODUCTION

Inverse rendering is a long-standing challenge that seeks to decompose a 3D scene's physical attributes—geometry, materials, and lighting—from captured images. This decomposition enables downstream tasks such as relighting and editing. The problem is particularly challenging due to the complex interplay of these attributes during rendering, especially under unknown illumination conditions, which make it inherently under-constrained. Neural Radiance Fields (NeRF) [36] have achieved remarkable success in novel view synthesis, laying the groundwork for inverse rendering. Methods such as [7, 32, 62, 64] use implicit neural representations, like Multi-Layer Perceptrons (MLPs), to decompose physical components. However, MLPs suffer from limited expressiveness and high computational costs, making it challenging to balance quality and efficiency. 3D Gaussian Splatting (3DGS) [25] improves both the speed and quality of learning-based volumetric rendering, and several methods [16, 31, 43] have integrated physically-based rendering into this framework. However, spherical harmonic functions lack the directional resolution needed to accurately represent specular reflections, and overfitting during Gaussian splatting and cloning can introduce floating artifacts.

Accurate geometry is crucial for decomposing materials and lighting from complex appearances. However, high-frequency details can cause overfitting, leading to floating artifacts that deviate from physically smooth surfaces and compromise geometric accuracy. To address this issue, we propose using a reflection map to store specular components, isolating high-frequency appearance details from the radiance component to mitigate overfitting.

Additionally, we replace independent spherical harmonics with radiance transfer rendering, which imposes stronger global low-frequency constraints when computing radiance components. By separating high-frequency and low-frequency appearances, our method enables accurate recovery of geometric structures with arbitrary reflectance properties. Following geometry reconstruction, we model occlusion and indirect illumination by baking visibility into 3D voxels and introducing indirect lighting parameters. This approach reduces aliasing artifacts in albedo, shadows, and lighting during decomposition. Finally, we achieve effective material and lighting decomposition by integrating an additional differentiable, physically-based deferred rendering branch.

The primary contribution of our work is the introduction of a Gaussian splatting-based inverse rendering framework, RTR-GS, which accurately estimates surface normals, bidirectional reflectance distribution functions (BRDF), and environmental lighting from multi-view images of both diffuse and specular objects. Specifically, it includes the following key aspects:

- We propose a 3DGS-based hybrid rendering model that integrates reflection maps with radiance transfer, effectively separating high-frequency and low-frequency appearances. This enables efficient rendering of objects with arbitrary reflectance properties while reducing floating artifacts, thereby improving geometric structure recovery with high-quality normals.
- We further enhance appearance decomposition through a dual-branch rendering approach, enabling efficient and accurate material and lighting decomposition via rational lighting modeling and occlusion data baked into 3D voxels.
- Comprehensive experiments demonstrate that our method achieves state-of-the-art performance in novel view synthesis and relighting, producing credible results for both diffuse and specular objects.

2 RELATED WORK

2.1 Neural representations

Recent advancements in Neural Radiance Fields (NeRF) [36] have garnered significant attention. Subsequent research has focused on enhancing rendering quality [2, 4, 26], improving surface reconstruction [29, 47, 54], and advancing object generation [11, 40, 49, 59], among other areas. Additionally, some methods aim to balance speed and quality [10, 12, 15, 20, 37, 45], facilitating more efficient evaluations.

3D Gaussian Splatting [25] effectively combines radiance field rendering with rasterization by leveraging discrete Gaussian distributions and the splatting technique. Subsequent research has focused on enhancing rendering quality [33, 57], more accurate geometry reconstruction [21, 35, 58], expanding editability [34, 60], and increasing scalability [39]. However, these methods do not decompose appearance into materials and lighting, limiting their suitability for relighting and editing tasks.

2.2 Inverse rendering

Inverse rendering aims to decompose physically-based attributes from observations, including geometry, material, and lighting. A variety of methods simplify this problem by assuming controllable lighting conditions [1, 5, 6, 17, 41]. Some works relax these assumptions to consider direct lighting effects [7, 8, 62]. These works [13, 51, 53, 61, 64, 65] model secondary lighting effects using additional MLPs. To reduce computational overhead, some methods [23, 28] employ tensor decomposition techniques inspired by TensoRF [12]. For compatibility with existing rendering pipelines, NvDiffrec [38] and NvDiffrecMC [19] utilize differentiable rendering with rasterization or ray-tracing pipelines.

Methods based on 3D Gaussian Splatting (3DGS) have significantly accelerated training and rendering. GS-IR [31], GIR [43], and R3DG [16] constrain surface normals using pseudo normals derived from depth and model shadows and indirect lighting through baking or ray-tracing. By leveraging pre-computed radiance transfer, PRT-GS [18] enables relighting, including secondary lighting effects. Phys3DGS [14] integrates 3D Gaussian splats with mesh-based representations. Although these methods retain the high efficiency of 3DGS, using spherical harmonic functions as a radiance representation for geometry recovery often introduces floating artifacts on reflective surfaces, leading to geometric inaccuracies.

2.3 Reflective object reconstruction

Reconstructing reflective objects poses a significant challenge in inverse rendering tasks due to the high-frequency appearance changes that result in view inconsistencies. Ref-NeRF [46] tries to address this by using reflection directions instead of view directions and introducing Integrating Direction Encoding (IDE) to model reflections effectively. NeRO [32] explicitly models the reflection process. Spec-Gaussian [52] simulates reflections using anisotropic Gaussians. Deferred rendering approaches, such as DeferredGS [50], 3DGS-DR [55], GS-ROR [66], and GUS-IR[30] replace forward rendering to better handle reflections. GaussianShader [22] separates specular components and incorporates residual terms to capture secondary lighting effects. Additionally, PRD-GS [56] introduces progressive radiance distillation.

Inspired by these works, we adopt 3D Gaussians as the scene representation and develop an inverse rendering framework capable of effectively rendering object with arbitrary reflectance properties while also decomposing material and lighting components.

