EFFICIENT FORWARD PATH TRACING ON PLANAR-BASED GAUSSIAN SPLATTING FOR INVERSE RENDERING

Anonymous authors
Paper under double-blind review

ABSTRACT

Gaussian splatting has emerged as a promising representation for novel view synthesis, inverse rendering, and relighting, owing to its efficient rasterization and high-fidelity scene representation. Existing Gaussian-based inverse rendering methods typically adopt forward or deferred shading with precomputed or approximate visibility and indirect illumination, limiting their ability to model complex global light transport. Gaussian ray tracing methods can explicitly model visibility and indirect illumination, but incur high computational cost due slow ray-Gaussian intersection. Therefore, accurately disentangling material properties from illumination under complex global illumination effects while maintaining computational efficiency remains a key challenge in Gaussian-based inverse rendering. To address this challenge, we propose an efficient inverse rendering framework that combines planar-based Gaussians with the reconstructed scene mesh for efficient and physically-based material and illumination optimization. In the geometry reconstruction stage, we adopt the modified VA-GS to reconstruct accurate scene geometry, which is then kept fixed during inverse rendering. Based on the reconstructed geometry, we introduce a per-Gaussian forward shading method that leverages the scene mesh to query visibility and model multi-bounce light transport, enabling the outgoing radiance of each Gaussian primitive to be computed before rasterization. The resulting per-Gaussian radiance is α-blended into the rendered image through differentiable Gaussian splatting to jointly optimize the material and illumination parameters. Experiments demonstrate that our method outperforms existing inverse rendering baselines while maintaining high-fidelity novel view synthesis.

1 INTRODUCTION

Recent advances in novel view synthesis (NVS) have enabled high-quality rendering from multi-view observations. In particular, 3D Gaussian splatting (3DGS) (Kerbl et al., 2023) combines explicit anisotropic Gaussian primitives with differentiable rasterization to achieve real-time rendering. Subsequent methods improve rendering quality and view consistency through anti-aliasing, view-adaptive representations, and hierarchical sorting (Yu et al., 2024a; Lu et al., 2024; Radl et al., 2024). More recent work accelerates optimization through dense correspondence-based initialization that eliminates iterative densification (Kotovenko et al., 2026), and extends Gaussian splatting to non-linear camera models while supporting integration with secondary-ray tracing (Wu et al., 2025a). Despite these advances, the learned radiance primarily describes scene appearance under the captured illumination rather than intrinsic surface reflectance. Reconstructing appearance for NVS therefore does not directly provide the material–illumination decomposition required for relighting and material editing.

Gaussian-based inverse rendering methods (Liang et al., 2024; Wu et al., 2025b; Gao et al., 2024; Sun et al., 2025) extend Gaussian splatting with explicit reflectance and illumination models, enabling their estimation through differentiable physically-based rendering. Recent approaches incorporate visibility evaluation and indirect light modeling (Ye et al., 2025; Chen et al., 2025b; Gu et al., 2025), while radiometric consistency constraints provide additional supervision for radiance along unobserved viewpoints (Han et al., 2026). Nevertheless, errors in surface geometry and incident

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

lighting can be absorbed into material estimates, resulting in an inaccurate decomposition despite agreement with the observed images (Ye et al., 2025; Gu et al., 2025; Han et al., 2026). Several Gaussian-based inverse rendering methods adopt deferred shading, in which geometric and material attributes are rasterized before pixel-wise shading (Liang et al., 2024; Wu et al., 2025b; Chen et al., 2025b; Gu et al., 2025; Bo et al., 2026; Han et al., 2026). When combined with Monte Carlo light transport evaluation, this formulation incurs a shading cost that scales with the number of shaded pixels and the sampling budget. Moreover, blended geometric and material attributes vary with the camera position and the field of view, and may not correspond to a valid surface point, particularly at multi-layer regions and depth boundaries. Therefore, artifacts such as dark boundaries and aliasing occur. These considerations motivate us to evaluate light transport at each individual Gaussian primitive using an explicit surface representation, prior to α-blending the Gaussians.

We present an efficient inverse rendering framework that combines planar Gaussian splatting with a surface mesh. We first reconstruct the scene geometry using the modified VA-GS (Li et al., 2025) and extract a corresponding mesh, keeping both the Gaussian geometry and the mesh fixed during material and illumination optimization. The Gaussian primitives provide material parameterization while the mesh supports visibility queries and secondary ray intersections for multi-bounce light transport. We perform forward path tracing for each Gaussian primitive to estimate its outgoing radiance, which is then α-blended into the image through differentiable Gaussian rasterization. We jointly optimize per-Gaussian material properties and the environment map. This framework separates geometry reconstruction from material and illumination estimation while integrating physically-based light transport with Gaussian rasterization.

In conclusion, our main contributions are as follows:

- A per-Gaussian forward shading method that integrates mesh-based light transport with differentiable Gaussian rasterization for material and lighting optimization.
- A two-stage inverse rendering framework that combines planar Gaussians with the scene meshes, separating geometry reconstruction from material and lighting estimation.
- An efficient training and relighting pipeline that combines hardware-accelerated ray tracing with Gaussian rasterization, substantially reducing optimization time and enabling fast rendering under novel lighting conditions.

2 Related Works

Gaussian Splatting. 3D Gaussian splatting (3DGS) (Kerbl et al., 2023) represents scenes using anisotropic Gaussian primitives and differentiable rasterization for efficient NVS. 3D Gaussian ray tracing (3DGRT) (Moenne-Loccoz et al., 2024) extends this representation to hardware-accelerated ray tracing, supporting distorted cameras and secondary-ray effects. Surface-oriented methods improve geometric reconstruction through primitive design and geometric constraints. SuGaR (Guédon & Lepetit, 2024) introduces surface-alignment regularization for mesh extraction, while 2DGS (Huang et al., 2024) and PGSR (Chen et al., 2025a) adopt planar Gaussian primitives with depth and normal constraints. Gaussian opacity fields (GOF) (Yu et al., 2024b) instead extracts surfaces from a continuous Gaussian-derived opacity field. VA-GS Li et al. (2025), GSPrior (Noda et al., 2026), and AmbiSuR (Li et al., 2026) further strengthen geometric optimization through cross-view alignment, self-derived TSDF priors, and ambiguity-aware regularization, respectively.

Inverse Rendering with Gaussian Splatting. Gaussian-based inverse rendering commonly adopts deferred or forward shading, distinguished by whether shading is evaluated after or before α-blending the Gaussians. Deferred shading methods first rasterize geometric and material attributes into screen-space buffers and then evaluate shading per pixel. GS-IR (Liang et al., 2024) combines depth-derived normal regularization with occlusion baking, whereas DeferredGS (Wu et al., 2025b) distills surface normals from an auxiliary signed distance field. GI-GS (Chen et al., 2025b), IRGS (Gu et al., 2025) further estimate indirect illumination through G-buffer-based path tracing or 2D Gaussian ray tracing, respectively. GHPT (Bo et al., 2026) combines deferred shading on Gaussian-rendered G-buffers with mesh-based path tracing for visibility and indirect illumination. Its three-stage framework estimates mesh materials and illumination before optimizing per-Gaussian materials through factorized inverse path tracing. RadioGS (Han et al., 2026) enforces consistency

between learned surfel radiance and rendered radiance, including along unobserved viewpoints. In contrast, forward shading methods evaluate radiance at Gaussian primitives before α-blending the Gaussians. R3DG (Gao et al., 2024) combines per-Gaussian reflectance estimation with ray-traced visibility, while SVG-IR (Sun et al., 2025) introduces Gaussian vertex shading and color interpolation to represent spatially-varying appearance.

3 PRELIMINARIES

Gaussian Splatting. 3D Gaussian splatting (3DGS) (Kerbl et al., 2023) represents a scene as a set of anisotropic Gaussian primitives. Each Gaussian is defined by its center $\boldsymbol{\mu}_i \in \mathbb{R}^3$ and covariance $\boldsymbol{\Sigma}_i \in \mathbb{R}^{3\times3}$:

$$G_i(\boldsymbol{x}) = \exp\left(-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu}_i)^\top \boldsymbol{\Sigma}_i^{-1}(\boldsymbol{x}-\boldsymbol{\mu}_i)\right). \tag{1}$$

To ensure a valid positive semi-definite covariance, $\boldsymbol{\Sigma}_i$ is parameterized as

$$\boldsymbol{\Sigma}_i = \boldsymbol{R}_i \boldsymbol{S}_i \boldsymbol{S}_i^\top \boldsymbol{R}_i^\top, \qquad \boldsymbol{S}_i = \mathrm{diag}(s_{i,x}, s_{i,y}, s_{i,z}), \tag{2}$$

where $\boldsymbol{R}_i$ is the rotation matrix derived from a unit quaternion and $\boldsymbol{S}_i$ contains the principal scales.

Given the camera intrinsic matrix $\boldsymbol{K}$ and the world-to-camera transformation $(\boldsymbol{R}_c, \boldsymbol{t}_c)$, each Gaussian is projected onto the image plane as

$$\begin{aligned} z_i \begin{bmatrix} \boldsymbol{\mu}_i' \\ 1 \end{bmatrix} &= \boldsymbol{K}\left(\boldsymbol{R}_c\boldsymbol{\mu}_i + \boldsymbol{t}_c\right), \\ \boldsymbol{\Sigma}_i' &= \boldsymbol{J}_i \boldsymbol{R}_c \boldsymbol{\Sigma}_i \boldsymbol{R}_c^\top \boldsymbol{J}_i^\top, \end{aligned} \tag{3}$$

where $\boldsymbol{J}_i$ denotes the Jacobian of the perspective projection evaluated at the Gaussian center. The projected Gaussians are depth-sorted and blended through front-to-back α-blending:

$$\boldsymbol{C}(\boldsymbol{u}) = \sum_{i=1}^{N(\boldsymbol{u})} T_i(\boldsymbol{u})\alpha_i(\boldsymbol{u})\boldsymbol{c}_i, \qquad T_i(\boldsymbol{u}) = \prod_{j<i}\left(1-\alpha_j(\boldsymbol{u})\right), \tag{4}$$

where $\alpha_i(\boldsymbol{u})$ combines the projected Gaussian response with its learned opacity, $T_i(\boldsymbol{u})$ denotes the accumulated transmittance, and $\boldsymbol{c}_i$ is the view-dependent color represented using spherical harmonics (SH).

