Occlusion-Aware 3D Gaussian Splatting for Real-Time Inverse Rendering

Likun Gao[1], Yijun Feng[1], Shibang Xiao[1], Xiaohui Liang[1,2*]
[1]State Key Laboratory of Virtual Reality Technology and Systems, Beihang University, Beijing, China
[2]Zhongguancun Laboratory, Beijing, China
gaolikun@buaa.edu.cn

Abstract

Inverse rendering from multiview photographs is an ill-posed problem because it must jointly disentangle geometry, reflectance, and illumination. Existing 3DGS pipelines either optimize all factors in a single stage, failing to model view-dependent occlusion, or split the task into multiple stages, forfeiting end-to-end optimality. We present a one-stage 3DGS inverse-rendering framework that recovers geometry, BRDFs and view-dependent occlusion in real time. Each Gaussian stores 4th-order spherical-harmonic visibility coefficients learned from splat transparencies, eliminating ray tracing. A prototype-based material prior decouples lighting from reflectance, and deferred split–sum shading keeps the pipeline fully differentiable while enabling HDR relighting. Experiments on indoor and outdoor benchmarks demonstrate that our approach improves inverse-rendering objectives, while remaining competitive on novel-view synthesis compared to strong reconstruction baselines such as 3DGS.

Keywords: Inverse rendering, 3D Gaussian Splatting, BRDF, prototype, occlusion

1. Introduction

Inverse rendering from multiview photographs is an ill-posed task that must recover geometry, materials and lighting simultaneously. Accurate estimates of these factors enable editable digital twins for film and game production, photorealistic augmented reality overlays, and physics-aware scene understanding for robotics. Because countless combinations of shape, reflectance, and lighting can reproduce the same imagery, success hinges on accurate modeling of the lighting process, precise geometric recovery, and well-chosen priors on illumination and materials.

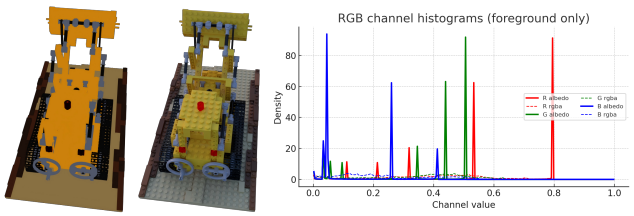
A breakthrough came with Neural Radiance Fields (NeRF) [26], which showed that a continuous volumetric multilayer perceptron (MLP) trained on multiview RGB images can synthesize novel views while implicitly inferring scene structure. Unfortunately, NeRF requires hundreds of network evaluations along every camera ray, making interactive optimization prohibitively slow. To bypass this bottleneck, researchers turned to explicit raster-friendly alternatives, the most influential being 3D Gaussian Splatting (3DGS) [16]. By representing a scene as anisotropic Gaussian primitives that are both differentiable and directly rasterisable, 3DGS attains NeRF-level fidelity with orders-of-magnitude faster training and rendering, making real-time inverse rendering feasible for the first time [6, 34, 19].

However, speed alone does not solve the challenge of decomposition. One-stage pipelines that jointly optimize geometry, materials, and lighting struggle to capture direction-dependent occlusion—effects that traditionally rely on costly ray tracing or offline light baking and therefore integrate poorly with end-to-end learning. Multi-stage strategies alleviate these issues by first fitting geometry with vanilla 3DGS, then baking occlusion offline (or ray tracing), and finally separating lighting from materials; yet this staging prevents the three factors from converging to a globally optimal solution. Although recent methods such as GS-IR [20] and GaussianShader [13] avoid per-ray MLP evaluations, they still cannot accurately model occlusion within a truly end-to-end pipeline, which limits the fidelity of the recovered lighting, material, and geometry components.

Equally problematic is the treatment of material–lighting disentanglement. In real scenes, surface reflectance is not a smoothly varying field but a piecewise constant, cluster-like distribution corresponding to a small set of canonical materials (wood, plastic, brushed metal, etc.). Figure 1 illustrates this: the RGB image (left) contains a wide range of colors, yet its measured BRDF map (right) collapses into just a handful of tight clusters in albedo–roughness space. The current 3DGS pipelines model every gaussian BRDF as an independent continuous parameter; without a discrete prior, optimization drifts, reflectance errors bleed into the lighting term, and the final decomposition deteriorates.

We present an end-to-end 3DGS framework that retains full joint optimization while explicitly modeling directional ambient occlusion: each Gaussian stores a compact spherical-harmonic vector of visibility coefficients that

*Corresponding author.



(a) Albedo (b) RGBA (c) RGB histograms

Figure 1. **Albedo exhibits a highly concentrated color distribution.** The albedo curves show a few sharp peaks and low variance, indicating clustered reflectance values.

captures its orientation-dependent exposure to environment lighting, thereby providing differentiable ambient occlusion without any ray tracing or offline baking. To stabilize this visibility signal, we introduce an occlusion loss, treating the per-gaussian transmittance generated during splatting as a pseudo-label so that every Gaussian learns its relative occlusion state in the scene. Complementing this, we impose a prototype-driven reflectance prior: every albedo is expressed as a mixture of K learnable prototypes plus a sparse residual, anchoring reflectance estimates and preventing leakage into the lighting term. An unsupervised material-aggregation loss pushes the mixture weights to collapse toward a single prototype, while an ℓ_1 sparsity loss keeps the residual minimal. Together, these contributions enable accurate, real-time inverse rendering of geometry, materials, and lighting within a single optimization loop.

Our contributions are as follows.

- Devise a per-Gaussian spherical-harmonic visibility field supervised by an occlusion pseudo-label, enabling fully differentiable, direction-aware occlusion.
- Introduce a prototype-driven reflectance prior that anchors each Gaussian's BRDF to a learnable dictionary, improving reflectance decomposition.
- Demonstrate on synthetic and real benchmarks that the proposed framework outperforms existing 3DGS inverse renderers in novel-view synthesis and relighting tasks.

2. Related Work

2.1. Neural Representation and Rendering

Mildenhall et al. introduced Neural Radiance Fields (NeRF) [26], pioneering a novel paradigm for representing 3D scenes implicitly using multilayer perceptrons (MLPs). NeRF encodes a scene as a continuous volumetric radiance field function, mapping 5D input vectors consisting of spatial coordinates and viewing directions to output volume density and view-dependent radiance. This implicit coordinate-to-radiance representation, supervised by differentiable volume rendering from multi-view images, enables the synthesis of photo-realistic novel views, significantly outperforming prior approaches. Variants now tackle dynamic scenes (e.g., Nerfies [29], HyperNeRF [30], 4D Gaussian Splatting [42]), high-fidelity 3D/4D reconstruction [24, 32, 21, 10], large-scale outdoor captures (Mega-NeRF [38], Block-NeRF [37]), and generative content creation (DreamFusion [31], Neuralangelo [18]).

However, this approach incurs substantial computational overhead: rendering each ray typically requires hundreds of queries to an MLP, resulting in lengthy training and inference times for NeRF. To address efficiency concerns, numerous subsequent studies have explored more effective neural representations and acceleration strategies, including explicit discrete representations and more compact network structures. Anti-aliasing cone tracing in Mip-NeRF [1] cuts queries, while fully *explicit* radiance grids (Plenoxels [8], DirectVoxGO [36]) slash training from hours to minutes. **Instant-NGP** [27] couples a tiny MLP with a multi-resolution hash grid, achieving sub-minute optimisation. Factorised fields such as **TensoRF** [5] compress the 4D radiance tensor into rank-one planes and vectors, converging within mere minutes. Nevertheless, all of these *ray-march* through space, hindering real-time deployment.

In recent years, researchers have begun exploring novel scene representations beyond the implicit fields introduced by NeRF. Among these, the most notable is **3D Gaussian Splatting (3DGS)** [16]. Unlike NeRF, which models scenes implicitly as continuous density fields and renders via ray integration, 3DGS explicitly represents scenes as sets of spatially distributed 3D Gaussian primitives (Gaussian spheres or ellipsoids). These Gaussians are directly projected ("splatted") onto the image plane through rasterization, enabling real-time rendering while maintaining NeRF-level visual quality. Subsequent work improves spatial filtering (GS++ [12]), mobile deployment (Rtgs [22]), city-scale capture (Street-Gaussians [44]), and animatable avatars (3dgs-avatar [33]). The *explicit yet differentiable* nature of Gaussians bridges photorealistic view synthesis with classical raster graphics, laying the foundation for the inverse-rendering pipelines reviewed next.