3 METHOD

3.1 Overview

Figure 2 illustrates the overall framework of the proposed RTR-GS. We initialize 3D Gaussians using sparse point clouds generated randomly or estimated by COLMAP [42]. To model reflections, it is essential to define the normals for the Gaussians. We define normals as the shortest axis of each Gaussian, oriented toward the viewing direction, and optimize them synergistically using deferred rendering of reflections and pseudo-normals derived from a depth map (Sec. 3.2). Subsequently, we refine the Gaussians by introducing additional parameters and integrating key components into a hybrid rendering model (Sec. 3.3). This model combines radiance from forward rendering with reflections from deferred rendering, effectively separating high-frequency and low-frequency appearances to better represent complex materials and achieve high-quality scene reconstruction. Next, we decompose the appearance using differentiable physically-based deferred rendering, incorporating occlusion

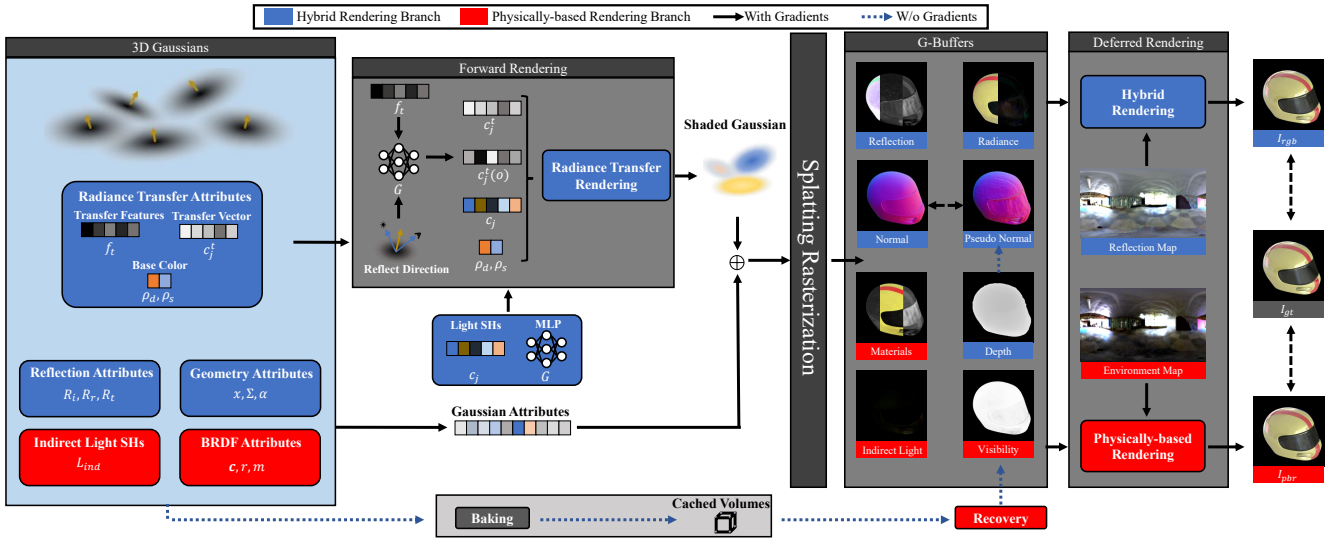

Figure 2: RTR-GS Rendering Pipeline. Our rendering pipeline consists of a hybrid rendering branch and a physically-based rendering branch. The hybrid rendering branch computes the radiance color for each Gaussian using forward rendering through radiance transfer, which is then blended with the reflections from deferred rendering after splatting. The physically-based rendering branch is fully implemented during the deferred rendering phase. Initially, the hybrid rendering branch reconstructs the fundamental geometric structure and stores visibility in voxel grids. The physically-based rendering branch is then activated to further decompose material appearances.

baking, indirect lighting modeling, and additional BRDF parameters. During this process, we employ two rendering branches simultaneously to refine the geometry (Sec. 3.4). Finally, we enhance the results through rendering losses and additional regularization terms (Sec. 3.5).

3.2 Deferred Rendering and Normal Modeling

In the 3DGS framework, the attributes of multiple Gaussians are blended in the image plane using splatting and alpha blending, as follows:

$$I_f = \sum_{i=0}^{N} f_i \alpha_i T_i \tag{1}$$

where α_i is the opacity, $T_i = \prod_{j=1}^{i-1}(1-\alpha_j)$ represents the accumulated transmittance, f_i denotes the parameters of the i-th Gaussian, and I_f represents the splatted screen-space attribute buffer. In vanilla 3DGS [25], outgoing radiance is computed per-Gaussian before blending. This process is referred to as forward rendering. Additionally, the attributes associated with each Gaussian can be transformed into screen space for subsequent shading, a process known as deferred rendering. The following section explains our normal design and optimization based on the deferred rendering implementation.

Accurate normals are essential for modeling reflection. We define the normal direction as the shortest axis of the Gaussian. During the optimization process, the Gaussian shape typically flattens as it aligns with the surface, causing the shortest axis to correspond to a larger area. Similar to GS-IR [31] and R3DG [16], we optimize normals by enforcing consistency between the pseudo-normal map $\hat{\mathbf{n}}_\mathbf{d}$, derived from the depth map, and the Gaussian normals map $\mathbf{n}$, as follows:

$$\mathcal{L}_n = \|\mathbf{n} - \hat{\mathbf{n}}_\mathbf{d}\|_2 \tag{2}$$

This constraint is effective in optimizing normals when the depth map is smooth enough. Additionally, normals are used to compute reflection directions and contribute to deferred rendering. This process enables rendering losses to be backpropagated to the normals, refining the Gaussian shape. When specular reflection is dominant, rendering losses from reflections primarily drive normal optimization. Conversely, in diffuse regions, depth-derived pseudo-normals impose a stronger constraint. Figure 3 illustrates the normal optimization process. Inspired by 3DGS-DR [55], we also introduce a simplified normal propagation mechanism that periodically enhances Gaussian opacity, improving the model's robustness against extreme specular reflections.

3.3 Hybrid Rendering and Radiance Transfer

To effectively render appearances with diverse variations and to mitigate Gaussian floating artifacts caused by limited representation capability, we propose a hybrid rendering approach to replace the spherical harmonics-based forward rendering in 3DGS [25]. Our hybrid rendering model separates radiance and reflection to capture low-frequency and high-frequency components, respectively. Specifically, the radiance is computed using forward rendering, while the reflection is obtained through deferred rendering. The two components are then adaptively blended based on the reflection intensity as follows:

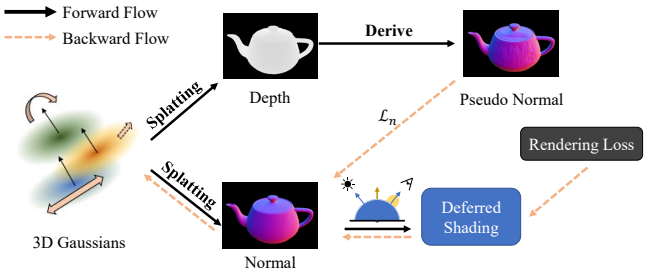

Figure 3: By adjusting the shapes of the Gaussians using the pseudo normals and gradients from the reflection map, the normals are optimized.

$$I_{rgb} = C_r \cdot (1.0 - R_i) + C_{ref} \cdot R_i \tag{3}$$

where C_r is the radiance color, C_{ref} is the reflection color, and R_i is the reflection intensity. The final blending is done in screen space. Further details on the reflection and radiance components are provided in the following sections.