Planar-Based Gaussian Splatting. We adopt the planar Gaussian representation proposed in PGSR (Chen et al., 2025a) to prevent self-intersection of the rays cast from the Gaussian centers with the scene mesh, in which each anisotropic Gaussian is associated with a local plane. Specifically, the shortest axis of the Gaussian primitive defines the plane normal $\boldsymbol{n}$, while the Gaussian center $\boldsymbol{\mu}$ specifies a point on the corresponding local plane.

Under the same world-to-camera transformation $(\boldsymbol{R}_c, \boldsymbol{t}_c)$, the Gaussian center and normal in camera coordinates are given by

$$\boldsymbol{\mu}^c = \boldsymbol{R}_c\boldsymbol{\mu} + \boldsymbol{t}_c, \qquad \boldsymbol{n}^c = \boldsymbol{R}_c\boldsymbol{n}. \tag{5}$$

The local plane in camera space can therefore be expressed as

$$(\boldsymbol{n}^c)^\top \boldsymbol{x} = d, \qquad d = (\boldsymbol{n}^c)^\top \boldsymbol{\mu}^c. \tag{6}$$

For a pixel p, the corresponding camera ray is defined by $\boldsymbol{K}^{-1}\tilde{\boldsymbol{p}}$, where $\tilde{\boldsymbol{p}}$ denotes the homogeneous pixel coordinate. Intersecting this ray with the local plane yields the planar depth

$$z(p) = \frac{d}{(\boldsymbol{n}^c)^\top \boldsymbol{K}^{-1}\tilde{\boldsymbol{p}}}. \tag{7}$$

This planar formulation associates Gaussian primitives with explicit local surface geometry, providing depth and normal information for geometry reconstruction and subsequent physically-based shading.

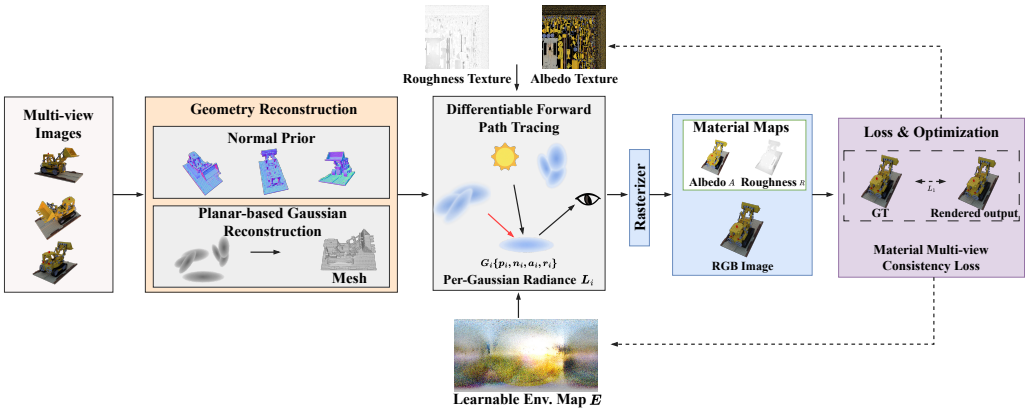

Figure 1: Our Gaussian-based inverse rendering pipeline with per-Gaussian forward shading. We first reconstruct the scene geometry using the modified VA-GS with multi-view images and normal priors and extract the surface mesh for visibility evaluation and secondary ray intersections during light transport computation. The geometric parameters of the Gaussian primitives and mesh are fixed during material and illumination optimization. We perform multi-bounce path tracing at Gaussian centers to estimate outgoing radiance of each Gaussian before differentiable Gaussian rasterization. The Gaussian material parameters and environment lighting are jointly optimized.

4 METHOD

Overview. Figure 1 illustrates our two-stage inverse rendering framework. We first reconstruct the scene geometry using the modified VA-GS (Li et al., 2025) and extract a surface mesh, keeping both the Gaussian geometry and the mesh fixed in the second stage. We then perform path tracing for each Gaussian, using the mesh for visibility queries and secondary ray tracing to model multi-bounce light transport. The resulting outgoing radiance is α-blended through differentiable Gaussian splatting, enabling joint optimization of Gaussian material properties and the environment map.

4.1 FORWARD PATH TRACING ON PLANAR-BASED GAUSSIAN SPLATTING

We compute outgoing radiance for each Gaussian before rasterization. The extracted mesh is used for visibility queries and secondary ray intersections. During material and illumination estimation, the Gaussian geometry is fixed.

Each Gaussian primitive is characterized by its center $\boldsymbol{x}_k$, normal $\boldsymbol{n}_k$, and material parameters $\boldsymbol{\theta}_k$. The material parameters are defined as:

$$\boldsymbol{\theta}_k = (\boldsymbol{a}_k, r_k), \tag{8}$$

where $\boldsymbol{a}_k$ and r_k denote the albedo and roughness, respectively. The normal $\boldsymbol{n}$ is estimated from the shortest axis of the Gaussian covariance. Given a viewing direction $\boldsymbol{\omega}_\mathrm{o}$, the outgoing radiance of the k-th Gaussian primitive is computed at its center $\boldsymbol{x}$ as:

$$L_{\mathrm{o},k}(\boldsymbol{\omega}_\mathrm{o}) = \int_{\Omega_\mathrm{i}^+} f_\mathrm{r}(\boldsymbol{\omega}_\mathrm{i}, \boldsymbol{\omega}_\mathrm{o}; \boldsymbol{\theta}_k) L_\mathrm{i}(\boldsymbol{x}_k, \boldsymbol{\omega}_\mathrm{i})(\boldsymbol{n}_k \cdot \boldsymbol{\omega}_\mathrm{i})\, \mathrm{d}\boldsymbol{\omega}_\mathrm{i}, \tag{9}$$

Here, BRDF $f_\mathrm{r}(\boldsymbol{\omega}_\mathrm{i}, \boldsymbol{\omega}_\mathrm{o}; \boldsymbol{\theta}_k)$ combines a Lambertian diffuse term with a GGX microfacet specular term. We decompose the incident radiance into direct and indirect illumination

$$L_\mathrm{i}(\boldsymbol{\omega}) = V(\boldsymbol{\omega}) L_\mathrm{env}(\boldsymbol{\omega}) + L_\mathrm{indirect}(\boldsymbol{\omega}), \tag{10}$$

where $V(\boldsymbol{\omega})$ denotes visibility toward the environment map along direction $\boldsymbol{\omega}$. Visibility is evaluated by tracing rays against the mesh. Direct illumination is estimated using multiple importance sampling (MIS), combining environment map and BRDF samples with the balance heuristic.

Although planar-based Gaussian splatting is employed, the reconstructed mesh still slightly deviates from the Gaussian centers, self-intersections can occur during ray tracing. Following GHPT (Bo

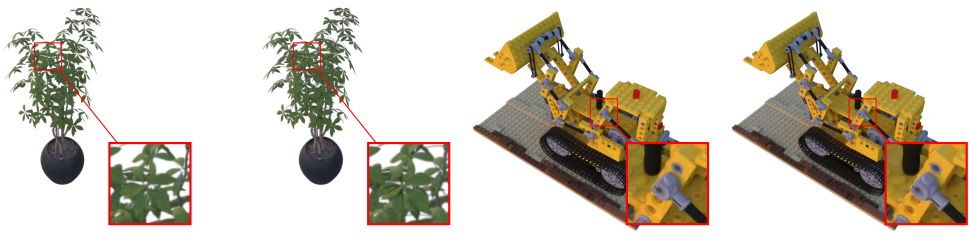

Figure 2: Qualitative comparison between forward and deferred shading on the Ficus and Lego scenes. From left to right: Ficus with forward shading, Ficus with deferred shading, Lego with forward shading, and Lego with deferred shading. We can observe artifacts such as dark boundaries (Ficus) and aliasing (Lego) in the results with deferred shading.

et al., 2026), we ignore intersections with very small hit distances and reject back-facing intersections.

Indirect illumination is estimated by continuing BRDF sampled paths from mesh intersections. At each intersection, the surface position, shading normal, and UV coordinates are obtained via barycentric interpolation, after which the material parameters are retrieved from the corresponding textures.

Substituting Eq. 10 into Eq. 9 separates the outgoing radiance into direct and indirect components. We estimate these contributions using path tracing to obtain the radiance of each Gaussian:

$$\widehat{L}_{\mathrm{o},k} = \widehat{L}^{\mathrm{direct}}_{\mathrm{o},k} + \widehat{L}^{\mathrm{indirect}}_{\mathrm{o},k}. \tag{11}$$

The shaded Gaussians are then rasterized and blended using α-blending to obtain the linear pixel color:

$$C(p) = \sum_{k=1}^{N} T_k(p)\alpha_k(p)L_{\mathrm{o},k}, \qquad T_k(p) = \prod_{j=1}^{k-1}\left(1-\alpha_j(p)\right). \tag{12}$$

Here, N is the number of Gaussians contributing to pixel p, $\alpha_k(p)$ is the opacity-weighted Gaussian value, and $T_k(p)$ is the accumulated transmittance.

Forward Shading vs. Deferred Shading. Gaussian-based inverse rendering adopts either forward or deferred shading. Deferred shading first rasterizes geometric and material attributes into screen-space buffers and then evaluates shading per pixel. This allows shading to vary at pixel resolution and can better resolve sharp specular highlights, provided that the rasterized attributes are sufficiently accurate. However, blending Gaussian attributes before shading may produce inaccurate depth, normal, and material estimation, particularly for multi-layer regions, depth boundaries, and thin objects such as leaves. The blended attributes also vary as the camera position and the field of view. Additionally, the shading cost also scales with the number of shaded pixels.

In contrast, the forward shading formulation evaluates the BRDF and incident lighting at each Gaussian center before α-blending the resulting radiance. This makes geometric and material attributes constant during shading, independent of the camera position and the field of view. For comparison, we also implement a deferred shading variant using the same training configuration as our forward shading approach. Figure 2 presents the qualitative comparison between forward and deferred shading, we can observe artifacts such as dark boundaries and aliasing due to the inaccurate depth in the deferred shading results.