2.2. Inverse Rendering

Inverse rendering seeks to recover a scene's *geometry, BRDFs, and illumination* from photos, a profoundly ill-posed task that classic set-ups address via controlled lighting [7, 25, 46]. Neural methods instead embed the rendering equation in a learned optimisation loop. The earliest NeRF-style solutions, **NRF** [3] and **NeRD** [4], augment the radiance MLP to emit normals and BRDFs, then Monte-Carlo-integrate reflected light. NeRFactor [48] factorises view-dependent colour into editable BRDF lobes plus visibility, while NeRV [35] and InvRender [49] learn an explicit *visibility field* to cast soft shadows. **PhySG** [47] replaces

Monte-Carlo estimators with analytic spherical-Gaussian integration, and **TensoIR** [14] extends tensor factorisation to unify radiance-field training with online one-bounce global illumination, finishing full object-level decomposition in hours.

Ray marching still throttles real-time feedback. 3D Gaussian Splatting (3DGS) has emerged as an explicit alternative for scene representation, offering drastic speed-ups in rendering. Considering the characteristics of 3DGS, its inverse rendering tasks can be divided into two categories: single-step and multi-step. For single-step inverse rendering based on 3DGS, geometry, BRDFs, and illumination are jointly optimized to estimate the scene's material structure and lighting. The advantage of this approach lies in the collaborative optimization of multiple interdependent variables, leading to better scene representation and more accurate estimation of scene parameters. For example, GaussianShader [13] associates each Gaussian with a normal (its shortest principal axis) and a micro-facet BRDF, achieving glossy relighting. Recent work has introduced variants of 3D Gaussian splatting that extend inverse rendering and relighting capabilities. DeferredGS decouples geometry, material, and illumination using deferred shading and normal field distillation, enabling editable scene decomposition and relighting with improved fidelity [43]. ReCap treats cross-environment captures as a multi-task supervision signal by jointly optimizing multiple lighting representations sharing a common material representation, addressing albedo–lighting ambiguities in relighting tasks [17].

For multi-step 3DGS inverse rendering, the primary workflow involves first using the original 3DGS to estimate scene structure, including Gaussian attributes and normals. These parameters are then fixed to enable physics-based rendering for estimating BRDFs and lighting. The advantage of this approach is that, with the structure fixed, occlusion information and indirect lighting become easier to model, leading to more accurate reconstructions. For example, GS-IR [20] bakes multi-directional shadow volumes for every Gaussian. RelightableGS [9] hybridizes forward splats with lightweight GPU ray tracing to support exact area-light shadows in cluttered indoor scenes. In contrast to baking-based approaches, GI-GS integrates G-buffer-based path tracing into the Gaussian splatting pipeline to dynamically estimate indirect illumination, enabling physically grounded global illumination at the cost of higher computational overhead [41].

3. Preliminaries

3D Gaussian Splatting (3DGS) [16] represents a static scene as a collection of anisotropic 3-D Gaussian primitives. Each primitive $G(\mathbf{x})$ is parameterized by a mean $\boldsymbol{\mu} \in \mathbb{R}^3$, a positive-definite covariance $\boldsymbol{\Sigma} \in \mathbb{R}^{3\times3}$, and a view-dependent color $\mathbf{c}$ encoded with third-order spherical harmonics (SH). To ensure $\boldsymbol{\Sigma}$ stays positive-definite and is amenable to gradient-based optimization, it is factorized as

$$\boldsymbol{\Sigma} = \mathbf{R}\,\mathbf{S}\,\mathbf{S}^\top\,\mathbf{R}^\top, \tag{1}$$

where $\mathbf{S} = \mathrm{diag}(s_x, s_y, s_z)$ specifies the axis lengths and $\mathbf{R} = \mathrm{quat}(q_0, q_1, q_2, q_3)$ is the orientation quaternion. The Gaussian signal

$$G(\mathbf{x}) = \exp\Big(-\tfrac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^\top \boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})\Big) \tag{2}$$

is learned end-to-end by minimizing the photometric error between rendered and reference images.

For rendering, each 3-D Gaussian is projected into screen space via the extrinsic matrix $\mathbf{T}$ and intrinsic matrix $\mathbf{K}$, yielding a 2-D Gaussian $\tilde{G}$ with mean $\tilde{\boldsymbol{\mu}}$ and covariance $\tilde{\boldsymbol{\Sigma}}$. At pixel $\mathbf{u}$, the k-th Gaussian contributes an opacity

$$\alpha_k = \tilde{G}\big(\mathbf{u} \mid \tilde{\boldsymbol{\mu}}_k, \tilde{\boldsymbol{\Sigma}}_k\big)\, o_k, \tag{3}$$

and the cumulative transmittance up to k is

$$T_k = \prod_{j<k}(1-\alpha_j). \tag{4}$$

The final pixel color is then obtained by front-to-back α-blending:

$$\hat{\mathbf{c}}(\mathbf{u}) = \sum_{k=1}^{N} T_k\,\alpha_k\,\mathbf{c}_k. \tag{5}$$

Because the entire splatting–rasterization–shading pipeline is fully differentiable, 3DGS serves as the backbone of our one-step, joint inverse-rendering framework, enabling simultaneous optimisation of geometry and appearance.

3.1. BRDF Rendering and Split-Sum Approximation

In our inverse-rendering pipeline, the outgoing radiance L_o at a point $\mathbf{x}$ on a 3-D Gaussian is given by the rendering equation

$$L_o(\boldsymbol{\omega}_o, \mathbf{x}) = \int_\Omega f\big(\boldsymbol{\omega}_o, \boldsymbol{\omega}_i, \mathbf{x}\big)\, L_i(\boldsymbol{\omega}_i, \mathbf{x})\, (\boldsymbol{\omega}_i \cdot \mathbf{n})\, \mathrm{d}\boldsymbol{\omega}_i, \tag{6}$$

where L_i is the incident illumination from direction $\boldsymbol{\omega}_i$, $\boldsymbol{\omega}_o$ is the view direction, and $\mathbf{n}$ is the surface normal. We adopt the Disney micro-facet BRDF, which splits f into a Lambertian diffuse term $\frac{a(1-m)}{\pi}$ and a GGX-based specular term $\frac{D\,F\,G}{4(\boldsymbol{\omega}_i\mathbf{n})(\boldsymbol{\omega}_o\mathbf{n})}$. Here, a is the albedo, m the metallic factor, and D, F, G denote the normal distribution, Fresnel, and geometry functions, all controlled by roughness r.

Direct evaluation of this integral at every shading point is computationally expensive. To accelerate specular shading, we employ the split–sum approximation [15]. The diffuse component is prefiltered into an irradiance map

$$L_{\mathrm{diff}}(\mathbf{n}) = \int_\Omega L_i(\boldsymbol{\omega}_i)\, (\boldsymbol{\omega}_i \cdot \mathbf{n})\, \mathrm{d}\boldsymbol{\omega}_i, \tag{7}$$

so that

$$c_{\text{diff}} = \frac{a\,(1-m)}{\pi}\, L_{\text{diff}}(\mathbf{n}). \tag{8}$$

For the specular branch, we factor the original integral into two separable terms: one depending only on roughness r, and one on the viewing angle $\boldsymbol{\omega}_o \cdot \mathbf{n}$.

$$c_{\text{spec}} \approx L_{\text{prefilter}}(r, R) \cdot \Big[F_0\, \text{LUT}_x(N \cdot V, r) + \text{LUT}_y(N \cdot V, r)\Big]. \tag{9}$$

where $R = 2(\mathbf{n} \cdot \boldsymbol{\omega}_i)\mathbf{n} - \boldsymbol{\omega}_i$ is the reflection direction, N is the normal map, V is the view direction map and F_0 the base Fresnel reflectance. Both the environment prefilter $L_{\text{prefilter}}$ and the 2D BRDF lookup tables $(\text{LUT}_x, \text{LUT}_y)$ are precomputed via importance-sampled GGX convolution and stored as mip-mapped textures. At runtime, specular shading reduces to a few texture fetches.

Combining diffuse and specular components, each Gaussian's shaded color becomes

$$\hat{\mathbf{c}} = c_{\text{diff}} + c_{\text{spec}}. \tag{10}$$

Thanks to the split–sum approximation's separation of lighting and material response, it integrates seamlessly with 3DGS's differentiable splatting pipeline, enabling efficient joint optimization of geometry, materials, and illumination.

4. Method

Unlike earlier two-stage pipelines that separate geometry from appearance, our method is cast as a single-stage inverse-rendering framework in which geometry, BRDF, visibility and global illumination are optimized jointly. After recovering per-Gaussian albedo with the prototype-driven material prior, we rasterize the scene through a deferred pipeline: normals, albedo, metallic, roughness and spherical-harmonic occlusion coefficients are accumulated into screenspace G-buffers. A physically based split–sum shader then consumes these buffers, together with the estimated environment map, to generate the final image (see Figure 2).