Reflection. In forward rendering, BRDF lobes are computed individually using the respective normal of each Gaussian and are then blended after shading. However, this blending process broadens the final BRDF lobe, resulting in blurry rendering effects. In contrast, deferred rendering generates a single BRDF lobe based on the blended normal, providing higher precision and better preservation of BRDF sharpness. Similar observations have been analyzed in GUS-IR [30] and GS-ROR [66].

For each Gaussian, we introduce additional reflection attributes for deferred rendering: reflection tint R_t and reflection roughness R_r. We adopt a microfacet BRDF to simulate surfaces with varying roughness levels and achieve efficient computation using the split-sum approximation [24]. The final reflection color is computed as:

$$C_{ref} = R_t \cdot F_{ref}(E_r, R_r, \mathbf{n}, \mathbf{v}) \tag{4}$$

where E_r is a learnable reflection map, $\mathbf{n}$ and $\mathbf{v}$ denote the normal and the view direction, respectively. F_{ref} represents the split-sum approximation [24], which will be explained in more detail in Section 3.4.

Radiance. Inspired by Precomputed Radiance Transfer (PRT) [44], we adopt radiance transfer instead of spherical harmonics to compute outgoing radiance. Firstly, we will describe how radiance transfer is used to shade each Gaussian, including both view-independent and view-dependent components. Then we will explain the motivation behind using radiance transfer.

The view-independent component is consistent with the radiance transfer rendering in PRT. This calculation approximates the diffuse part of rendering equation as a dot product of two vectors as follows:

$$C_d \approx \boldsymbol{\rho_d} \sum_{j=0}^{n^2} c_j c_j^t \tag{5}$$

where $\boldsymbol{\rho_d}$ represents the diffuse base color, c_j denotes the coefficients of the spherical harmonics lighting, and c_j^t represents the transfer vector. Notably, all Gaussians share the same spherical harmonics lighting c_j but use individual transfer vector c_j^t.

For the view-dependent component, following the derivation in PRT [44], we need to compute a radiance transfer matrix to convert environmental lighting into transferred lighting. However, n-order spherical harmonics lighting requires n^2 parameters to store the transfer matrix, leading to rapidly increasing storage costs as the number of Gaussians grows. To address this issue, we adopt neural radiance transfer for the view-dependent component and compute it in a manner similar to the view-independent case. Specifically, for each Gaussian, we introduce a set of randomly initialized radiance transfer features f_t and a specular base color $\boldsymbol{\rho_s}$. We decode f_t and the reflection direction $\mathbf{o}$ using a lightweight MLP G to obtain the neural radiance transfer vector $c_j^t(\mathbf{o})$. The view-dependent outgoing radiance is computed as:

$$C_s(\mathbf{o}) \approx \boldsymbol{\rho_s} \sum_{j=0}^{n^2} c_j c_j^t(\mathbf{o}), \quad with \quad c_j^t(\mathbf{o}) = G(f_t, \mathbf{o}) \tag{6}$$

The total outgoing radiance is given by $C_r = C_d + C_s(\mathbf{o})$. After Gaussian splatting and blending, this radiance further participates in the blending process during deferred rendering. A detailed derivation of our radiance transfer implementation is provided in the supplementary materials.

Compared to spherical harmonics, radiance transfer allows us to maintain enougth representational capacity while providing stronger global low-frequency constraints. In the shading process, all Gaussians share two global components: the spherical harmonics lighting c_j and the MLP G. This design enables shading across Gaussians to be connected through shared components, promoting the representation of overall low-frequency variations. Meanwhile, each Gaussian has its own independent transfer vector and transfer features, along with base color attributes. This enables our radiance transfer representation to better handle components that are difficult to recover in the reflection part, such as local reflections and shadows. Figure 4 illustrates the differences between our radiance transfer representation and spherical harmonics in modeling the radiance component. While the rendering results exhibit comparable visual quality, radiance transfer demonstrates better performance in low-frequency component fitting, prevents artifact generation, and maintains geometric smoothness.

3.4 Illumination Modeling and Decomposition

We primarily use differentiable physically-based deferred rendering to decompose appearance into material and lighting components. To prevent aliasing artifacts in shadows, lighting, and albedo, we leverage the recovered geometric structure to bake occlusion information into a voxel grid, following the approach in GS-IR [31]. Specifically, we set the background color to white and assign black to the Gaussian regions. The scene is then projected to generate a cubemap texture, which is converted into spherical harmonics coefficients and stored in the voxel grid. In the following, we describe our material and illumination modeling in detail.

For materials, we assign BRDF attributes to each Gaussian, including albedo $\mathbf{c}$, metallic m, and roughness r. For illumination, we use an environmental cubemap to implement image-based lighting

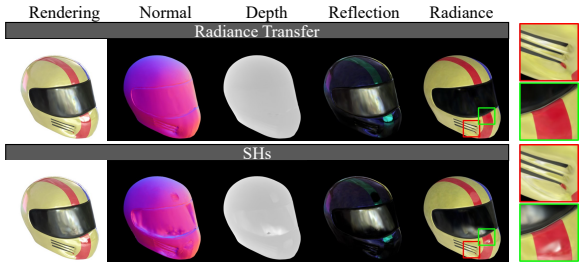

Figure 4: Radiance transfer provides a better representation of low-frequency appearances and helps prevent artifacts caused by overfitting high-frequency details.Such artifacts can degrade the smoothness of depth and normal estimations, reducing the quality of the reconstructed geometry, and adversely affect subsequent decomposition processes.