4.2 Modified VA-GS

We reconstruct the scene geometry using the modified VA-GS (Li et al., 2025), which is a variant of PGSR. Since accurate geometry is essential for subsequent material decomposition and lighting estimation, we maintain normal-based geometry alignment to the end of training that allows the Gaussian primitives to better converge to the underlying surface structure. Meanwhile, we discard the normal smoothing loss for better details.

We further enhance the geometry optimization with additional constraints. To alleviate the influence of specular reflections on geometry reconstruction, we leverage a pretrained monocular normal estimation model, MoGe-3 (Kong et al., 2026), to provide auxiliary normal supervision. The normal consistency loss is defined as

$$\mathcal{L}_{\text{normal}} = \sum \left(1 - \boldsymbol{n}_{\text{render}} \cdot \boldsymbol{n}_{\text{mono}}\right). \tag{13}$$

where $\boldsymbol{n}_{\text{mono}}$ denotes the normal estimated by MoGe-3, while $\boldsymbol{n}_{\text{render}}$ denotes the corresponding rendered normal obtained through α-blending. Following prior work (Gao et al., 2024; Gu et al., 2025; Bo et al., 2026), we also employ a binary cross-entropy loss to constrain the reconstructed geometry using the foreground object mask:

$$\mathcal{L}_{\text{mask}} = -\mathcal{M} \log \mathcal{O} - (1 - \mathcal{M}) \log (1 - \mathcal{O}). \tag{14}$$

where $\mathcal{M}$ and $\mathcal{O}$ represent the binary object mask and accumulated opacity, respectively.

4.3 Inverse Rendering with Forward Path Tracing on VA-GS

We optimize the Gaussian material parameters and the environment map such that the rendered images closely match the observed images.

$$\mathcal{L}_{\text{rgb}} = \frac{1}{|\mathcal{P}|} \sum_{p \in \mathcal{P}} \left\| C(p) - I_{\text{gt}}(p) \right\|_1, \tag{15}$$

Although material attributes can be optimized using the image reconstruction objective, observations captured from different viewpoints may provide conflicting appearance cues. Consequently, the material properties estimated for the same physical surface can vary across views. To alleviate this inconsistency, following previous work (Zhang et al., 2025), we impose a multi-view consistency constraint on the estimated albedo and roughness maps. We additionally employ material smoothness losses for rasterized albedo and roughness, denoted as L_{as} and L_{rs}, to encourage spatial coherence of material features. Therefore, the overall optimization objective for inverse rendering is formulated as a weighted sum of the following loss terms:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{rgb}} + \lambda_{\text{as}} \mathcal{L}_{\text{as}} + \lambda_{\text{rs}} \mathcal{L}_{\text{rs}} + \lambda_{\text{a}} \mathcal{L}^{\text{a}}_{\text{mv}} + \lambda_{\text{r}} \mathcal{L}^{\text{r}}_{\text{mv}}. \tag{16}$$

For indirect illumination, material parameters at secondary intersections are sampled from mesh UV textures. These textures are separately optimized using an ℓ_1 loss against the rendered Gaussian material maps. The updated textures are used in subsequent path tracing. Please refer to Appendix A for further details.

5 Experiments

5.1 Datasets and Metrics

We evaluate our method on two synthetic datasets: Synthetic4Relight (Zhang et al., 2022) and TensoIR (Jin et al., 2023), including NVS, albedo and roughness estimation, and relighting capabilities. We employ PSNR, SSIM, and LPIPS for evaluating NVS, albedo, roughness, and relighting.

5.2 Implementation Details

Gaussian and Mesh Reconstruction. In the first stage, we optimize the modified VA-GS for 30K iterations, and set the weights of the normal prior and binary cross entropy to 0.2 and 0.05, respectively. We reconstruct the meshes with a voxel size of 0.002 for the scenes, and the reconstructed meshes are decimated to 500K triangles using quadric error metrics. We also use Blender to generate UV atlases for the meshes, providing the UV coordinates required for the differentiable hybrid forward path tracing.

Differentiable Forward Path Tracing. In the second stage, we implement the differentiable hybrid forward path tracing using SlangPy (Kallweit et al., 2025) with PyTorch interoperation, which supports hardware-accelerated ray tracing and automatic differentiation through Slang.D (Bangaru

Table 1: **Quantitative comparisons on the Synthetic4Relight dataset** (Zhang et al., 2022).

Method	Novel View Synthesis			Albedo			Relighting		
	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓
TensoIR	35.80	0.978	0.049	29.69	0.951	0.079	30.58	0.946	0.065
GS-IR	33.95	0.965	0.057	19.48	0.896	0.117	25.40	0.924	0.083
GI-GS	35.42	0.973	0.042	24.68	0.931	0.085	27.36	0.945	0.070
R3DG	36.06	0.980	0.031	28.67	0.953	0.063	32.89	0.968	0.051
IRGS	35.66	0.970	0.050	33.07	0.953	0.060	34.76	0.963	0.055
GHPT	37.11	0.976	0.037	32.20	0.950	0.074	35.87	0.971	0.045
RadioGS	33.81	0.976	0.045	29.57	0.943	0.071	34.29	0.975	0.041
Ours	39.84	0.982	0.031	33.01	0.957	0.070	37.15	0.976	0.039

Table 2: **Quantitative comparisons on the TensoIR dataset** (Jin et al., 2023).

Method	Novel View Synthesis			Albedo			Relight		
	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓
TensoIR	35.09	0.976	0.041	29.28	0.950	0.085	28.83	0.950	0.077
GS-IR	35.02	0.964	0.043	32.25	0.943	0.085	24.69	0.886	0.099
GI-GS	36.75	0.973	0.037	31.97	0.941	0.086	24.70	0.886	0.106
R3DG	33.15	0.960	0.040	28.55	0.922	0.087	27.60	0.921	0.081
IRGS	35.58	0.965	0.048	33.40	0.954	0.076	30.63	0.935	0.075
SVG-IR	36.71	0.975	0.033	30.34	0.951	0.074	31.10	0.946	0.056
GHPT	35.57	0.967	0.038	33.67	0.938	0.088	32.46	0.948	0.057
RadioGS	37.79	0.981	0.025	33.03	0.946	0.078	32.16	0.954	0.048
Ours	38.09	0.977	0.030	33.55	0.940	0.086	32.88	0.955	0.048

et al., 2023). We perform optimization for 5K iterations, using 256 samples and permitting up to 5 bounces for both training and inference. We represent the environment map at a resolution of 512×256 and initialize it with a uniform value of 0.5. The mesh material textures are stored at a resolution of 2048×2048, while the albedo and roughness of each Gaussian are initialized to 0.1 and 0.9, respectively. The learning rates for the Gaussian material attributes and the environment map are set to 0.05 and 0.005, respectively, while the mesh material textures are optimized using the Adam optimizer with a learning rate of 0.003. We further employ a multi-view material consistency loss on the estimated albedo and roughness, with weights set to 0.05 and 0.05, respectively. The weight of the material smoothness loss is set to 0.03.

Experiments are conducted on an NVIDIA RTX PRO 4000 Blackwell GPU with 24 GB VRAM. Total optimization takes approximately 40 minutes per scene (30 minutes for the first stage and 10 minutes for the second stage).

5.3 Comparison

Baselines. We compare our method with NeRF-based and Gaussian-based inverse rendering baselines, including TensoIR (Jin et al., 2023), GS-IR (Liang et al., 2024), GI-GS (Chen et al., 2025b), R3DG(Gao et al., 2024), SVG-IR (Sun et al., 2025), IRGS (Gu et al., 2025), GHPT (Bo et al., 2026), and RadioGS (Han et al., 2026). Following prior work (Liang et al., 2024; Chen et al., 2025b; Sun et al., 2025; Gu et al., 2025; Bo et al., 2026; Han et al., 2026), we resolve the inherent ambiguity between albedo and lighting by globally aligning each RGB channel of the predicted albedo with the ground truth. The results of R3DG (Gao et al., 2024), IRGS(Gu et al., 2025), SVG-IR (Sun et al., 2025) and RadioGS(Han et al., 2026) are obtained using their official implementations.

Synthetic4Relight. Table 1 shows that our method achieves the highest PSNR and SSIM for NVS and the same lowest LPIPS as R3DG. For albedo estimation, our method achieves the highest SSIM and the second-highest PSNR, slightly below IRGS, although its LPIPS is higher than that of several baselines. For relighting, our method achieves the best results across all three metrics. Figure 3 provides qualitative comparisons of NVS, albedo and roughness estimation, and relighting under different environment maps. Additional qualitative results are provided in Appendix B.

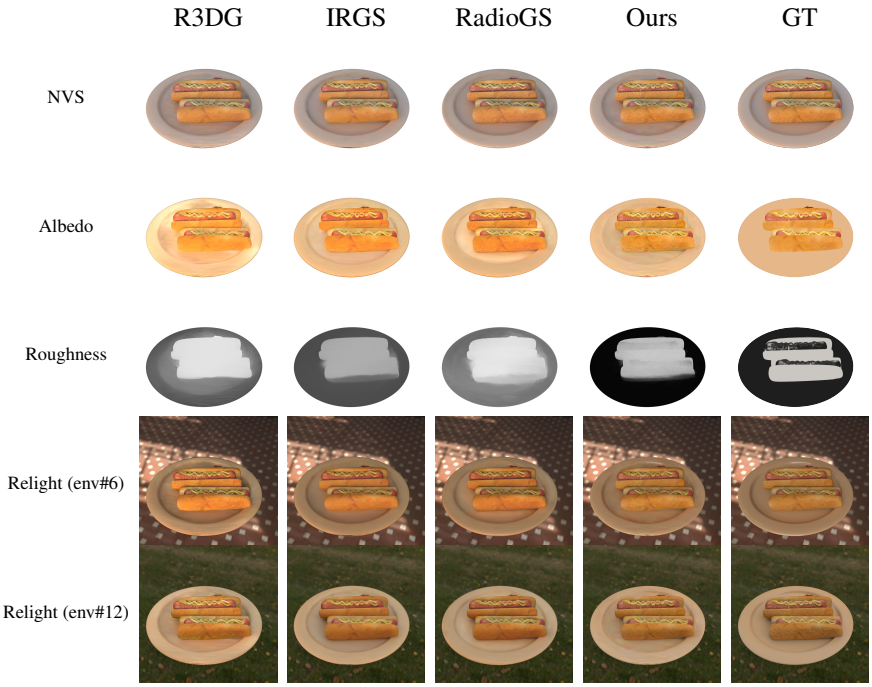

Figure 3: Qualitative comparison of NVS, material estimation, and relighting results on Hotdog from the Synthetic4Relight dataset.