4.1. Learnable Occlusion

Data Representation. To explicitly encode visibility within 3DGS, every Gaussian particle i is extended with a 25-dimensional set of occlusion spherical-harmonic coefficients (occlusion SH).

$$\mathbf{c}^{\text{occ},i} = \{c_{l\mu}^{\text{occ},i}\}_{0 \le l \le 4,\, -l \le \mu \le l}, \tag{11}$$

defined on the real basis $Y_{l\mu}$ up to the 4$^{\text{th}}$ order. These coefficients approximate the visibility function $V_g(\boldsymbol{\omega})$ via

$$V_i(\boldsymbol{\omega}) \;\approx\; \sum_{l\mu} c_{l\mu}^{\text{occ},i}\, Y_{l\mu}(\boldsymbol{\omega}), \tag{12}$$

During inference, all coefficients whose Gaussians fall inside the same screen-space tile are blended using their α-weights to produce $\hat{c}_{l\mu}^{\text{occ}}(\mathbf{x})$, a tile-level descriptor used exclusively for calculating ambient occlusion,

$$V(\boldsymbol{\omega}) \;\approx\; \sum_{l\mu} \hat{c}_{l\mu}^{\text{occ}}\, Y_{l\mu}(\boldsymbol{\omega}), \tag{13}$$

Backward Transmittance Pseudo-labels. During standard 3DGS rasterisation (Eq. 4), each Gaussian i is processed in front-to-back order and contributes an opacity α_i. The accumulated forward transmittance $T_i = \prod_{j<i}(1-\alpha_j)$ is used for color compositing, but it is not a reliable proxy for *environment* visibility: it only accounts for occluders between the camera and i, and implicitly assumes that the half-space behind the camera is empty. In unbounded *scene*-level captures, this assumption can be violated by unknown geometry outside the captured viewing frustum, making forward-direction visibility targets unreliable for global illumination arriving from arbitrary directions.

We therefore exploit the *backward* transmittance

$$\hat{T}_i = \prod_{j>i} \big(1 - \alpha_j\big), \tag{14}$$

which measures how much light can reach i from the opposite direction $\boldsymbol{\omega}_b = -\boldsymbol{\omega}_f$ ($\boldsymbol{\omega}_f$ points from the pixel centre to i). Crucially, $\hat{T}_i \in [0,1]$ depends only on geometry *in front of the camera*, providing a stable, differentiable self-supervision signal aligned with the 3DGS compositing model. From a physical perspective, transmittance directly measures light attenuation along a ray, and therefore serves as a continuous relaxation of binary visibility; in this sense, backward transmittance and directional visibility are physically equivalent quantities under the volumetric rendering model.

A natural alternative is to also supervise visibility along the *forward* direction $\boldsymbol{\omega}_f$, or to impose dual-direction constraints $(\boldsymbol{\omega}_b, \boldsymbol{\omega}_f)$. However, for *scene*-level reconstruction such supervision is generally *not* well-defined: forward-direction transmittance is conditioned on the camera position and the observed frustum, and it may be inconsistent with true environment occlusion when uncaptured/unknown occluders exist behind the sensor or outside the observed views. Consequently, enforcing dual-direction targets can bias the learned visibility field toward camera-specific artifacts rather than global environment visibility.

We adopt $\hat{T}_i$ as a self-supervised pseudo-label ($\hat{p}_i = \hat{T}_i$) and train the occlusion SH coefficients with a binary cross-entropy loss (Fig. 3),

$$\mathcal{L}_{\text{occ}} = -\hat{p}_i \log V(\boldsymbol{\omega}_b) - (1-\hat{p}_i) \log\big(1 - V(\boldsymbol{\omega}_b)\big). \tag{15}$$

Ambient Occlusion. By definition, ambient occlusion at surface point $\mathbf{x}$ is the hemispherical integral of binary

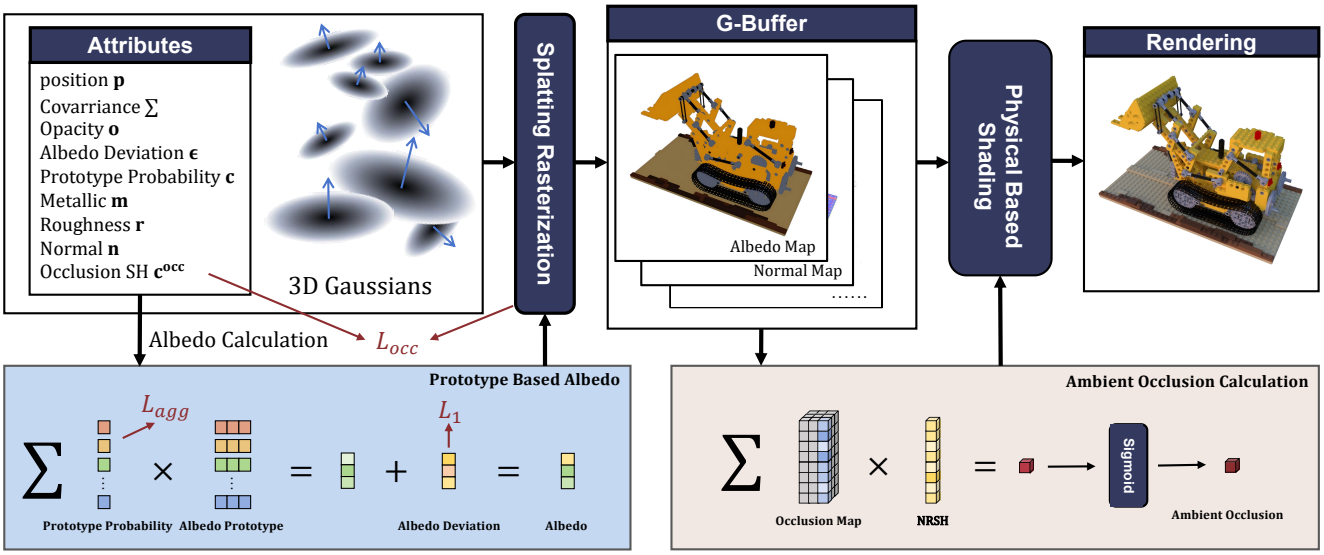


Figure 2. A prototype-driven material prior first expresses each Gaussian albedo as a mixture of learnable prototypes. Together with the Gaussian's metallic, roughness, normals and spherical-harmonic occlusion coefficients, these attributes are splatted once into screen-space G-buffers. A deferred split–sum PBR shader, guided by the learned HDR environment, then fuses the G-buffers to produce the final image.

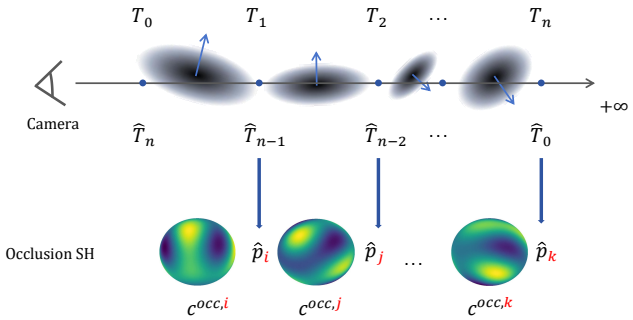


Figure 3. Illustration of the pseudo-label derivation in L_{occ}. The backward transmittance computed during the inverse pass is treated as pseudo-labels to guide the occlusion SH coefficients. The red letters highlight the correspondence between the occlusion SH and their pseudo-labels.

visibility $V(\boldsymbol{\omega})$ weighted by the Lambertian cosine term:

$$AO = \frac{1}{\pi} \int_{\Omega^+} V(\boldsymbol{\omega})\, (\boldsymbol{n}\cdot\boldsymbol{\omega})\, d\boldsymbol{\omega}, \tag{16}$$

where Ω^+ is the hemisphere centred on the surface normal $\boldsymbol{n}$. A naïve Monte-Carlo evaluation of (16) is both noisy and computationally expensive. Instead, we project the integrand onto the real spherical-harmonic (SH) basis up to order 4. For a fixed normal $\boldsymbol{n}$, the coefficients of the cosine kernel $(\boldsymbol{n}\cdot\boldsymbol{\omega})^+$ are predetermined; we pre-compute and store them as

$$c_{l\mu}^{\text{NRSH}}(\boldsymbol{n}), \qquad 0 \le l \le 4,\ -l \le \mu \le l, \tag{17}$$

which we call the Normal-Related SH (NRSH) coefficients. The visibility of each screen pixel $\mathbf{u}$ is obtained by blending the per-Gaussian occlusion SH into $\hat{c}_{l\mu}^{\text{occ}}(\mathbf{u})$.