(IBL) for handling direct lighting. Additionally, we add a parameter $L_{ind} \in [0,1]^3$ for each Gaussian to represent diffuse indirect lighting. The rendering equation $L(\mathbf{o}) = \int_\Omega L_i(\mathbf{i}) f(\mathbf{i},\mathbf{o})(\mathbf{i}\cdot\mathbf{n}) d\mathbf{i}$ is separated into diffuse and specular components to simplify computation. The diffuse component L_d is computed as follows:

$$\begin{aligned} L_d(\mathbf{x}) &= \frac{\boldsymbol{c}}{\pi}\int_\Omega L_i(\mathbf{x},\mathbf{i})(\mathbf{n}\cdot\mathbf{i}) d\mathbf{i} \\ &= \frac{\boldsymbol{c}}{\pi}[\int_\Omega L_i^{dir}(\mathbf{x},\mathbf{i})(\mathbf{n}\cdot\mathbf{i}) d\mathbf{i} + \int_\Omega L_i^{ind}(\mathbf{x},\mathbf{i})(\mathbf{n},\mathbf{i}) d\mathbf{i}] \\ &\approx \frac{\boldsymbol{c}}{\pi}[V(\mathbf{x}) L_d^{dir}(\mathbf{x}) + (1 - V(\mathbf{x})) L_d^{ind}(\mathbf{x}))] \end{aligned} \tag{7}$$

where $L_d^{dir}(\mathbf{x})$ represents the direct environmental illumination, which depends only on the normal direction $\mathbf{n}$. This value is precomputed for efficiency and stored in a 2D texture. The indirect illumination $L_d^{ind}(\mathbf{x})$ is derived through the splatting and blending of L_{ind}. The visibility term $V(\mathbf{x})$ is determined by applying trilinear interpolation to the precomputed spherical harmonics stored in the baked voxel grid.

For the specular L_s, we employ the split-sum approximation [24], treating it as the product of two independent integrals as follows:

$$L_s(\mathbf{x},\mathbf{o}) \approx \int_\Omega f_s(\mathbf{i},\mathbf{o})(\mathbf{n}\cdot\mathbf{i}) d\mathbf{i} \int_\Omega L_i(\mathbf{x},\mathbf{i}) D(\mathbf{i},\mathbf{o})(\mathbf{n}\cdot\mathbf{i}) d\mathbf{i} \tag{8}$$

where $f(\mathbf{i},\mathbf{o})$ represents the microfacet BRDF [9]. The first term of the integral represents the BRDF, which is independent of the lighting. It is precomputed and stored in a Look-Up Table (LUT). The second term accounts for the incoming radiance modulated by the normal distribution function (NDF) D, which is pre-integrated and represented using a filtered cubemap. Finally, the outgoing radiance is expressed as:

$$L_o(\mathbf{x},\mathbf{o}) = L_d(\mathbf{x}) + L_s(\mathbf{x},\mathbf{o}) \tag{9}$$

After completing deferred rendering, we obtain the final PBR result I_{pbr}.

In the decomposition process, we use both the previously mentioned hybrid rendering and PBR branches simultaneously, rather than freezing the geometric parameters or enabling only the PBR branch. This approach is adopted for two main reasons. Firstly, different rendering models still require corresponding geometric adjustments for proper adaptation, so completely freezing the geometric parameters is undesirable. We need to locally optimize the geometric attributes of the Gaussian to accommodate the newly introduced PBR branch. Secondly, since the PBR-related parameters are initialized randomly, using only PBR can easily lead to drastic changes in the geometric structure, which may render the baked visibility inapplicable. These two points will be further elaborated in the experimental section.

3.5 Optimization

Throughout the training process, we optimize the geometric attributes of the Gaussian, as well as various rendering attributes closely related to the two rendering branches, as illustrated by the 3D Gaussians in Figure 2. In addition, we need to optimize the small MLP G, which is a 3-layer network with 64 hidden units, used to decode the transfer feature and reflection direction, as well as two $6 \times 128 \times 128$ cubemaps: the reflection map for hybrid rendering and the environment map for PBR. We first activate the hybrid rendering branch and optimize the corresponding parameters. After restoring the basic geometric structure, we then activate the PBR branch and optimize all parameters. Finally, we outline the primary loss function and the specialized regularization terms.

Rendering losses. As in 3DGS[25], we calculate the hybrid rendering loss $\mathcal{L}_{HR}$ and PBR loss $\mathcal{L}_{PBR}$ using the following equation:

$$\mathcal{L} = (1-\lambda)\mathcal{L}_1(\hat{I}, I_{gt}) + \lambda\mathcal{L}_{D-SSIM}(\hat{I}, I_{gt}) \tag{10}$$

Light regularization. We apply a light regularization assuming a natural white incident light [32, 38] for optimizing environment map used in PBR as follows:

$$\mathcal{L}_{light} = \sum_c (L_c - \frac{1}{3}\sum_c L_c), c \in \{R,G,B\} \tag{11}$$

Metal reflection prior. Due to the reflective properties of metals, we aim to make the metallic parameter m in the PBR model as close as possible to the reflection intensity R_i in hybrid rendering, as follows:

$$\mathcal{L}_m = \mathcal{L}_1(m, R_i) \tag{12}$$

which encourages our two rendering branches to maintain appearance consistency in high-frequency regions. The effectiveness of this regularization term is discussed in the following section. In addition, we incorporate a bilateral smoothness term $\mathcal{L}_s$ and an object mask constraint $\mathcal{L}_o$. The final loss $\mathcal{L}$ is defined as:

$$\mathcal{L} = \mathcal{L}_{HR} + \lambda_{PBR}\mathcal{L}_{PBR} + \lambda_0\mathcal{L}_{light} + \lambda_1\mathcal{L}_m + \lambda_2\mathcal{L}_n + \mathcal{L}_s + \mathcal{L}_o \tag{13}$$

where $\lambda_{PBR} = 0$ or 1, $\lambda_0 = 0.003$, $\lambda_1 = 0.1$, $\lambda_2 = 0.02$. Detailed descriptions of $\mathcal{L}_s$ and $\mathcal{L}_o$ are provided in the supplementary materials.

4 EXPERIMENTS

4.1 Evaluation Setup

Dataset and Metrics. For synthetic objects in the TensoIR [23] and Shiny Blender [46] datasets, as well as real objects in the Stanford

Table 1: NVS quality, training time and FPS on TensoIR, Shiny Blender and Stanford ORB datasets. "HR" represents our hybrid rendering branch.

Methods	TensoIR			Shiny Blender			Stanford ORB			Training Time	FPS
	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓		
NeRO	32.60	0.933	0.082	30.96	0.953	0.081	29.25	0.970	0.060	8h	<1
TensoIR	35.18	0.976	0.040	27.95	0.896	0.159	34.81	0.983	0.029	5h	4
GS-IR	34.80	0.960	0.047	26.98	0.874	0.152	32.95	0.928	0.054	0.4h	189
R3DG	37.15	0.981	0.024	27.30	0.922	0.121	38.54	0.988	0.016	1h	16
3DGS-DR	38.15	0.979	0.031	32.03	0.960	0.084	39.80	0.987	**0.015**	0.4h	271
GShader	37.13	0.982	0.023	30.87	0.953	0.088	36.02	0.989	0.017	1h	65
Ours	**39.17**	**0.985**	**0.021**	**33.99**	**0.971**	**0.061**	**39.81**	**0.990**	0.016	0.5h	133
Ours(HR)	41.39	0.988	0.017	35.24	0.975	0.055	40.49	0.991	0.014	0.5h	96

Table 2: Relighting quality is evaluated on the TensoIR, Shiny Blender, and Stanford ORB datasets.