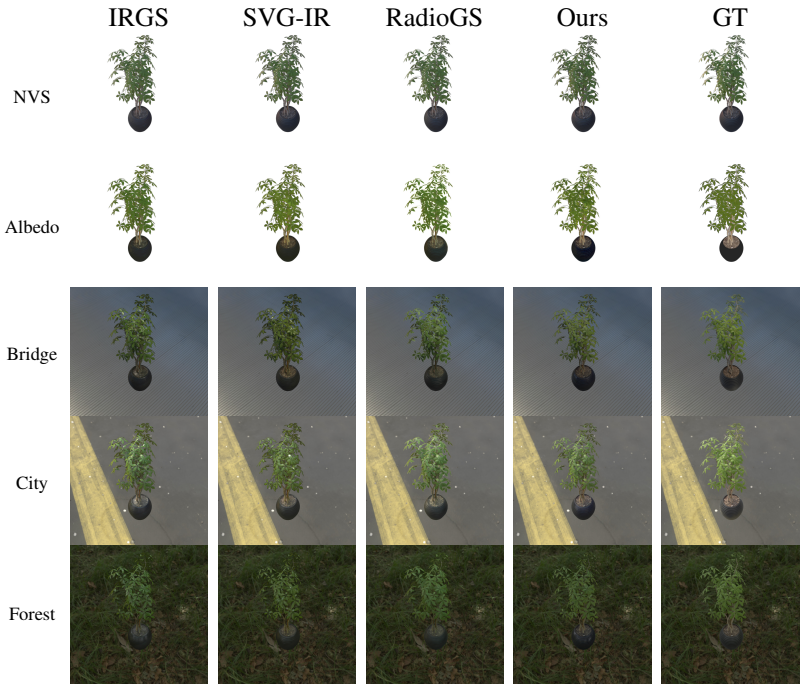

Figure 4: Qualitative comparison of NVS, material estimation, and relighting results on Ficus from the TensoIR dataset.

TensoIR. Table 2 shows that our method achieves the highest PSNR for NVS, while RadioGS obtains better SSIM and LPIPS. For albedo estimation, our method achieves the second-highest PSNR. For relighting, our method achieves the highest PSNR and SSIM and ties with RadioGS for the lowest LPIPS. These results demonstrate that our method combines efficient training with high-quality

Table 3: Ablation studies on the Synthetic4Relight (Zhang et al., 2022) and TensoIR (Jin et al., 2023) datasets on NVS, albedo estimation, and relighting performance.

	Synthetic4Relight					TensoIR				
	NVS	Albedo	Relight			NVS	Albedo	Relight		
	PSNR↑	PSNR↑	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	PSNR↑	PSNR↑	SSIM↑	LPIPS↓
16 spp (relight)	38.69	–	32.49	0.970	0.049	37.31	–	32.46	0.947	0.056
64 spp (relight)	39.65	–	37.05	0.975	0.041	37.97	–	32.81	0.953	0.050
w/o indirect	39.51	32.73	35.59	0.973	0.041	37.91	**33.97**	32.86	0.954	0.050
w/o indirect (relight)	37.14	–	34.68	0.968	0.047	36.44	–	32.28	0.947	0.058
w/o L_{mv}	39.65	32.75	36.94	**0.976**	0.040	37.90	33.68	**32.96**	**0.955**	**0.048**
Full	**39.83**	**33.01**	**37.15**	**0.976**	**0.039**	**38.09**	33.55	32.88	**0.955**	**0.048**

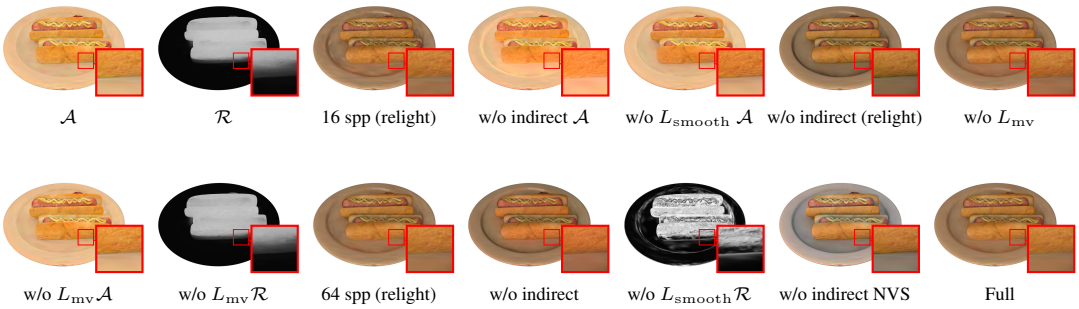

Figure 5: Ablation study on Hotdog from the Synthetic4Relight dataset. $\mathcal{A}$ and $\mathcal{R}$ denote the estimated albedo and roughness, respectively. We analyze the effects of sample count, multi-view material consistency, indirect illumination, and material smoothness loss.

NVS and relighting. Figure 4 provides qualitative comparisons of NVS, albedo, and relighting under different environment maps. Additional qualitative results are provided in Appendix B.

5.4 Ablation Study

We conduct ablation studies on the Synthetic4Relight and TensoIR datasets to investigate the effects of different components of our method on NVS, material estimation, and relighting performance. We report PSNR for NVS and albedo estimation, while PSNR, SSIM, and LPIPS are reported for relighting in Table 3. Qualitative comparisons are shown in Figure 5.

We also visualize the recovered albedo and roughness maps to analyze the effects of material consistency and indirect illumination. Removing the multi-view material consistency loss generally leads to less accurate material decomposition, as reflected in the degraded albedo estimation and roughness recovery. When indirect illumination is omitted during training, interreflection effects tend to be incorrectly absorbed into the estimated albedo, particularly in concave regions. Omitting indirect illumination during relighting also reduces the fidelity of the rendered appearance.

6 Conclusion and Future Work

In this paper, we present an efficient inverse rendering framework that combines planar Gaussians with a surface mesh for material and illumination decomposition and relightable scene representation. The mesh reconstructed by the modified VA-GS enables effective modeling of visibility and multi-bounce light transport, while per-Gaussian forward shading supports efficient joint optimization without sacrificing the rendering efficiency of Gaussian splatting. Experiments demonstrate improved inverse rendering and relighting performance over existing Gaussian-based inverse rendering methods, while preserving high-fidelity NVS. Our framework is also compatible with other planar Gaussian representations, such as 2DGS (Huang et al., 2024) and its variants. Future work will focus on improving geometry and material estimation and modeling spatially-varying materials and normals to further reduce shading artifacts.

REFERENCES

Sai Praveen Bangaru, Lifan Wu, Tzu-Mao Li, Jacob Munkberg, Gilbert Bernstein, Jonathan Ragan-Kelley, Frédo Durand, Aaron Lefohn, and Yong He. SLANG.D: Fast, modular and differentiable shader programming. *ACM Transactions on Graphics*, 42(6):264:1–264:28, December 2023. ISSN 0730-0301.

Jonathan T. Barron, Ben Mildenhall, Dor Verbin, Pratul P. Srinivasan, and Peter Hedman. Mip-NeRF 360: Unbounded anti-aliased neural radiance fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5470–5479, Los Alamitos, CA, USA, 2022. IEEE Computer Society.

Jinyang Bo, Fan Dou, Wenrui Quan, Shangxun Liu, Yang Xu, Yuhe Zhang, Kang Li, and Guohua Geng. GHPT: Real-time relightable gaussian splatting using hybrid path tracing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 25990–25999, June 2026.

Danpeng Chen, Hai Li, Weicai Ye, Yifan Wang, Weijian Xie, Shangjin Zhai, Nan Wang, Haomin Liu, Hujun Bao, and Guofeng Zhang. PGSR: Planar-based Gaussian splatting for efficient and high-fidelity surface reconstruction. *IEEE Transactions on Visualization and Computer Graphics*, 31(9):6100–6111, September 2025a.

Hongze Chen, Zehong Lin, and Jun Zhang. GI-GS: Global illumination decomposition on Gaussian splatting for inverse rendering. In *International Conference on Learning Representations*, volume 2025, pp. 75713–75742, 2025b.

Jian Gao, Chun Gu, Youtian Lin, Zhihao Li, Hao Zhu, Xun Cao, Li Zhang, and Yao Yao. Relightable 3D Gaussians: Realistic point cloud relighting with brdf decomposition and ray tracing. In Aleš Leonardis, Elisa Ricci, Stefan Roth, Olga Russakovsky, Torsten Sattler, and Gül Varol (eds.), *European Conference on Computer Vision*, pp. 73–89, Cham, Switzerland, 2024. Springer Nature Switzerland. ISBN 978-3-031-72995-9.

Chun Gu, Xiaofei Wei, Zixuan Zeng, Yuxuan Yao, and Li Zhang. IRGS: Inter-reflective Gaussian splatting with 2D Gaussian ray tracing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10943–10952, Los Alamitos, CA, USA, 2025. IEEE Computer Society.

Antoine Guédon and Vincent Lepetit. SuGaR: Surface-aligned gaussian splatting for efficient 3D mesh reconstruction and high-quality mesh rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5354–5363, June 2024.

Kyu Beom Han, Jaeyoon Kim, Woo Jae Kim, Jinhwan Seo, and Sung-Eui Yoon. Radiometrically consistent gaussian surfels for inverse rendering. In *The Fourteenth International Conference on Learning Representations*, 2026.