Because the SH basis is orthonormal, the hemispherical integral reduces to a dot product of the two coefficient sets:

$$AO(\mathbf{u}) = \sum_{l=0}^{4} \sum_{m=-l}^{l} \hat{c}_{l\mu}^{\text{occ}}(\mathbf{u})\, c_{l\mu}^{\text{NRSH}}(\boldsymbol{n}). \tag{18}$$

Equation (18) yields an ambient occlusion estimate with only 25 fused-multiply-add operations per pixel, dramatically faster than sampling-based methods.

Integration with Split–Sum. The learned occlusion factor AO, which quantifies the fraction of incoming light masked by the surrounding geometry, is injected into the Split–Sum shading pipeline as,

$$c_{\text{diff}} = AO \cdot \frac{a\,(1-m)}{\pi}\, L_{\text{diff}}(\mathbf{n}). \tag{19}$$

$$c_{\text{spec}} \approx \big(V(R)\cdot L_{\text{prefilter}}(r,R) + (1-V(R))\cdot L_{\text{indirect}}\big) \cdot \big[F_0\,\text{LUT}_x(N\cdot V, r) + \text{LUT}_y(N\cdot V, r)\big]. \tag{20}$$

Here, L_{indirect} denotes a low-frequency approximation of indirect environment illumination. Due to space limitations, we defer the detailed computation to Appendix.

4.2. Prototype Based Material Representation

Real-world reflectances rarely vary continuously; instead, they tend to cluster into a handful of canonical materials such as wood, plastic, or brushed metal. To capture this discreteness, we endow every Gaussian primitive i with a learnable probability vector $\mathbf{p}_i \in \mathbb{R}^K$, where the k-th entry

p_{ik} denotes the likelihood that the primitive belongs to the k-th *material prototype*. The prototype dictionary itself is a global table $\mathcal{A} = \{\mathbf{A}_k \in \mathbb{R}^3\}_{k=1}^{K}$ of RGB reflectances that is jointly optimized with all other model parameters via back-propagation. The number of prototypes K is pre-defined per dataset to cover material diversity, and we find that even for the most complex scene-level benchmarks a moderate value (at most K=32) is sufficient in practice. All prototype colors are randomly initialized.

Base reflectance (albedo) is therefore no longer stored per primitive; instead, it is reconstructed on the fly as a convex combination of prototypes plus a narrow residual term,

$$a_i = \sum_{d=1}^{D} p_{id}\, \mathbf{A}_d \;+\; \boldsymbol{\varepsilon}_i, \tag{21}$$

where $\boldsymbol{\varepsilon}_i$ is a learnable, per-Gaussian residual that captures subtle deviations from the dominant prototype. The assignment probabilities p_{ik} are treated as learnable parameters constrained to the probability simplex, allowing each Gaussian to softly select among prototypes during optimization. The composite albedo in (21), together with the remaining Gaussian parameters, is fed into the 3DGS rasterization pipeline to produce view-dependent albedo maps that drive the subsequent physically-based rendering stage.

Inspired by sparse losses [23] and probabilistic prototype representations [45], we introduce a *material-aggregation loss* that encourages each Gaussian's albedo to concentrate on a single dominant prototype (the corresponding theoretical proof is provided in the appendix),

$$\mathcal{L}_{\text{agg}} = -\sum_{i} \log\Big(\sum_{d=1}^{D} p_{id} \prod_{j\neq d} (1 - p_{ij})\Big), \tag{22}$$

where the expression inside the logarithm represents the probability that the albedo of Gaussian primitive i originates from one specific prototype. Minimizing $\mathcal{L}_{\text{agg}}$ promotes near one-hot assignments at the per-primitive level, without imposing explicit global constraints on prototype usage. In practice, the geometric and appearance complexity of real scenes naturally prevents degenerate collapse to a small subset of prototypes, as such collapse would incur large reconstruction error.

In addition, we introduce a complementary *residual-sparsity loss*

$$\mathcal{L}_{\text{res}} = \sum_{i} \big\|\boldsymbol{\varepsilon}_i\big\|_1, \tag{23}$$

which encourages the prototypes to explain the dominant, low-frequency color trends, while the residual captures complex or continuous reflectance variations that are not well represented by a small set of prototypes. The prototype prior is a *soft* regularization rather than a hard sparsity constraint: $\boldsymbol{\varepsilon}_i$ provides a flexible escape hatch when real scenes deviate from the ideal clustered-material assumption. The residuals are randomly initialized and gradually suppressed by the ℓ_1 penalty only when they are unnecessary for image reconstruction. Together, Eq. (21)–(23) yield a compact yet flexible material representation that integrates seamlessly with geometry, visibility, and illumination in our one-stage inverse-rendering framework.

4.3. Deferred Shading in Gaussian Splatting

Rather than blending the shaded color of each Gaussian into the frame buffer during rasterization ("forward" shading), we switch to a deferred pipeline. In the first pass, the same differentiable projector used in Eq.5 accumulates, per pixel $\mathbf{u}$, the weighted attributes $C_X(\mathbf{u}) = \sum_i T_i \alpha_i X_i$ with $X \in \{n_i, a_i, m_i, r_i, \mathbf{c}_{l\mu}^{\text{occ},i}\}$, where a is the base albedo, m the metallic factor, r the roughness, and $\mathbf{c}_{l\mu}^{\text{occ}}$ $(0 \le l \le 4)$ are the learned occlusion SH. This produces five screen-space G-buffers $I_n, I_a, I_m, I_r, I_{\text{occ}}$.

In a second pass, each pixel is shaded independently with Eq.(20): diffuse and specular responses are fetched from the split–sum lookup tables using (I_a, I_m, I_r), visibility is reconstructed from I_{occ} and I_n to produce AO, and the result is combined with the learned global-illumination L_{HDR}. Because geometry and material parameters $(\boldsymbol{\mu}, \boldsymbol{\Sigma}, o, a, m, r, \mathbf{c}_{l\mu}^{\text{occ}})$ are now disentangled from lighting, the model relights faithfully under novel HDR maps or point lights while remaining fully differentiable.

4.4. Loss Function and Optimization Strategy

Our one-step training minimizes a composite objective that now balances six aspects: image fidelity, illumination neutrality, spatial smoothness, visibility consistency, material coherence, and prototype compactness. So that geometry, prototype based appearance, lighting, and occlusion converge coherently,

$$\begin{aligned}\mathcal{L} = \; & \lambda_{\text{img}}\mathcal{L}_{\text{img}} + \lambda_l \mathcal{L}_l + \lambda_s \sum_{x\in\{m,r,n\}} \mathcal{L}_s(x) \\ & + \lambda_{\text{occ}}\mathcal{L}_{\text{occ}} + \lambda_{\text{agg}}\mathcal{L}_{\text{agg}} + \lambda_{\text{res}}\mathcal{L}_{\text{res}}\,.\end{aligned} \tag{24}$$

Image-reconstruction term $\mathcal{L}_{\text{img}}$. Weighted average of pixel-wise MAE and perceptual D-SSIM,

$$\mathcal{L}_{\text{img}} = \text{MAE}(I, \hat{I}) + \text{D}-\text{SSIM}(I, \hat{I}). \tag{25}$$

Environmental-light regularizer $\mathcal{L}_l$. Encourages a neutral-white HDR map by penalizing channel deviations:

$$\mathcal{L}_l = \frac{1}{H_l W_l} \sum_{i,j} \big|\, l_{i,j} - \bar{l}_{i,j} \big|. \tag{26}$$

Edge-aware smoothness term $\mathcal{L}_s(x)$. Applied only to metallic m, roughness r, and normal n, suppresses speckles

Table 1. Quantitative comparison on the TensoIR Synthetic dataset. Higher is better for PSNR and SSIM, lower is better for LPIPS and MAE. Best results are **bold**; second best are underlined.