Methods	TensoIR			Shiny Blender			Stanford ORB		
	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
TensoIR	28.55	**0.945**	0.080	22.30	0.842	0.184	26.22	0.947	0.049
GShader	26.86	0.930	0.063	19.20	0.874	0.131	26.23	0.952	0.043
GS-IR	25.98	0.897	0.092	21.18	0.846	0.160	28.44	0.960	0.038
R3DG	28.52	0.931	0.069	20.69	0.869	0.141	27.88	0.957	0.039
Ours	**30.10**	0.944	**0.053**	**26.16**	**0.928**	**0.084**	**28.93**	**0.967**	**0.029**

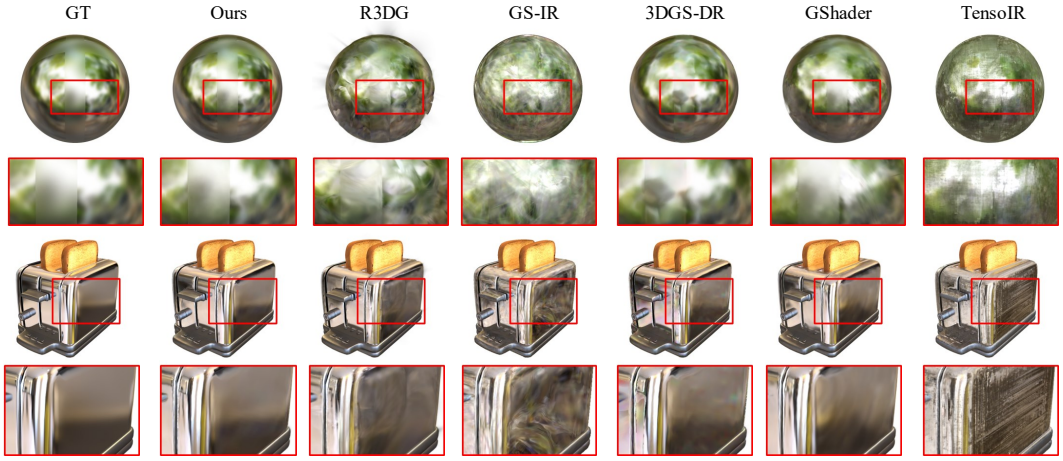

Figure 5: Qualitative comparisons on a synthetic dataset. Our method retains more details, particularly in specular regions.

ORB dataset [27], we evaluate the performance of novel view synthesis and relighting using PSNR, SSIM [48], and LPIPS [63] metrics. For the ball object in the Shiny Blender dataset, only qualitative results are provided due to the absence of relighting ground truth (GT). In addition, we use mean angular error (MAE) to evaluate the quality of normal estimation. In addition, we have also provided the results of training duration and inference speed (FPS). We further evaluate novel view synthesis on the Ref-Real [46] and MipNeRF-360 [3] datasets. **Numbers** in bold represent the best performance, while underscored numbers indicate the second-best performance.

Methods for Comparison. We compared the quality of novel view synthesis against several NeRF-based methods [23, 32] and 3DGS-based methods [16, 22, 31, 55]. In addition, we evaluated the relighting quality between different inverse rendering methods. All methods were implemented and trained using their publicly available code and default configurations.

4.2 Comparison with previous works

Novel view synthesis. Table 1 presents the quantitative comparison results for novel view synthesis (NVS) on object-level datasets. Our PBR results show clear advantages over other methods. Additionally, we provide our Hybrid Rendering (HR) branch results to demonstrate the effectiveness of the hybrid rendering model. Visual comparisons are provided in Figure 5. Notably, our method preserves stable geometric structures even with high-frequency surface variations, producing clearer and more accurate novel views.

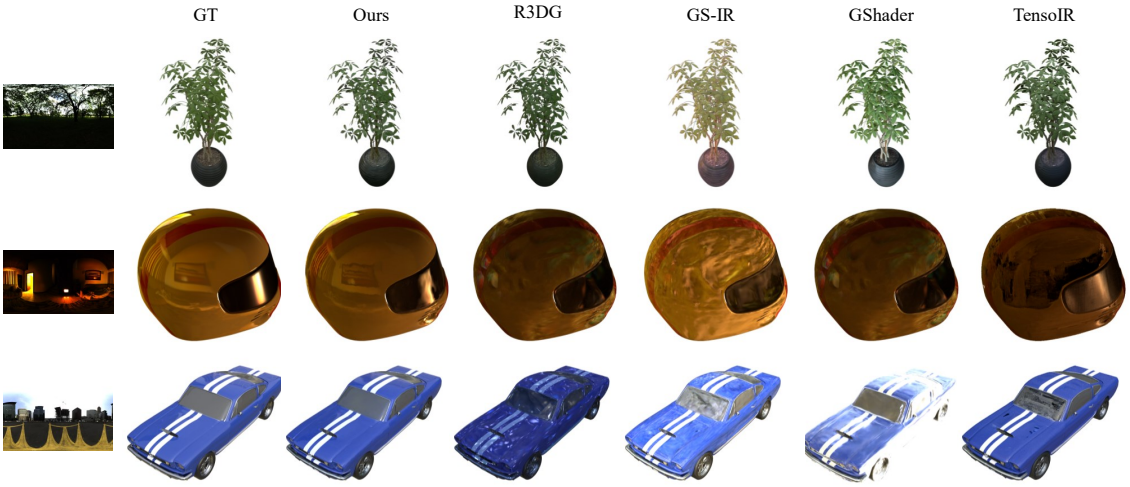

Figure 6: Qualitative comparisons of relighting with different environment lighting conditions.

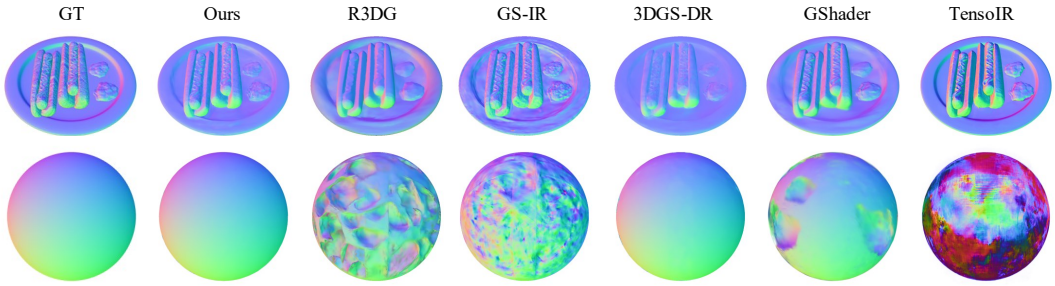

Figure 7: Qualitative comparisons of normal produced by different methods. Our method provides robust normal estimation.