Binbin Huang, Zehao Yu, Anpei Chen, Andreas Geiger, and Shenghua Gao. 2D Gaussian splatting for geometrically accurate radiance fields. In *ACM Conference on Computer Graphics and Interactive Techniques*, pp. 32:1–32:11, New York, NY, USA, 2024. Association for Computing Machinery. ISBN 9798400705250.

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

Simon Kallweit, Chris Cummings, Benedikt Bitterli, Sai Bangaru, and Yong He. SlangPy. Computer software, 2025. Version 0.38.0.

Bernhard Kerbl, Georgios Kopanas, Thomas Leimkuehler, and George Drettakis. 3D Gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics*, 42(4):139:1–139:14, July 2023. ISSN 0730-0301.

Lingyu Kong, Ruicheng Li, Ruicheng Wang, Sicheng Xu, Chengtang Yao, Jianfeng Xiang, and Jiaolong Yang. MoGe-3: Fine-detail monocular geometry estimation with self-guided sparse volumetric refinement. arXiv preprint arXiv:2607.17967, 2026.

Dmytro Kotovenko, Olga Grebenkova, and Björn Ommer. EDGS: Eliminating densification for efficient convergence of 3DGS. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 41065–41076, 2026.

Jiahe Li, Jiawei Zhang, Xiao Bai, Jin Zheng, Xiaohan Yu, Lin Gu, and Gim Hee Lee. Revisiting photometric ambiguity for accurate gaussian-splatting surface reconstruction. In *International Conference on Machine Learning*. PMLR, 2026.

Qing Li, Huifang Feng, Xun Gong, and Yu-shen Liu. VA-GS: Enhancing the geometric representation of gaussian splatting via view alignment. In *Advances in Neural Information Processing Systems*, volume 38, 2025.

Zhihao Liang, Qi Zhang, Ying Feng, Ying Shan, and Kui Jia. GS-IR: 3D Gaussian splatting for inverse rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 21644–21653, Los Alamitos, CA, USA, 2024. IEEE Computer Society.

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

Nicolas Moenne-Loccoz, Ashkan Mirzaei, Or Perel, Riccardo de Lutio, Janick Martinez Esturo, Gavriel State, Sanja Fidler, Nicholas Sharp, and Zan Gojcic. 3D Gaussian ray tracing: Fast tracing of particle scenes. *ACM Transactions on Graphics*, 43(6):232:1–232:19, November 2024. ISSN 0730-0301.

Takeshi Noda, Yu-shen Liu, and Zhizhong Han. 3D gaussian splatting with self-constrained priors for high fidelity surface reconstruction. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 26041–26051, June 2026.

Lukas Radl, Michael Steiner, Mathias Parger, Alexander Weinrauch, Bernhard Kerbl, and Markus Steinberger. StopThePop: Sorted Gaussian Splatting for view-consistent real-time rendering. *ACM Transactions on Graphics*, 43(4):64:1–64:17, July 2024. ISSN 0730-0301.

Hanxiao Sun, Yupeng Gao, Jin Xie, Jian Yang, and Beibei Wang. SVG-IR: Spatially-varying Gaussian splatting for inverse rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 16143–16152, Los Alamitos, CA, USA, 2025. IEEE Computer Society.

Qi Wu, Janick Martinez Esturo, Ashkan Mirzaei, Nicolas Moënne-Loccoz, and Zan Gojcic. 3DGUT: Enabling distorted cameras and secondary rays in gaussian splatting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 26036–26046, 2025a.

Tong Wu, Jia-Mu Sun, Yu-Kun Lai, Yuewen Ma, Leif Kobbelt, and Lin Gao. DeferredGS: Decoupled and relightable gaussian splatting with deferred shading. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 47(8):6307–6319, 2025b.

Kai Ye, Chong Gao, Guanbin Li, Wenzheng Chen, and Baoquan Chen. GeoSplatting: Towards geometry guided gaussian splatting for physically-based inverse rendering. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 28991–29000, 2025.

Zehao Yu, Anpei Chen, Binbin Huang, Torsten Sattler, and Andreas Geiger. Mip-Splatting: Alias-free 3D Gaussian Splatting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 19447–19456, 2024a.

Zehao Yu, Torsten Sattler, and Andreas Geiger. Gaussian opacity fields: Efficient adaptive surface reconstruction in unbounded scenes. *ACM Transactions on Graphics*, 43(6):271:1–271:13, November 2024b. ISSN 0730-0301.

Wenyuan Zhang, Jimin Tang, Weiqi Zhang, Yi Fang, Yu-shen Liu, and Zhizhong Han. MaterialRefGS: Reflective gaussian splatting with multi-view consistent material inference. In *Advances in Neural Information Processing Systems*, 2025.

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

APPENDIX

THE USE OF LLMS

In this work, we used ChatGPT for language refinement, and all associated revisions were reviewed and approved by the authors.

A IMPLEMENTATION DETAILS

A.1 FORWARD PATH TRACING ON PLANAR-BASED GAUSSIAN SPLATTING

The pseudocode of our method is given in Algorithm 1. We adopt a forward shading strategy to compute the outgoing radiance associated with each Gaussian primitive prior to Gaussian rasterization. Specifically, we treat the center of each Gaussian primitive as a shading point and evaluate its lighting response based on the corresponding normal, material properties, and viewing direction. For each Gaussian primitive, incident directions are generated using a combination of environment map importance sampling and BRDF importance sampling. Differentiable path tracing is then performed along these sampled directions to estimate light transport. Radiance contributions from different surface interactions are accumulated over multiple path bounces. The initial sampling step accounts for direct lighting from the environment map, while subsequent bounces model the reflection of incident light according to the surface material, progressively estimating indirect illumination. During path tracing, the extracted surface mesh is used for ray intersection tests and visibility queries. When a sampled ray intersects the mesh, the path state is updated using the hit position, surface normal, and corresponding material properties, after which light sampling continues for subsequent bounces. This process yields the outgoing radiance of each Gaussian primitive, which is then passed to the Gaussian rasterizer for α-blending. The forward pass evaluates both direct and indirect illumination, whereas the backward pass differentiates only the direct illumination component.

A.2 MATERIAL ALIGNMENT

Material parameters are associated with each Gaussian primitive, whereas mesh textures provide material values at secondary ray intersections. The two material representations are aligned in image space with the ℓ_1 loss between the rendered mesh material maps and rendered Gaussian material maps. The mesh textures are updated once per iteration using the Adam optimizer.

A.3 OPTIMIZATION

Multi-View Material Consistency. Following MaterialRefGS (Zhang et al., 2025), we constrain the rendered albedo and roughness maps across neighboring views. Patch correspondences are established using local plane-induced homographies and filtered by reprojection consistency. We apply a confidence-weighted L_1 loss $\mathcal{L}_{\text{mv}}^{\text{a}}$ to albedo and a directional penalty $\mathcal{L}_{\text{mv}}^{\text{r}}$ toward the detached minimum of paired roughness values, excluding background and normal-edge regions for the latter. Reference samples, warping coordinates, and confidence weights are detached; gradients propagate through neighboring-view material samples.

Material Smoothness. To encourage spatial coherence and suppress local fluctuations in the estimated materials, we introduce a smoothness loss on the rasterized albedo and roughness maps. For

Algorithm 1: Forward Path Tracing on Gaussian Splatting

Input: Gaussians $\mathcal{G}$, mesh $\mathcal{M}$, training images $\{I_v\}$, and initial environment map E^0
Output: Per-Gaussian materials $\Theta^*_{\mathcal{G}} = \{A^*, R^*\}$ and environment map E^*

Initialize $\Theta_{\mathcal{G}} = \{A, R\}$, $E \leftarrow E^0$, and $\mathcal{B} \leftarrow \mathrm{BVH}(\mathcal{M})$;
Initialize auxiliary mesh materials $\Theta_{\mathcal{M}}$;
for $t \leftarrow 1$ **to** N_{iter} **do**
  $v \sim \mathcal{V}, \quad \mathcal{P}_E \leftarrow \mathrm{Alias}(E), \quad \{\boldsymbol{n}_i\} \leftarrow \mathrm{Normal}(\mathcal{G}, v)$; `// View and lighting setup`
  foreach $g_i \in \mathcal{G}$ ***in parallel*** **do**
    $\mathcal{S}_i^E, \mathcal{S}_i^B \leftarrow \emptyset, \quad L_i^{\text{ind}} \leftarrow 0$;
    for $s \leftarrow 1$ **to** N_{spp} **do**
      $(\boldsymbol{x}, \boldsymbol{n}, \Theta) \leftarrow (\boldsymbol{x}_i, \boldsymbol{n}_i, \Theta_i), \quad \omega_o \leftarrow \mathrm{normalize}(\boldsymbol{c}_v - \boldsymbol{x}_i), \quad \beta \leftarrow 1$;
      for $k \leftarrow 0$ **to** $N_{\text{bounce}} - 1$ **do**
        $\omega_e \sim p_E(\omega; E)$; `// Environment map importance sampling`
        $(\mathcal{S}_i^E, L_i^{\text{ind}}) \leftarrow \mathrm{SampleEnv}(\boldsymbol{x}, \omega_e, \beta, \mathcal{B})$;
        $\omega_b \sim p_B(\omega \mid \Theta, \boldsymbol{n}, \omega_o)$; `// BRDF importance sampling`
        $h \leftarrow \mathrm{Intersect}(\boldsymbol{x}, \omega_b; \mathcal{B})$; `// Mesh intersection`
        if $h =$ **then**
          $(\mathcal{S}_i^B, L_i^{\text{ind}}) \leftarrow \mathrm{EnvHit}(\omega_b, \beta, E)$;
          break;
        $\beta \leftarrow \beta \frac{f_r(\omega_o, \omega_b; \Theta) |\boldsymbol{n}^\top \omega_b|}{p_B(\omega_b)}$;
        $(\boldsymbol{x}, \boldsymbol{n}, \Theta, \omega_o) \leftarrow (\boldsymbol{x}_h, \boldsymbol{n}_h, \Theta_{\mathcal{M}}(\boldsymbol{x}_h), -\omega_b)$;
    $L_i \leftarrow \mathrm{Shade}\left(\mathcal{S}_i^E, \mathcal{S}_i^B, L_i^{\text{ind}}, \Theta_i, E\right)$; `// Per-Gaussian radiance`
  $(\widetilde{I}_v, A_v, R_v) \leftarrow \mathcal{R}_{\text{GS}}(\mathcal{G}, v, \{L_i\})$; `// Gaussian splatting`
  $\mathcal{L} \leftarrow \hat{\mathcal{L}}$
  if $N_{\text{bounce}} > 1$ **then**
    $\Theta_{\mathcal{M}} \leftarrow \mathrm{Align}(\Theta_{\mathcal{M}}, A_v, R_v)$
  $(\Theta_{\mathcal{G}}, E) \leftarrow \mathrm{Update}(\Theta_{\mathcal{G}}, E, \nabla\mathcal{L})$;

a material map X, the loss is defined as

$$\mathcal{S}(X, I) = \frac{1}{|\Omega|} \sum_{p \in \Omega} \bar{I}(p) \left\| \nabla \bar{X}(p) \right\|_1, \tag{17}$$

where Ω denotes the set of image pixels, $\bar{X}$ and $\bar{I}$ are the channel means of the material map and observed image, respectively, and ∇ is the spatial gradient computed using Sobel operators.