Method	Normal	Novel-View Synthesis			Relight			Family
	MAE ↓	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓	
NeRFactor [48]	6.314	24.679	0.922	0.120	23.383	0.908	0.131	NeRF
InvRender [49]	5.074	27.367	0.934	0.089	23.973	0.901	0.101	NeRF
NVDiffrec [28]	6.078	30.696	0.962	0.052	19.880	0.879	0.104	NeRF
TensoIR [14]	**4.100**	35.088	0.976	0.040	**28.580**	**0.944**	0.081	NeRF
GS-IR [20]	4.948	35.333	0.974	0.039	24.374	0.885	0.096	3DGS
GaussianShader [13]	6.525	37.573	0.984	0.022	23.372	0.874	0.105	3DGS
R3DG [9]	4.728	37.131	0.984	0.022	21.404	0.860	0.127	3DGS
Ours	4.215	**38.049**	**0.986**	**0.020**	27.424	0.923	**0.068**	3DGS

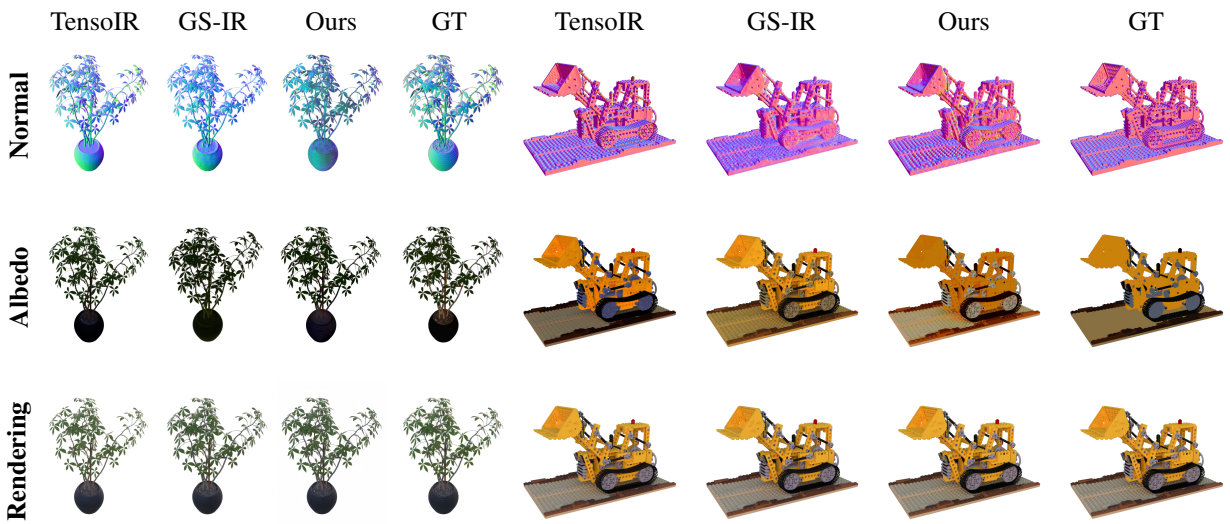


Figure 4. Qualitative comparison on two real objects. Rows show surface **Normal**, estimated **Albedo**, and **Rendering**. Our results are visually closest to the ground-truth (GT) in every category.

while respecting image edges,

$$\mathcal{L}_s(x) = \frac{1}{HW} \sum_{i,j} \left(|\partial_x x_{ij}| \, e^{-|\partial_x I_{ij}|} + |\partial_y x_{ij}| \, e^{-|\partial_y I_{ij}|} \right). \tag{27}$$

5. Experiments

This section follows the structure of the previous work: Metrics and datasets, Comparative evaluation on both synthetic and real-world datasets, and Ablation study. Additional experiments are provided in the appendix.

5.1. Datasets and Metrics

We evaluate on four benchmarks: TensoIR Synthetic [14] and Shiny-Blender for object-level decomposition, and Mip-NeRF360 [2] and Ref-NeRF [39] for scene-level reconstruction. TensoIR Synthetic contains four Blender objects with ground-truth, spatially varying material parameters; Shiny-Blender focuses on specular, view-dependent appearances to stress reflectance modeling. For scenes, Mip-NeRF360 offers real, unbounded 360° captures (indoor/outdoor), while Ref-NeRF adds real scenes with strong specularities. Together, Mip-NeRF360 and Ref-NeRF provide ten publicly released real-world scenes, enabling balanced evaluation across object- and scene-scale settings.

To assess the accuracy of normal reconstruction, we report the mean angular error (MAE) in the TensoIR synthetic dataset. For novel-view synthesis, we evaluate our results on public datasets using three widely adopted image metrics: Peak Signal-to-Noise Ratio (PSNR), Structural Similarity Index (SSIM), and Learned Perceptual Image Patch Similarity (LPIPS). Note that relighting experiment is scored using the identical set of metrics as the rendered views.

Table 2. Performance on three real-world scene benchmarks. The first four methods above the horizontal line are non-inverse-rendering baselines, whereas the methods below the line are 3DGS inverse-rendering approaches.

Method	Mip-NeRF 360 Outdoor			Mip-NeRF 360 Indoor			Ref-NeRF Real		
	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓
Mip-NeRF 360 [2]	24.43	0.694	0.278	31.49	0.918	0.179	24.27	0.649	0.276
UniSDF [40]	24.77	0.723	0.241	31.28	0.901	0.181	23.70	0.635	0.266
3DGS [16]	24.64	0.731	0.234	30.41	0.917	0.190	23.67	0.632	0.288
2DGS [11]	24.33	0.709	0.284	30.39	0.922	0.183	23.65	0.634	0.285
GS-IR[20]	23.45	0.671	0.284	27.80	0.870	0.248	23.32	0.625	0.283
GaussianShader [13]	22.80	0.665	0.297	26.61	0.878	0.243	22.96	0.624	0.294
GUSIR [19]	23.76	0.696	0.276	28.98	0.902	0.222	23.67	0.646	0.282
Ours	**24.54**	**0.728**	**0.231**	**31.17**	**0.916**	**0.195**	**23.81**	**0.656**	**0.280**

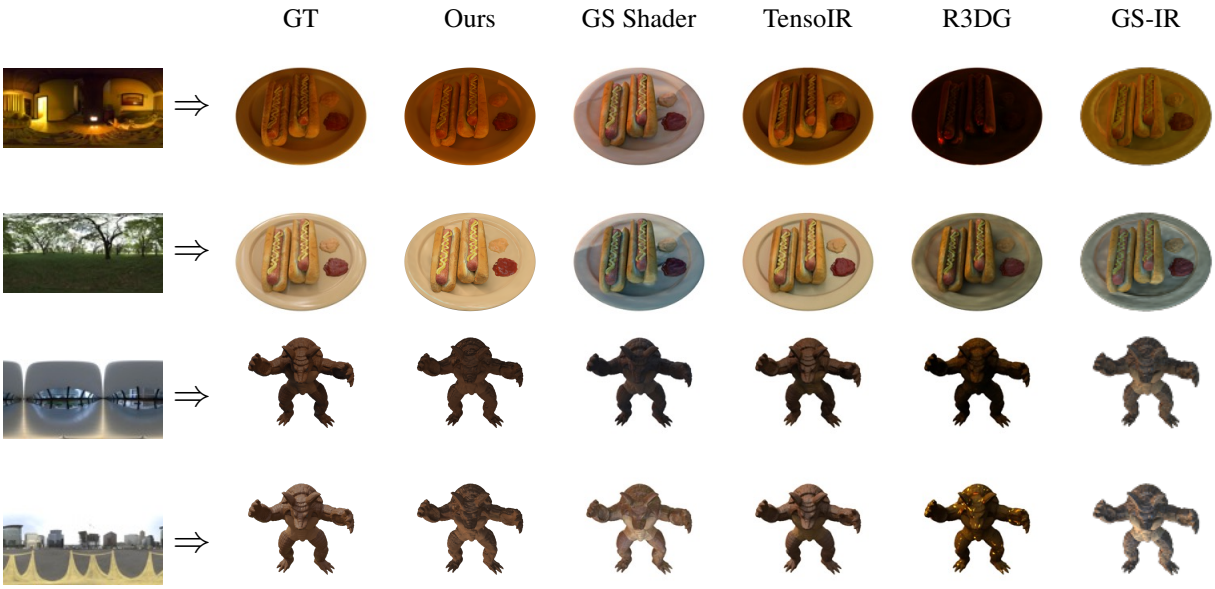


Figure 5. Qualitative relighting comparison under four novel HDR environments (left). Our results are visually closest to the ground truth (GT) across both object categories and illumination conditions.

5.2. Implement Details

Our framework is built on top of the official 3D Gaussian Splatting codebase. Adam is used with learning rates of 2×10^{-3} for geometry and 1×10^{-3} for appearance, decayed by 0.5 every 20 k iterations until convergence at $\approx$ 30 k iterations on one RTX 4090 GPU. We pre-set the prototype number 32 per dataset. Prototypes and residuals are randomly initialized.