Furthermore, Table 3 presents our results on the Ref-Real dataset [46] and the Mip-NeRF 360 dataset [3], where our method achieves competitive quantitative results.

Relighting. Table 2 presents the results of the relighting comparison. For the TensoIR and Shiny Blender datasets, albedo is aligned to the ground truth via channel-wise scaling before relighting as described in [27, 62]. For the Stanford ORB dataset, albedo scaling is disabled to more accurately evaluate absolute decomposition performance on real objects. Results for the TensoIR and Shiny Blender datasets are averaged over all viewpoints under five different environment maps. For the Stanford ORB dataset, relighting is evaluated using the provided 20 image-environment map pairs. Visual comparisons are provided in Figure 6. Our method's superior detail preservation and effectively suppresses aliasing artifacts in both albedo and lighting, leading to more realistic and visually consistent relighting results. Notably, our approach maintains credibility under different relighting conditions, without significant surface artifacts appearing on either rough or smooth objects.

Normal and materials estimation. Table 4 and Figure 7 present the results of our normal estimation. Notably, in the presence of high-frequency surface details, our method effectively prevents surface discontinuities caused by floating artifacts. In Figure 9, we visualize the estimated albedo, metallic, roughness, normal, and environmental lighting components. Our framework successfully decomposes both diffuse and specular objects. For specular objects, we achieve high-quality decomposition results with clearer environmental lighting. Additional albedo estimation results and more qualitative comparisons are provided in the supplementary materials.

Table 3: Novel view synthesis quality evaluated using PSNR, SSIM, and LPIPS on the Ref-Real dataset and the Mip-NeRF 360 dataset.

Methods	Ref-Real			Mip-NeRF 360		
	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
GS-IR	23.41	0.606	**0.297**	<u>26.18</u>	<u>0.801</u>	**0.200**
GShader	21.13	0.578	0.375	22.33	0.577	0.329
3DGS-DR	<u>23.51</u>	**0.638**	0.343	25.14	0.783	0.304
Ours	**23.54**	<u>0.627</u>	<u>0.337</u>	**26.65**	**0.806**	<u>0.233</u>

4.3 Ablation Study

We specifically evaluated the effectiveness of radiance transfer compared to spherical harmonics. Additionally, we performed ablation

Table 4: Normal estimation quality with Gaussian-based methods evaluated using MAE↓ on the TensoIR dataset and the Shiny Blender dataset.

	GS-IR	R3DG	3DGS-DR	GShader	Ours
TensoIR	5.313	5.914	5.728	**5.303**	5.347
Shiny Blender	9.328	9.238	3.632	4.800	**3.091**

Table 5: Ablation study of key components on the Shiny Blender dataset. "w/o radiance transfer" represents using SHs to calculate the radiance part in hybrid rendering. "Propagation" denotes simplified normal propagation. "Frozen geometry" indicates freezing geometry attributes during decomposition. "w/o hybrid rendering" refers to disabling the hybrid rendering branch during decomposition.

Ablations	NVS PSNR↑	Relighting PSNR↑
ours	**33.99**	**26.16**
w/o radiance transfer	32.15	25.85
w/o propagation	33.26	26.09
w/o $\mathcal{L}_m$	33.76	25.88
w/ frozen geometry	31.49	24.66
w/o hybrid rendering	32.90	25.18

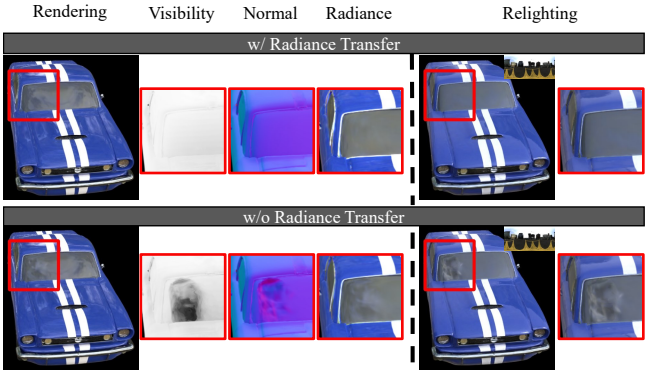

Figure 8: Radiance transfer can more effectively separate low-frequency components of appearance, thereby preventing artifacts caused by overfitting. These artifacts compromise geometric smoothness and degrade the quality of rendering and relighting.

studies on simplified normal propagation to validate the contribution of our proposed components. We also evaluate the impact of the metal reflection prior introduced in Sec. 3.5. For decomposition process, we further conducted experiments of using fixed geometric parameters and disabling the hybrid rendering branch (i.e., using only the PBR branch) during appearance decomposition, to demonstrate the advantages of our dual-branch rendering framework.

Analysis on radiance transfer. As illustrated in Figure 8, using radiance transfer instead of spherical harmonics to represent the radiance component in hybrid rendering reduces floating artifacts

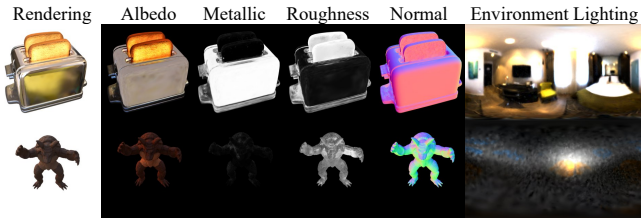

Figure 9: Normal, albedo, roughness, metallic and environment lighing results on synthetic dataset.

and prevents normal and visibility errors caused by local geometric inaccuracies, particularly for specular objects. These improvements significantly enhance the quality of relighting. As shown in Table 5, radiance transfer also leads to notable improvements in quantitative results.

Analysis on decomposition process. When decomposing the appearance, we simultaneously enable hybrid rendering and PBR to fine-tune the geometry, making it compatible with both rendering models. We also evaluate the effects of freezing geometric parameters or enabling only the PBR branch, which demonstrates the limitations of single-branch approaches. As shown in Table 5, both frozen geometry and enabling the PBR branch only lead to significant quality degradation. The former occurs because the geometric structure required for hybrid rendering does not fully meet PBR's requirements, while the latter leads to geometric mutations, rendering the baked occlusion ineffective.