B Additional Qualitative Results

We provide additional qualitative comparisons on the Synthetic4Relight (Zhang et al., 2022) and TensoIR (Jin et al., 2023) datasets in Figures 6 – 11. For the Synthetic4Relight dataset, we provide comparisons of NVS, albedo reconstruction, roughness estimation, and relighting results with the Gaussian-based methods R3DG Gao et al. (2024), IRGS (Gu et al., 2025), and RadioGS (Han et al., 2026). For the TensoIR dataset, we also provide comparisons of NVS, albedo estimation and relighting results with SVG-IR (Sun et al., 2025), IRGS (Gu et al., 2025), and RadioGS (Han et al., 2026). Additionally, we qualitatively evaluate our method on scenes from the Mip-NeRF 360 dataset (Barron et al., 2022) by providing the NVS and material decomposition results, including albedo and roughness, and the relighting results as shown in Figure 12. For the Mip-NeRF 360 (Barron et al., 2022) dataset, we reconstruct the meshes with a voxel size of 0.01, and training takes approximately 1.5 hours per scene for the first stage and approximately 1 hour per scene for the second stage.

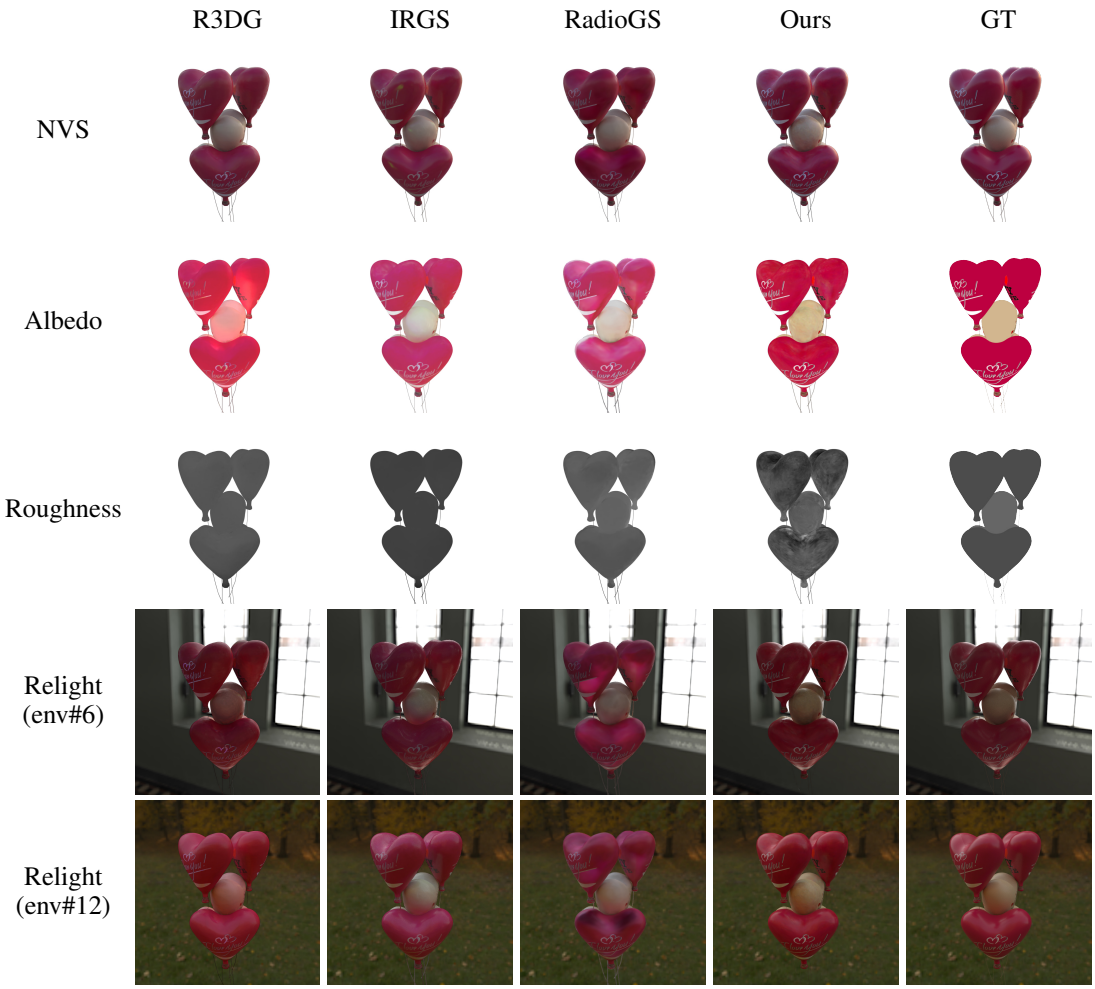

Figure 6: Qualitative comparison of NVS, material decomposition, and relighting results on Air Balloons from the Synthetic4Relight dataset.

B.1 Relighting

We show additional qualitative comparisons across scenes on the Synthetic4Relight and TensoIR datasets. R3DG(Gao et al., 2024) produces noticeable artifacts in the transitions between highlighted and shadowed regions. IRGS (Gu et al., 2025) tends to underestimate surface roughness, resulting in an overly smooth appearance, whereas R3DG (Gao et al., 2024) overestimates roughness, leading to weak specular reflections and an overall dim appearance. The relighting results produced by RadioGS(Han et al., 2026) exhibit noticeable rendering errors. In contrast, our method recovers surface materials more accurately, yielding relighting results that are closer to the ground truth.

B.2 Albedo And Roughness

Compared with other Gaussian-based methods, our method achieves more accurate material and lighting decomposition, producing cleaner albedo maps and roughness estimates that more closely match the ground truth.

B.3 Environment Map

We also conduct qualitative comparisons of the estimated environment maps across scenes from the Synthetic4Relight and TensoIR datasets in Figures 13 and 14. IRGS (Gu et al., 2025) generates

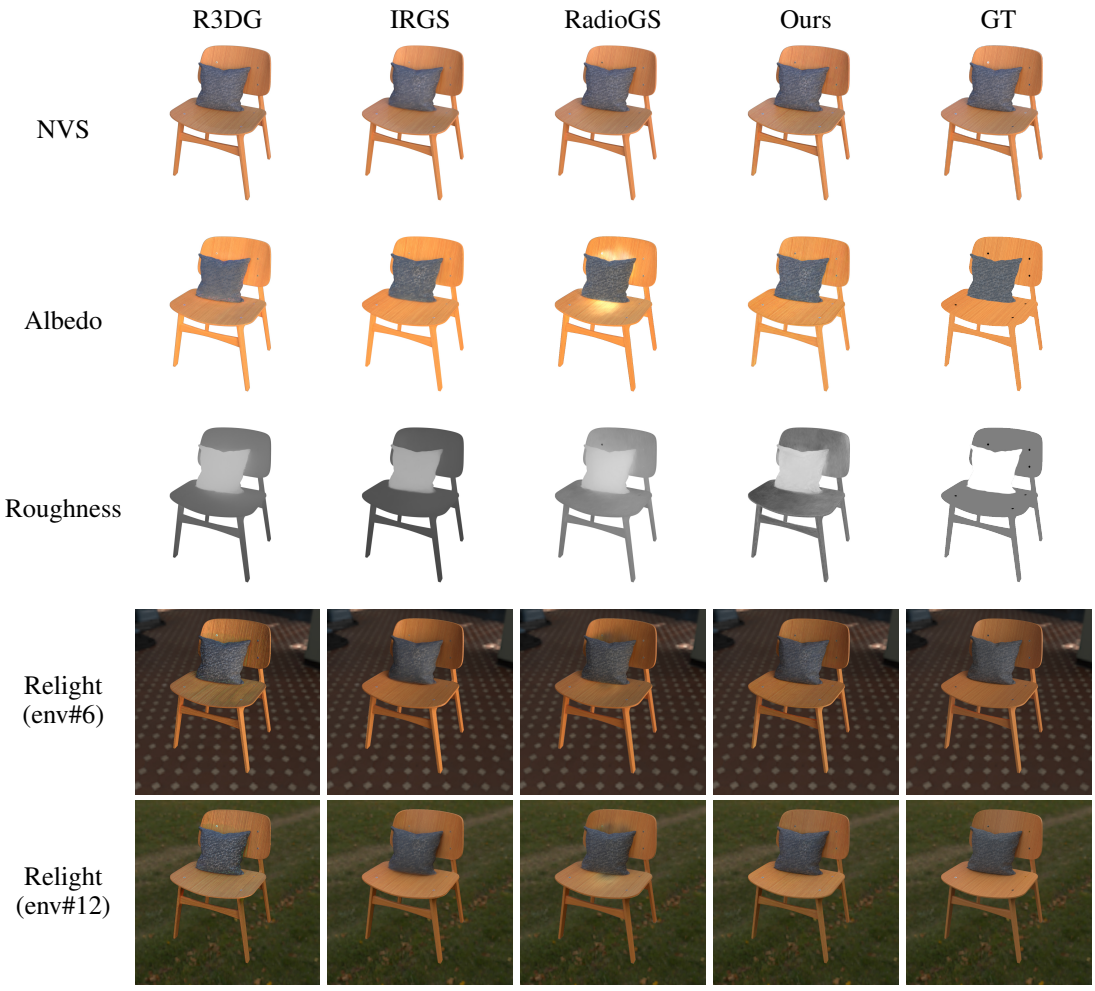

Figure 7: Qualitative comparison of NVS, material decomposition, and relighting results on Chair from the Synthetic4Relight dataset.

blurred and noisy environment maps. RadioGS (Han et al., 2026) produces environment maps with a noticeable grayish bias and a diffuse illumination distribution, failing to capture concentrated high-intensity light sources accurately. In contrast, our method recovers high-fidelity environment maps with finer details and fewer noise artifacts, closely matching the ground truth in both illumination distribution and sunlight direction.