5.3. Runtime Performance

Our method augments each Gaussian with additional learnable attributes for inverse rendering. Concretely, we introduce (i) a 4th-order real spherical-harmonic visibility field (25 scalars per Gaussian), (ii) a prototype assignment vector of length K, and (iii) a 3D residual albedo. These additions increase the per-Gaussian storage compared to vanilla 3DGS and incur extra computation during rendering.

To support concrete real-time claims, we report in Table 3 the training time and rendering throughput (FPS) under a fixed hardware and resolution setup. The proposed occlusion term requires additional evaluation and introduces extra splatting-related cost, which reduces FPS compared to some faster 3DGS baselines. Nevertheless, with practical engineering optimizations, our system still runs in real time (51 FPS).

The occlusion field is the dominant contributor to the added storage (25 floats per Gaussian). In practice, this leads to a moderate but non-negligible increase in final model size, representing an accuracy–efficiency trade-off. We measured the peak GPU memory footprint of our full model during training to be 8.7 GB under the same setup.

Method	Novel-View Synthesis			Training Time	FPS
	PSNR↑	SSIM↑	LPIPS↓		
TensoIR [14]	35.088	0.976	0.040	5h	4
GS-IR [20]	35.333	0.974	0.039	**0.4h**	**189**
GaussianShader [13]	37.573	0.984	0.022	1h	65
R3DG [9]	37.131	0.984	0.022	1h	16
Ours	**38.049**	**0.986**	**0.020**	1h	51

Table 3. Comparison on the TensoIR Synthetic dataset in terms of novel-view synthesis quality (PSNR/SSIM/LPIPS), training time, and rendering throughput (FPS) under a fixed hardware/resolution setup.

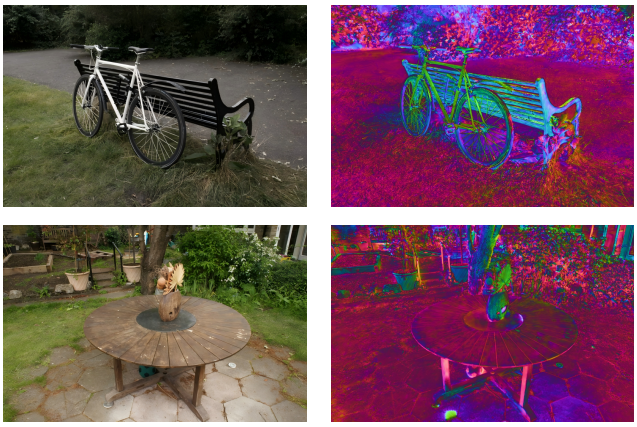


Figure 6. Novel-view synthesis results on two scenes. Left: rendered RGB; Right: predicted surface normals.

Reducing this overhead is an interesting direction for future work. For example, one could replace explicit per-Gaussian SH coefficients with lightweight neural encodings or structured/anchored parameterizations, potentially improving both memory footprint and runtime.

5.4. Comparative Evaluation

We choose 3D Gaussian Splatting (3DGS) baselines that are inverse-rendering/relighting oriented and represent complementary technical routes for shading and visibility. In particular, GS-IR [20] typifies a multi-stage baking/caching strategy that fits a 3DGS scene and then precomputes visibility-related information for relighting; GaussianShader [13] represents a single-stage analytic shading-on-Gaussians approach that attaches physically motivated BRDF terms (with estimated normals) directly to Gaussians while keeping the pipeline fully rasterized; and Relit 3DGS (R3DG) [9] follows a hybrid route that augments 3DGS with additional explicit visibility computation (e.g., ray tracing/BVH) to obtain more accurate shadows during relighting. We additionally include GUSIR [19] as a 3DGS inverse-rendering baseline with unified shading, and report standard 3DGS/2DGS [16, 11] as non-inverse-rendering reconstruction references to contextualize view-synthesis quality.

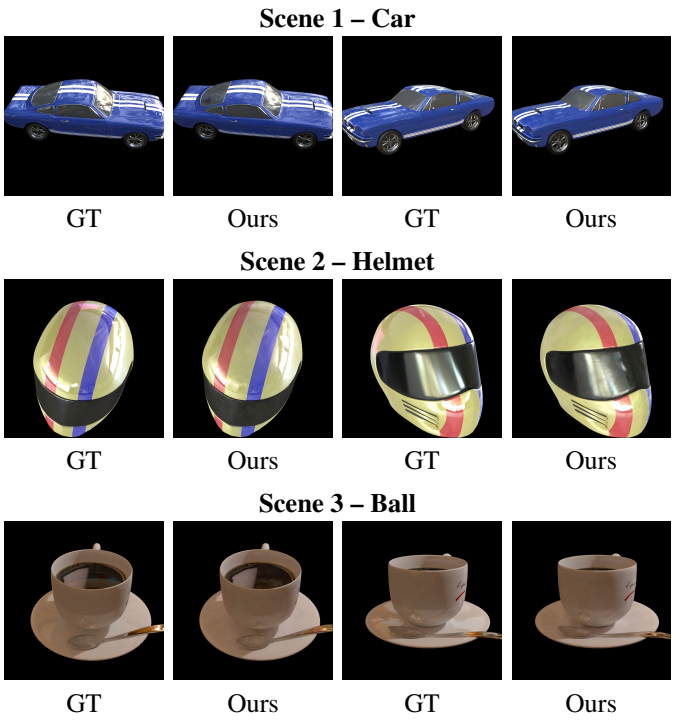


Figure 7. Qualitative novel-view synthesis results on three scenes from the *Shiny-Blender* dataset.

We benchmark our method against recent state-of-the-art inverse-rendering approaches on the TensoIR Synthetic dataset. All methods are trained solely from multi-view photographs captured under unknown illumination. We evaluate surface-normal reconstruction with the mean angular error (MAE), novel-view synthesis and relighting fidelity, and we record training throughput. Following NeRFactor [48] and TensoIR [14], we resolve the global scale ambiguity between illumination and albedo during evaluation by applying a least-squares global scale alignment to the predicted albedo and relighting images before computing PSNR, SSIM, and LPIPS.

As summarized in Table 1, our approach achieves the highest novel-view PSNR, underscoring the advantages of our explicit PBR integration. In addition, our relighting scores consistently surpass those of existing 3DGS-based inverse rendering baselines. Crucially, the convergence time remains on par with other 3DGS pipelines, allowing real-time rendering and demonstrating the practicality of the framework for complex inverse-rendering tasks. The qualitative comparisons in Figure 4 further show that our final renderings are closer to the ground truth than those of the competing methods, and our estimated albedos exhibit clustering characteristics similar to the ground truth. Figure 5 illustrates our relighting results and highlights the advantages of our method over other 3DGS baselines.

Tab. 4 reports peak-signal-to-noise ratio (PSNR↑) on the

Table 4. Novel-view-synthesis PSNR (↑) on the *Shiny-Blender* dataset. Bold marks the best score within each column.

Method	Car	Ball	Helmet	Teapot	Toaster	Coffee	Avg.	**Family**
NVDiffRec	24.63	18.04	27.12	40.13	23.79	30.77	27.41	NeRF
NVDiffMC	25.14	17.35	25.77	37.45	21.82	29.01	26.09	NeRF
TensoIR	26.04	21.89	27.17	43.01	18.31	31.71	27.69	NeRF
NeRO	25.24	**32.80**	<u>28.35</u>	42.13	**25.19**	31.24	<u>30.83</u>	NeRF
GS Shader	28.45	29.31	28.33	<u>43.42</u>	23.02	31.45	30.66	3DGS
GS-IR	25.57	19.41	25.06	38.57	18.83	30.66	26.35	3DGS
Relit 3DGS	26.55	20.18	26.92	**43.59**	19.91	<u>31.92</u>	28.18	3DGS
Ours	**29.10**	<u>32.57</u>	**29.06**	42.98	<u>24.66</u>	**32.62**	**31.83**	3DGS

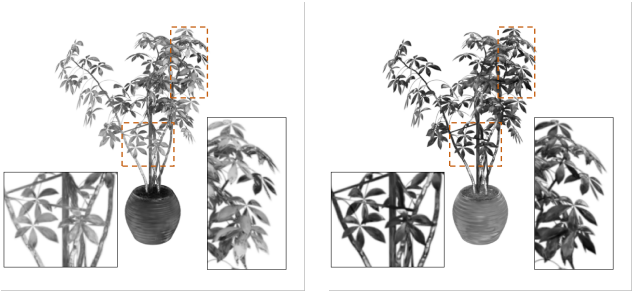


Figure 8. Occlusion map comparison. The left panel shows the occlusion map obtained with the guidance of L_{occ}, whereas the right panel shows the occlusion map generated without this guidance.