Limitation We assume that lighting originates from an infinite distance, which differs from actual lighting conditions in large-scale scenes. Additionally, our method does not consider more complex indirect lighting effects, such as inter-reflections. These limitations are shown in Figure 10.

Figure 10: Limitation of our method.

5 CONCLUSIONS

We introduce RTR-GS, an inverse rendering framework that enables realistic novel view synthesis and relighting through Gaussian splatting and deferred rendering. By separating high-frequency and low-frequency appearances using reflection maps and radiance transfer, we achieve high-quality hybrid rendering and normal estimation. Building on this, we further decompose material and lighting from the appearance by an additional PBR branch. Experimental results demonstrate that our method delivers competitive performance in novel view synthesis and relighting across various objects. In the future, we aim to explore more precise rendering techniques and incorporate more complex secondary lighting effects.

REFERENCES

[1] Dejan Azinovic, Tzu-Mao Li, Anton Kaplanyan, and Matthias Nießner. 2019. Inverse path tracing for joint material and lighting estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2447–2456.

[2] Jonathan T Barron, Ben Mildenhall, Matthew Tancik, Peter Hedman, Ricardo Martin-Brualla, and Pratul P Srinivasan. 2021. Mip-nerf: A multiscale representation for anti-aliasing neural radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 5855–5864.

[3] Jonathan T Barron, Ben Mildenhall, Dor Verbin, Pratul P Srinivasan, and Peter Hedman. 2022. Mip-nerf 360: Unbounded anti-aliased neural radiance fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 5470–5479.

[4] Jonathan T Barron, Ben Mildenhall, Dor Verbin, Pratul P Srinivasan, and Peter Hedman. 2023. Zip-nerf: Anti-aliased grid-based neural radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 19697–19705.

[5] Sai Bi, Zexiang Xu, Kalyan Sunkavalli, Miloš Hašan, Yannick Hold-Geoffroy, David Kriegman, and Ravi Ramamoorthi. 2020. 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.

[6] Sai Bi, Zexiang Xu, Kalyan Sunkavalli, David Kriegman, and Ravi Ramamoorthi. 2020. 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.

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

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

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

[10] Ang Cao and Justin Johnson. 2023. Hexplane: A fast representation for dynamic scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 130–141.

[11] Eric R Chan, Marco Monteiro, Petr Kellnhofer, Jiajun Wu, and Gordon Wetzstein. 2021. pi-gan: Periodic implicit generative adversarial networks for 3d-aware image synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 5799–5809.

[12] Anpei Chen, Zexiang Xu, Andreas Geiger, Jingyi Yu, and Hao Su. 2022. Tensorf: Tensorial radiance fields. In *European Conference on Computer Vision*. Springer, 333–350.

[13] Hao Chen, Bo He, Hanyu Wang, Yixuan Ren, Ser Nam Lim, and Abhinav Shrivastava. 2021. Nerv: Neural representations for videos. *Advances in Neural Information Processing Systems* 34 (2021), 21557–21568.

[14] Euntae Choi and Sungjoo Yoo. 2024. Phys3DGS: Physically-based 3D Gaussian splatting for inverse rendering. *arXiv preprint arXiv:2409.10335* (2024).

[15] Sara Fridovich-Keil, Giacomo Meanti, Frederik Rahbæk Warburg, Benjamin Recht, and Angjoo Kanazawa. 2023. K-planes: Explicit radiance fields in space, time, and appearance. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 12479–12488.

[16] Jian Gao, Chun Gu, Youtian Lin, Zhihao Li, Hao Zhu, Xun Cao, Li Zhang, and Yao Yao. 2025. Relightable 3D Gaussians: realistic point cloud relighting with BRDF decomposition and ray tracing. In *European Conference on Computer Vision*. Springer, 73–89.

[17] Kaiwen Guo, Peter Lincoln, Philip Davidson, Jay Busch, Xueming Yu, Matt Whalen, Geoff Harvey, Sergio Orts-Escolano, Rohit Pandey, Jason Dourgarian, et al. 2019. The relightables: Volumetric performance capture of humans with realistic relighting. *ACM Transactions on Graphics (ToG)* 38, 6 (2019), 1–19.

[18] Yijia Guo, Yuanxi Bai, Liwen Hu, Ziyi Guo, Mianzhi Liu, Yu Cai, Tiejun Huang, and Lei Ma. 2024. PRTGS: Precomputed radiance transfer of gaussian Splats for real-time high-quality relighting. In *Proceedings of the 32nd ACM International Conference on Multimedia*. 5112–5120.

[19] 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.

[20] Peter Hedman, Pratul P Srinivasan, Ben Mildenhall, Jonathan T Barron, and Paul Debevec. 2021. Baking neural radiance fields for real-time view synthesis. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 5875–5884.

[21] Binbin Huang, Zehao Yu, Anpei Chen, Andreas Geiger, and Shenghua Gao. 2024. 2d Gaussian splatting for geometrically accurate radiance fields. In *ACM SIGGRAPH 2024 Conference Papers*. 1–11.

[22] Yingwenqi Jiang, Jiadong Tu, Yuan Liu, Xifeng Gao, Xiaoxiao Long, Wenping Wang, and Yuexin Ma. 2024. Gaussianshader: 3d gaussian splatting with shading functions for reflective surfaces. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 5322–5332.

[23] 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.

[24] Brian Karis and Epic Games. 2013. Real shading in unreal engine 4. *Proc. Physically Based Shading Theory Practice* 4, 3 (2013), 1.

[25] 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.

[26] Simin Kou, Fang-Lue Zhang, Jakob Nazarenus, Reinhard Koch, and Neil A Dodgson. 2025. OmniPlane: A recolorable representation for dynamic scenes in omnidirectional videos. *IEEE Transactions on Visualization and Computer Graphics* (2025).

[27] Zhengfei Kuang, Yunzhi Zhang, Hong-Xing Yu, Samir Agarwala, Elliott Wu, Jiajun Wu, et al. 2023. Stanford-orb: a real-world 3d object inverse rendering benchmark. *Advances in Neural Information Processing Systems* 36 (2023), 46938–46957.

[28] 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 (TOG)* 43, 4 (2024), 1–13.

[29] Zhaoshuo Li, Thomas Müller, Alex Evans, Russell H Taylor, Mathias Unberath, Ming-Yu Liu, and Chen-Hsuan Lin. 2023. Neuralangelo: High-fidelity neural surface reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 8456–8465.