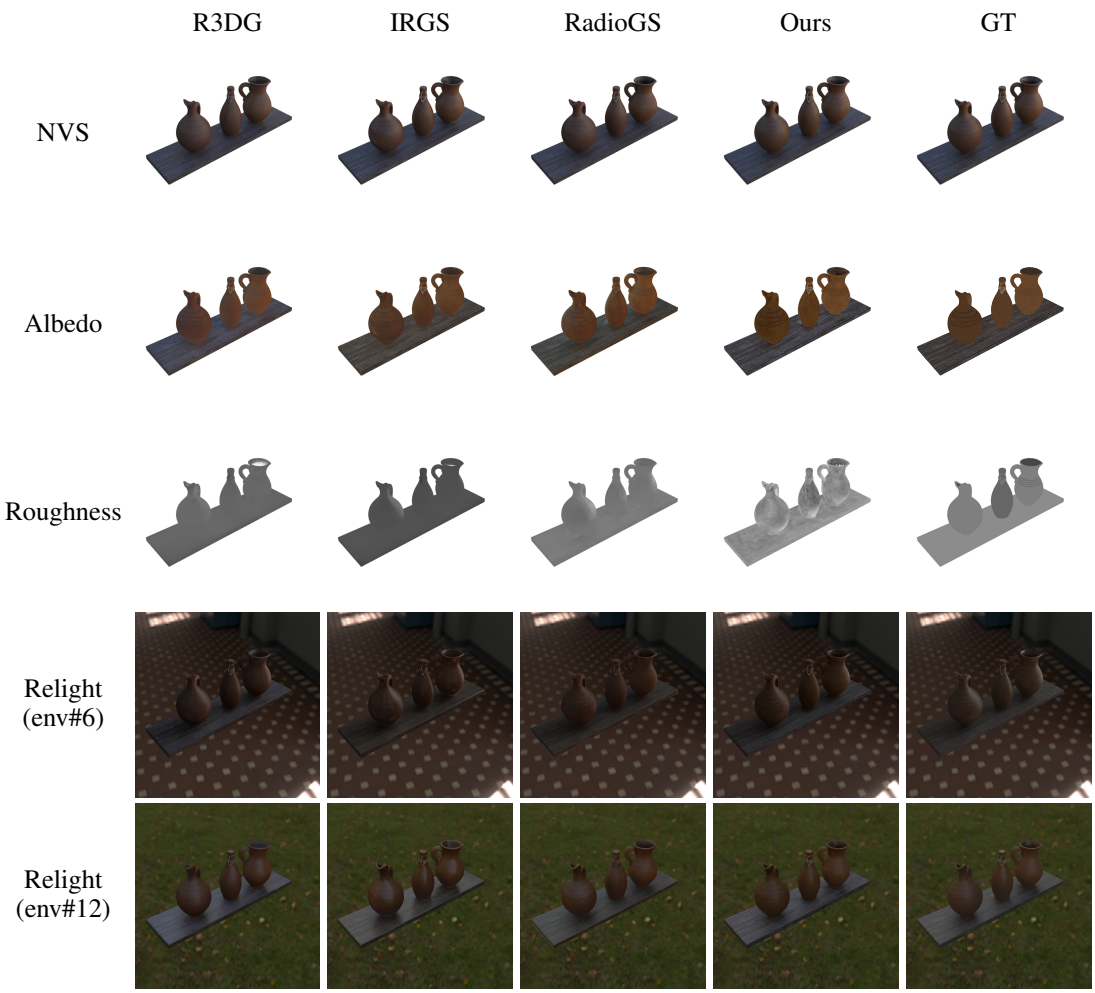

Figure 8: Qualitative comparison of NVS, material decomposition, and relighting results on Jugs from the Synthetic4Relight dataset.

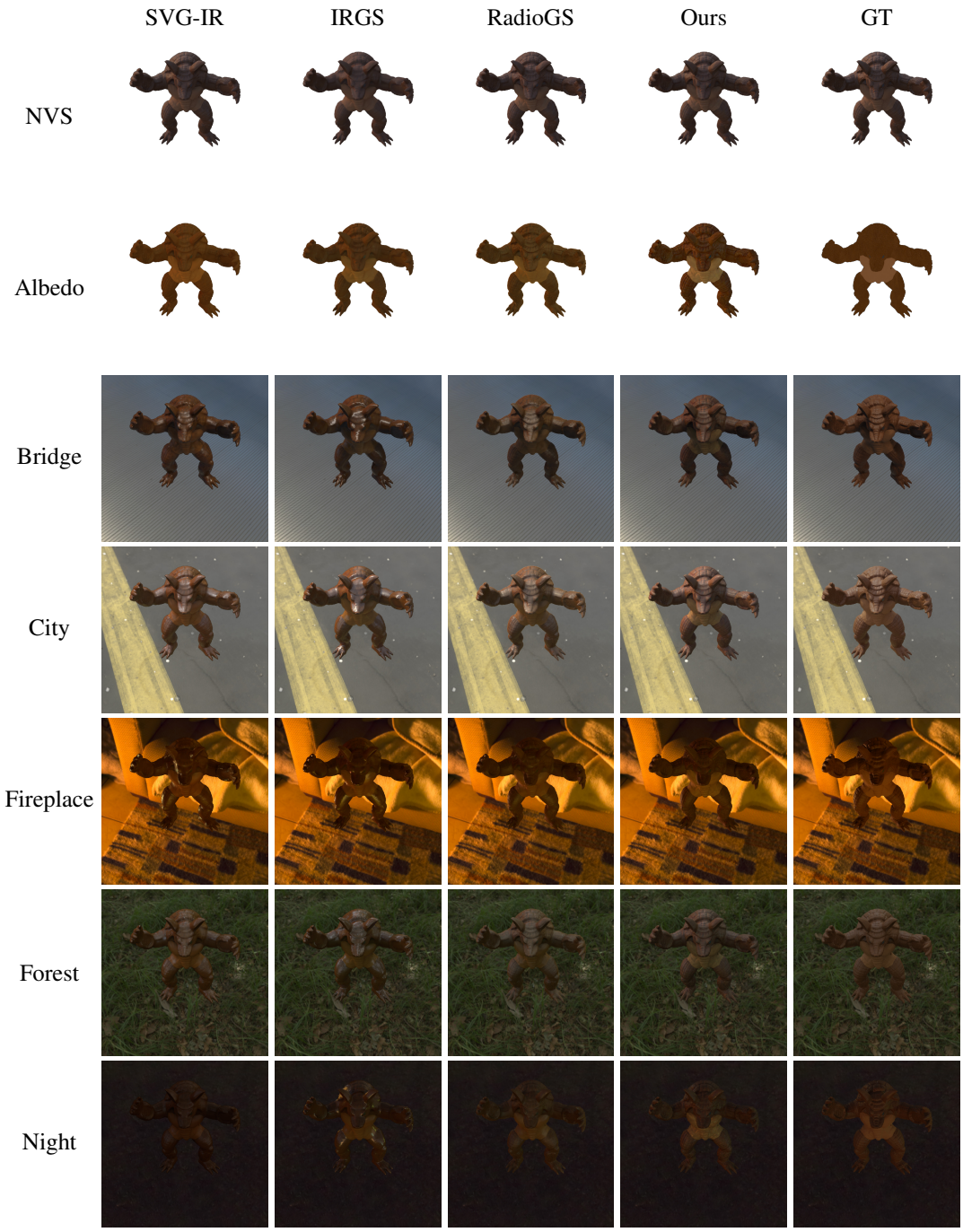

Figure 9: Qualitative comparison of NVS, albedo, and relighting results under different environment maps on Armadillo from the TensoIR dataset.