Shiny-Blender benchmark. Concentrating on the family of 3DGS pipelines, our method achieves the highest **average** score (31.83 dB), outperforming *GS Shader* by +1.17 dB, *Relit 3DGS* by +3.65 dB, and *GS-IR* by +5.48 dB. Scene-wise, it ranks first among 3DGS methods on five out of six objects (*Car*, *Ball*, *Helmet*, *Toaster*, and *Coffee*), evidencing robust gains on both glossy paints and highly specular metals (See Figure 7). These results indicate that our prototype-guided BRDF learning and occlusion-aware shader provide a clear advantage over prior 3DGS renderers while retaining real-time inference speed.

We further test on seven unbounded real-world scenes from Mip-NeRF360 and three unbounded real-world scenes from Ref-NeRF. As shown in Table 2 and Figure 6, unbounded real-scene novel-view synthesis remains challenging for current methods and noticeable gaps to the ground truth persist, especially on thin structures and high-frequency textures. Within this setting, our method achieves competitive view-synthesis quality and preserves finer geometric details. For a more complete qualitative comparison on real scenes, we additionally provide side-by-side visualizations against representative baselines in Appendix.

5.5. Ablation Study

We ablate on the Mip-NeRF360 benchmark starting from a 3DGS-based inverse-rendering baseline, then add the Prototype-Based Material Representation (PBMR) and the Learnable Occlusion (LO) modules individually to isolate their effects. This controlled setup quantifies how PBMR addresses material clustering and how the LO improves visibility handling. Below we detail the results for each component.

Analysis of Learnable Occlusion. To assess the impact of our *learnable occlusion* module, we compare against three ablations: (i) removing the occlusion field entirely, (ii) keeping the occlusion SH coefficients but omitting the pseudo-label–guided loss L_{occ}, and (iii) replacing the default backward-only supervision with dual-direction supervision (w_b, w_f). As reported in Table 6, introducing the occlusion field with L_{occ} improves performance over the no-occlusion baseline, while disabling L_{occ} yields a clear drop, confirming the importance of the pseudo-label guidance. Dual-direction supervision provides only marginal gains and does not consistently improve over the backward-only setting, consistent with our discussion that forward-direction targets can be less reliable in unbounded scenes. Qualitative results in Figure 8 further support this: occlusion maps learned with L_{occ} are more plausible (e.g., leaf surfaces are correctly identified as unoccluded), whereas removing L_{occ} produces inaccurate occlusion.

SH order ablation. We further analyze the effect of the spherical-harmonic (SH) order used in the occlusion field. Table 5 reports results with increasing SH order. Using a lower order provides insufficient angular expressiveness and slightly degrades performance. Increasing the SH order consistently improves quality, but the gains quickly saturate: higher orders yield only marginal improvements compared to the increased parameter count and memory cost. This suggests that most occlusion information is low-frequency, and a moderate SH order already captures the dominant visibility structure. Based on this trade-off, we adopt 4th-order SH as the default configuration.

Table 5. Ablation on the spherical-harmonic (SH) order for the learnable occlusion field. Higher is better for PSNR and SSIM, lower is better for LPIPS. The number of SH coefficients equals $(l+1)^2$.

SH order l	#coeffs	PSNR↑	SSIM↑	LPIPS↓
3	16	24.31	0.721	0.239
4	25	24.54	0.728	0.231
5	36	24.58	0.729	0.230
6	49	24.59	0.729	0.229

Figure 9. Albedo map comparison. The left image employs a PBRM, whereas the right image does not. The left result exhibits a smoother reflectance map that more closely matches the true reflectance distribution.

Table 6. Combined ablation on **Prototype Based Material Representation (PBMR)** and **Learnable Occlusion (LO)** on Mip-NeRF360 Outdoor dataset. We additionally ablate the occlusion supervision direction: backward-only (default) vs. dual-direction (w_b, w_f).

Modules							Metrics		
PBMR	$\mathcal{L}_{\text{agg}}$	$\mathcal{L}_{\text{res}}$	Occ. Field	L_{occ}	Sup.	Notes	PSNR↑	SSIM↑	LPIPS↓
					–	None	23.50	0.686	0.280
			✓		–	no L_{occ}	23.96	0.706	0.265
			✓	✓	w_b	LO only (backward)	24.18	0.720	0.244
			✓	✓	w_b, w_f	LO only (dual-dir)	24.01	0.712	0.256
✓					–	PBMR (no-loss)	23.51	0.688	0.288
✓	✓	✓			–	PBMR only (no LO)	23.92	0.704	0.268
✓	✓	✓	✓	✓	w_b	PBMR + LO (backward)	**24.54**	**0.728**	**0.231**

Prototype Based Material Representation. Table 6 compares three settings: the full PBMR, a variant that drops both losses (no-loss), and a no-prototype baseline. Without the losses, the prototype branch behaves nearly the same as no-prototype: in the absence of priors, introducing prototypes degenerates into a linear mixing layer and does not reshape the albedo distribution. By contrast, adding $\mathcal{L}_{\text{agg}}$ and $\mathcal{L}_{\text{res}}$ yields consistent gains: averaged over all scenes, PBMR improves PSNR by 0.42 dB over no-prototype and by 0.41 dB over no-loss, with SSIM increases of +0.018 and +0.014, respectively. Qualitatively (Figure 9), PBMR produces a smoother, piecewise constant albedo, near-uniform reflectance within a material and sharp transitions.

6. Limitations

Our framework is designed for *static* multi-view captures and can robustly model a wide range of static scenes under unknown environment illumination, benefiting from the efficiency and differentiability of rasterized 3D Gaussian Splatting. Nevertheless, the method has several important conceptual and practical boundaries.

Material-prior expressiveness. PBMR assumes that a scene's reflectance can be *approximately* explained by a small set of prototypes, enforced only as a soft prior. For real-world scenes with highly diverse, spatially continuous, or fine-scale material variation, the residual term may carry a larger portion of the appearance, which can reduce the interpretability of prototype assignments and weaken material–lighting disentanglement. Designing stronger yet still efficient material priors (e.g., richer prototype parameterizations or learned spatial regularizers) is an important direction for future work.

Static-scene assumption. Our learnable occlusion field is optimized from per-view splatting transmittance and is tied to a fixed scene configuration. While this works well for static scenes, it does *not* directly extend to dynamic settings where geometry and topology change over time. In such cases, the occlusion field would need to continuously track time-varying visibility; however, when objects move or deform, the previously learned visibility coefficients may no longer accurately represent environment occlusion, leading to inconsistent shading and degraded inverse-rendering decomposition. Extending occlusion-aware 3DGS inverse rendering to dynamic scenes would likely require explicit temporal modeling (e.g., deformation fields or per-time visibility representations) and remains future work.

Lighting-model approximations. Like many real-time inverse-rendering methods, our pipeline relies on a practical PBR approximation (deferred split–sum shading with simplified visibility integration) rather than full light-transport simulation. Under complex real-world illumination—for example, highly structured indoor lighting, strong inter-reflections, sharp cast shadows, or caustics—these approximations can introduce inaccuracies in the recovered illumination and, consequently, in the disentanglement of BRDF and lighting. This is a shared limitation of efficient PBR-based inverse renderers, and improving physical fidelity (e.g., more accurate visibility and multi-bounce transport) while preserving real-time optimization remains an important direction for future work.

7. Conclusion

We have presented a real-time, end-to-end inverse-rendering framework built on 3D Gaussian Splatting. By coupling a per-Gaussian spherical-harmonic visibility

field—trained via an occlusion-guided self-supervision signal—with a prototype-driven reflectance prior, our method jointly optimises geometry, materials, and illumination without any ray tracing or offline baking. Across synthetic and real-world benchmarks, the method consistently outperforms prior 3DGS inverse-rendering approaches and delivers NeRF-level decomposition quality at a fraction of the computational cost.

Acknowledgement

This work was supported by the National Natural Science Foundation of China (Project Number: 62272019).