[30] Zhihao Liang, Hongdong Li, Kui Jia, Kailing Guo, and Qi Zhang. 2024. GUS-IR: Gaussian splatting with unified shading for inverse rendering. *arXiv preprint arXiv:2411.07478* (2024).

[31] 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.

[32] 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.

[33] Tao Lu, Mulin Yu, Linning Xu, Yuanbo Xiangli, Limin Wang, Dahua Lin, and Bo Dai. 2024. Scaffold-gs: Structured 3d gaussians for view-adaptive rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 20654–20664.

[34] Guan Luo, Tian-Xing Xu, Ying-Tian Liu, Xiao-Xiong Fan, Fang-Lue Zhang, and Song-Hai Zhang. 2024. 3D Gaussian editing with a single image. In *Proceedings of the 32nd ACM International Conference on Multimedia*. 6627–6636.

[35] Xiaoyang Lyu, Yang-Tian Sun, Yi-Hua Huang, Xiuzhe Wu, Ziyi Yang, Yilun Chen, Jiangmiao Pang, and Xiaojuan Qi. 2024. 3dgsr: Implicit surface reconstruction with 3d gaussian splatting. *arXiv preprint arXiv:2404.00409* (2024).

[36] 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.

[37] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. 2022. Instant neural graphics primitives with a multiresolution hash encoding. *ACM Transactions on Graphics (TOG)* 41, 4 (2022), 1–15.

[38] 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*. 8280–8290.

[39] Jakob Nazarenus, Simin Kou, Fang-Lue Zhang, and Reinhard Koch. 2024. Arbitrary optics for Gaussian splatting using space warping. *Journal of Imaging* 10, 12 (2024), 330.

[40] Ben Poole, Ajay Jain, Jonathan T Barron, and Ben Mildenhall. 2022. Dreamfusion: Text-to-3d using 2d diffusion. *arXiv preprint arXiv:2209.14988* (2022).

[41] 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.

[42] Johannes L Schonberger and Jan-Michael Frahm. 2016. Structure-from-motion revisited. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 4104–4113.

[43] Yahao Shi, Yanmin Wu, Chenming Wu, Xing Liu, Chen Zhao, Haocheng Feng, Jingtuo Liu, Liangjun Zhang, Jian Zhang, Bin Zhou, et al. 2023. Gir: 3d gaussian inverse rendering for relightable scene factorization. *arXiv preprint arXiv:2312.05133* (2023).

[44] Peter-Pike Sloan, Jan Kautz, and John Snyder. 2023. Precomputed radiance transfer for real-time rendering in dynamic, low-frequency lighting environments. In *Seminal Graphics Papers: Pushing the Boundaries, Volume 2*. 339–348.

[45] Cheng Sun, Min Sun, and Hwann-Tzong Chen. 2022. Direct voxel grid optimization: Super-fast convergence for radiance fields reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 5459–5469.

[46] 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.

[47] 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. *arXiv preprint arXiv:2106.10689* (2021).

[48] 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.

[49] Zhengyi Wang, Cheng Lu, Yikai Wang, Fan Bao, Chongxuan Li, Hang Su, and Jun Zhu. 2024. Prolificdreamer: High-fidelity and diverse text-to-3d generation with variational score distillation. *Advances in Neural Information Processing Systems* 36 (2024).

[50] Tong Wu, Jiamu Sun, Yukun Lai, Yuewen Ma, Leif Kobbelt, and Lin Gao. 2024. DeferredGS: Decoupled and editable Gaussian splatting with deferred shading. *arXiv preprint arXiv:2404.09412* (2024).

[51] Ziyi Yang, Yanzhen Chen, Xinyu Gao, Yazhen Yuan, Yu Wu, Xiaowei Zhou, and Xiaogang Jin. 2023. Sire-ir: Inverse rendering for brdf reconstruction with shadow and illumination removal in high-illuminance scenes. *arXiv preprint arXiv:2310.13030* (2023).

[52] Ziyi Yang, Xinyu Gao, Yangtian Sun, Yihua Huang, Xiaoyang Lyu, Wen Zhou, Shaohui Jiao, Xiaojuan Qi, and Xiaogang Jin. 2024. Spec-gaussian: Anisotropic view-dependent appearance for 3d gaussian splatting. *arXiv preprint arXiv:2402.15870* (2024).

[53] 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.

[54] Lior Yariv, Jiatao Gu, Yoni Kasten, and Yaron Lipman. 2021. Volume rendering of neural implicit surfaces. *Advances in Neural Information Processing Systems* 34 (2021), 4805–4815.

[55] Keyang Ye, Qiming Hou, and Kun Zhou. 2024. 3d gaussian splatting with deferred reflection. In *ACM SIGGRAPH 2024 Conference Papers*. 1–10.

[56] Keyang Ye, Qiming Hou, and Kun Zhou. 2024. Progressive radiance distillation for inverse rendering with Gaussian splatting. *arXiv preprint arXiv:2408.07595* (2024).

[57] Zehao Yu, Anpei Chen, Binbin Huang, Torsten Sattler, and Andreas Geiger. 2024. Mip-splatting: Alias-free 3d gaussian splatting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 19447–19456.

[58] Zehao Yu, Torsten Sattler, and Andreas Geiger. 2024. Gaussian opacity fields: Efficient and compact surface reconstruction in unbounded scenes. *arXiv preprint arXiv:2404.10772* (2024).

[59] Yu-Jie Yuan, Xinyang Han, Yue He, Fang-Lue Zhang, and Lin Gao. 2024. Munerf: Robust makeup transfer in neural radiance fields. *IEEE Transactions on Visualization and Computer Graphics* (2024).

[60] Dingxi Zhang, Yu-Jie Yuan, Zhuoxun Chen, Fang-Lue Zhang, Zhenliang He, Shiguang Shan, and Lin Gao. 2024. Stylizedgs: Controllable stylization for 3d gaussian splatting. *arXiv preprint arXiv:2404.05220* (2024).

[61] Jingyang Zhang, Yao Yao, Shiwei Li, Jingbo Liu, Tian Fang, David McKinnon, Yanghai Tsin, and Long Quan. 2023. Neilf++: Inter-reflectable light fields for geometry and material estimation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 3601–3610.

[62] Kai Zhang, Fujun Luan, Qianqian Wang, Kavita Bala, and Noah Snavely. 2021. 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.

[63] 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.

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

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

[66] Zuo-Liang Zhu, Beibei Wang, and Jian Yang. 2024. Gs-ror: 3d gaussian splatting for reflective object relighting via sdf priors. *arXiv preprint arXiv:2406.18544* (2024).