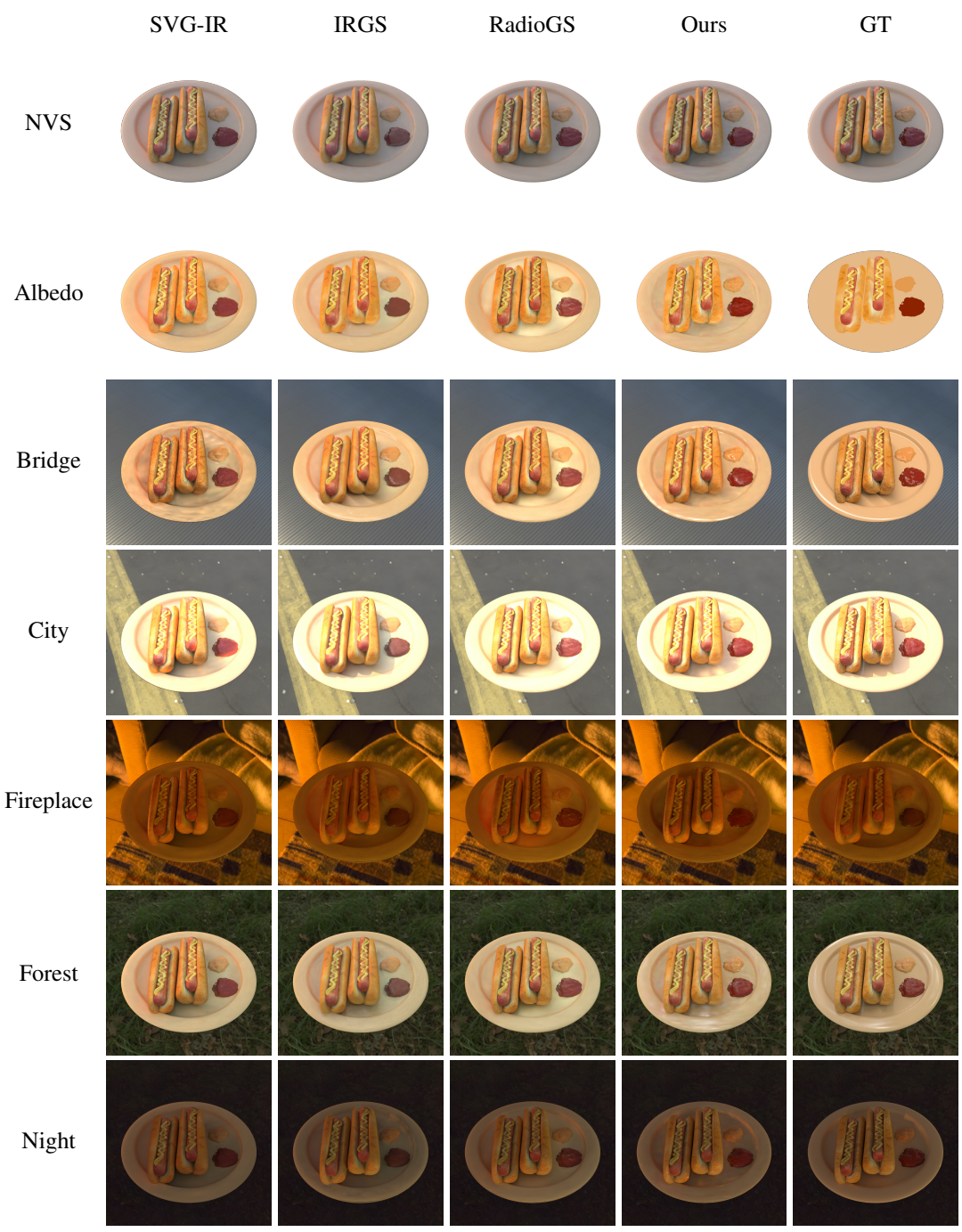

Figure 10: Qualitative comparison of NVS, albedo, and relighting results under different environment maps on Hotdog from the TensoIR dataset.



Figure 11: Qualitative comparison of NVS, albedo, and relighting results under different environment maps on Lego from the TensoIR dataset.

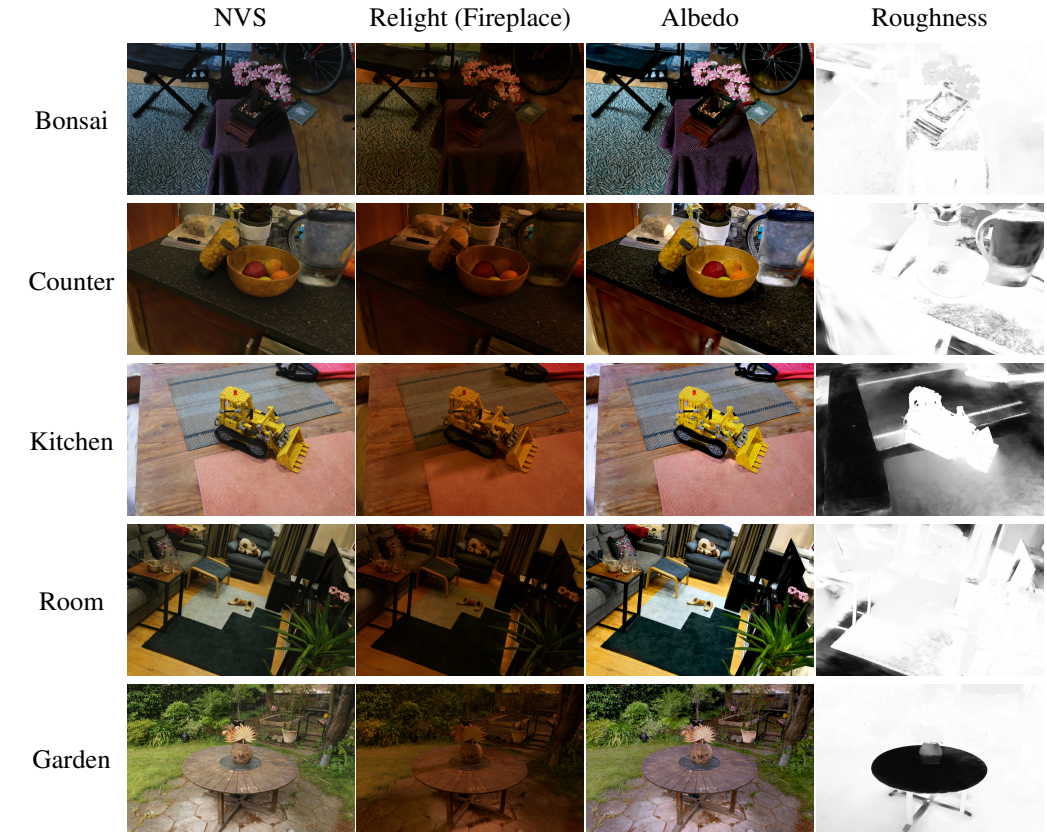

Figure 12: Qualitative results of NVS, relighting under the Fireplace environment map, and estimated albedo and roughness on five scenes from the Mip-NeRF 360 dataset.

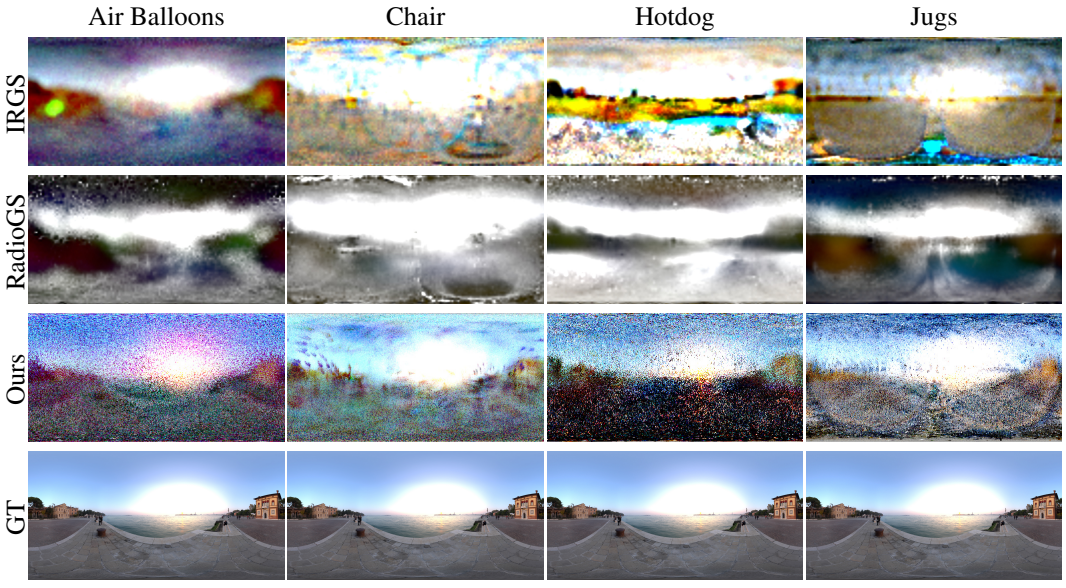

Figure 13: Qualitative comparison of the recovered environment maps on the Air Balloons, Chair, Hotdog, and Jugs scenes from the Synthetic4Relight dataset. The results of IRGS, RadioGS, our method, and the ground truth (GT) are shown from top to bottom.

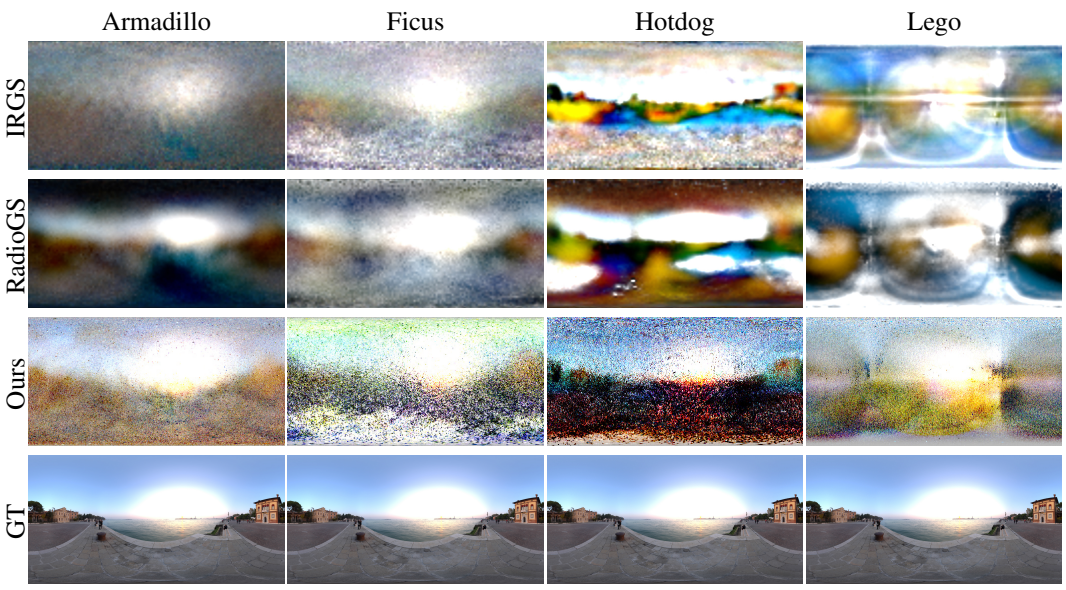

Figure 14: Qualitative comparison of the recovered environment maps on the Armadillo, Ficus, Hotdog, and Lego scenes from the TensoIR dataset. The results of IRGS, RadioGS, our method, and the ground truth (GT) are shown from top to bottom.