References

[1] J. T. Barron, B. Mildenhall, M. Tancik, P. Hedman, R. Martin-Brualla, and P. P. Srinivasan. Mip-nerf: A multiscale representation for anti-aliasing neural radiance fields. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 5855–5864, 2021. 2

[2] J. T. Barron, B. Mildenhall, D. Verbin, P. P. Srinivasan, and P. Hedman. Mip-nerf 360: Unbounded anti-aliased neural radiance fields. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5470–5479, 2022. 7, 8

[3] S. Bi, Z. Xu, P. Srinivasan, B. Mildenhall, K. Sunkavalli, M. Hašan, Y. Hold-Geoffroy, D. Kriegman, and R. Ramamoorthi. Neural reflectance fields for appearance acquisition. *arXiv preprint arXiv:2008.03824*, 2020. 2

[4] M. Boss, R. Braun, V. Jampani, J. T. Barron, C. Liu, and H. Lensch. Nerd: Neural reflectance decomposition from image collections. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 12684–12694, 2021. 2

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

[6] H. Chen, Z. Lin, and J. Zhang. Gi-gs: Global illumination decomposition on gaussian splatting for inverse rendering. *arXiv preprint arXiv:2410.02619*, 2024. 1

[7] P. Debevec. Rendering synthetic objects into real scenes: Bridging traditional and image-based graphics with global illumination and high dynamic range photography. In *Acm siggraph 2008 classes*, pages 1–10. 2008. 2

[8] S. Fridovich-Keil, A. Yu, M. Tancik, Q. Chen, B. Recht, and A. Kanazawa. Plenoxels: Radiance fields without neural networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5501–5510, 2022. 2

[9] J. Gao, C. Gu, Y. Lin, Z. Li, H. Zhu, X. Cao, L. Zhang, and Y. Yao. Relightable 3d gaussians: Realistic point cloud relighting with brdf decomposition and ray tracing. In *European Conference on Computer Vision*, pages 73–89. Springer, 2024. 3, 7, 9

[10] P. Hedman, P. P. Srinivasan, B. Mildenhall, J. T. Barron, and P. Debevec. Baking neural radiance fields for real-time view synthesis. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 5875–5884, 2021. 2

[11] B. Huang, Z. Yu, A. Chen, A. Geiger, and S. Gao. 2d gaussian splatting for geometrically accurate radiance fields. In *ACM SIGGRAPH 2024 conference papers*, pages 1–11, 2024. 8, 9

[12] L. Huang, J. Bai, J. Guo, and Y. Guo. Gs++: Error analyzing and optimal gaussian splatting. *arXiv preprint arXiv:2402.00752*, 2(3), 2024. 2

[13] Y. Jiang, J. Tu, Y. Liu, X. Gao, X. Long, W. Wang, and Y. Ma. Gaussianshader: 3d gaussian splatting with shading functions for reflective surfaces. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5322–5332, 2024. 1, 3, 7, 8, 9

[14] 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. 3, 7, 9

[15] B. Karis and E. Games. Real shading in unreal engine 4. *Proc. Physically Based Shading Theory Practice*, 4(3):1, 2013. 3

[16] 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. 1, 2, 3, 8, 9

[17] J. Li, Z. Wu, E. Zamfir, and R. Timofte. Recap: Better gaussian relighting with cross-environment captures. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages –, 2025. 3

[18] Z. Li, T. Müller, A. Evans, R. H. Taylor, M. Unberath, M.-Y. Liu, and C.-H. Lin. Neuralangelo: High-fidelity neural surface reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8456–8465, 2023. 2

[19] Z. Liang, H. Li, K. Jia, K. Guo, and Q. Zhang. Gus-ir: Gaussian splatting with unified shading for inverse rendering. *arXiv preprint arXiv:2411.07478*, 2024. 1, 8, 9

[20] Z. Liang, Q. Zhang, Y. Feng, Y. Shan, and K. Jia. Gs-ir: 3d gaussian splatting for inverse rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 21644–21653, 2024. 1, 3, 7, 8, 9

[21] C.-H. Lin, W.-C. Ma, A. Torralba, and S. Lucey. Barf: Bundle-adjusting neural radiance fields. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 5741–5751, 2021. 2

[22] W. Lin, Y. Feng, and Y. Zhu. Rtgs: Enabling real-time gaussian splatting on mobile devices using efficiency-guided pruning and foveated rendering. *arXiv e-prints*, pages arXiv–2407, 2024. 2

[23] S. Lombardi, T. Simon, J. Saragih, G. Schwartz, A. Lehrmann, and Y. Sheikh. Neural volumes: Learning dynamic renderable volumes from images. *arXiv preprint arXiv:1906.07751*, 2019. 6

[24] R. Martin-Brualla, N. Radwan, M. S. Sajjadi, J. T. Barron, A. Dosovitskiy, and D. Duckworth. Nerf in the wild: Neural radiance fields for unconstrained photo collections. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 7210–7219, 2021. 2

[25] A. Meka, M. Maximov, M. Zollhoefer, A. Chatterjee, H.-P. Seidel, C. Richardt, and C. Theobalt. Lime: Live intrinsic

material estimation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6315–6324, 2018. 2

[26] 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. *Communications of the ACM*, 65(1):99–106, 2021. 1, 2

[27] T. Müller, A. Evans, C. Schied, and A. Keller. Instant neural graphics primitives with a multiresolution hash encoding. *ACM transactions on graphics (TOG)*, 41(4):1–15, 2022. 2

[28] 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. 7

[29] K. Park, U. Sinha, J. T. Barron, S. Bouaziz, D. B. Goldman, S. M. Seitz, and R. Martin-Brualla. Nerfies: Deformable neural radiance fields. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 5865–5874, 2021. 2

[30] K. Park, U. Sinha, P. Hedman, J. T. Barron, S. Bouaziz, D. B. Goldman, R. Martin-Brualla, and S. M. Seitz. Hypernerf: A higher-dimensional representation for topologically varying neural radiance fields. *arXiv preprint arXiv:2106.13228*, 2021. 2

[31] B. Poole, A. Jain, J. T. Barron, and B. Mildenhall. Dreamfusion: Text-to-3d using 2d diffusion. *arXiv preprint arXiv:2209.14988*, 2022. 2

[32] A. Pumarola, E. Corona, G. Pons-Moll, and F. Moreno-Noguer. D-nerf: Neural radiance fields for dynamic scenes. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10318–10327, 2021. 2

[33] Z. Qian, S. Wang, M. Mihajlovic, A. Geiger, and S. Tang. 3dgs-avatar: Animatable avatars via deformable 3d gaussian splatting. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5020–5030, 2024. 2

[34] Y. Shi, Y. Wu, C. Wu, X. Liu, C. Zhao, H. Feng, J. Zhang, B. Zhou, E. Ding, and J. Wang. Gir: 3d gaussian inverse rendering for relightable scene factorization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2025. 1

[35] 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. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 7495–7504, 2021. 2

[36] C. Sun, M. Sun, and H.-T. Chen. Direct voxel grid optimization: Super-fast convergence for radiance fields reconstruction. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5459–5469, 2022. 2

[37] M. Tancik, V. Casser, X. Yan, S. Pradhan, B. Mildenhall, P. P. Srinivasan, J. T. Barron, and H. Kretzschmar. Block-nerf: Scalable large scene neural view synthesis. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8248–8258, 2022. 2

[38] H. Turki, D. Ramanan, and M. Satyanarayanan. Mega-nerf: Scalable construction of large-scale nerfs for virtual fly-throughs. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 12922–12931, 2022. 2

[39] 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. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5481–5490. IEEE, 2022. 7

[40] F. Wang, M.-J. Rakotosaona, M. Niemeyer, R. Szeliski, M. Pollefeys, and F. Tombari. Unisdf: Unifying neural representations for high-fidelity 3d reconstruction of complex scenes with reflections. *Advances in Neural Information Processing Systems*, 37:3157–3184, 2024. 8

[41] J. Wang, Z. Xu, X. Liu, and W. Wang. Gi-gs: Global illumination with gaussian splatting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024. 3

[42] G. Wu, T. Yi, J. Fang, L. Xie, X. Zhang, W. Wei, W. Liu, Q. Tian, and X. Wang. 4d gaussian splatting for real-time dynamic scene rendering. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 20310–20320, 2024. 2

[43] T. Wu, J.-M. Sun, Y.-K. Lai, Y. Ma, L. Kobbelt, and L. Gao. Deferredgs: Decoupled and relightable gaussian splatting with deferred shading. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2025. 3

[44] Y. Yan, H. Lin, C. Zhou, W. Wang, H. Sun, K. Zhan, X. Lang, X. Zhou, and S. Peng. Street gaussians: Modeling dynamic urban scenes with gaussian splatting. In *European Conference on Computer Vision*, pages 156–173. Springer, 2024. 2

[45] H.-M. Yang, X.-Y. Zhang, F. Yin, Q. Yang, and C.-L. Liu. Convolutional prototype network for open set recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(5):2358–2370, 2020. 6

[46] Y. Yu and W. A. Smith. Inverserendernet: Learning single image inverse rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3155–3164, 2019. 2

[47] 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 *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5453–5462, 2021. 2

[48] 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(6):1–18, 2021. 2, 7, 9

[49] 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. 2, 7