INVERSE RENDERING WITHOUT TRAINING FROM SCRATCH: UNSUPERVISED POST-TRAINING OF FEEDFORWARD 3D GAUSSIAN MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Despite the great success of feedforward 3D Gaussian models in scene reconstruction, their learned representations cannot provide editable physical information. While optimization-based inverse rendering methods are able to produce these attributes, they are not compatible to feedforward paradigm. To close this gap, we introduce an unsupervised post-training framework that adapts feedforward 3D Gaussian models to inverse rendering without training a new network from scratch. We first analyze the propagation of geometry errors from off-the-shelf Gaussian scaffold produced by feedforward models to PBR maps, and are motivated to develop a post-splat decoding paradigm. Specifically, the features produced by feedforward 3D Gaussian models are aggregated to construct scene memory. Afterwards, a learnable adapter is introduced to generate view-independent material latents conditioned on the scene memory, which are then decoded into PBR parameters after splatting. Experiments across seven backbones demonstrate that our framework achieves state-of-the-art performance against previous approaches with only 1.5%–7.0% additional inference parameters. Moreover, the ablation experiments validate the effectiveness of our post-splat decoding paradigm and scene memory, while qualitative results on real-world scenes illustrate promising relighting and illumination editing results.

1 INTRODUCTION

Inverse rendering aims at recovering intrinsic scene properties, such as geometry, materials, and lighting, from observed images, thereby providing editable physical representations for applications such as relighting and material editing (Li et al., 2020; Lin et al., 2025; Liang et al., 2025a). To handle this highly ill-posed problem, early approaches decompose it into multiple subproblems (e.g., intrinsic image decomposition and shape from shading) and rely on hand-crafted priors for optimization (Barron & Malik, 2015). However, these priors often break down in complex real-world scenes, leading to unstable or inaccurate decompositions (Liang et al., 2024a; Lin et al., 2025). In recent years, learning-based inverse rendering has achieved significant progress, including image- and video-domain diffusion methods (Zeng et al., 2024; Jin et al., 2024; Liang et al., 2025a). These methods do not directly reconstruct explicit 3D scenes and therefore do not, on their own, provide a scene representation for arbitrary-view rendering (Zeng et al., 2024; Jin et al., 2024; Liang et al., 2025a).

Recent advances in 3D neural representations, ranging from implicit neural fields (Mildenhall et al., 2020; Jin et al., 2023) to explicit Gaussian representations (Kerbl et al., 2023; Shi et al., 2025), have driven inverse rendering from 2D intrinsic decomposition toward joint modeling of scene properties in 3D space. For instance, GS-IR (Liang et al., 2024b) first introduces 3D Gaussian Splatting (3DGS) into inverse rendering, and develops an efficient optimization scheme for normal estimation and a baking-based occlusion model for indirect lighting. Instead of storing indirect lighting either in static 3D Gaussians or additional volumes, GI-GS (Chen et al., 2025) calculates indirect lighting by path tracing based on deferred shading to better capture the dynamic nature of light. Benefiting from iterative refinement, these optimization-based inverse rendering methods produce satisfactory results. However, they require costly per-scene optimization and numerous iterations to converge, limiting their practicality in real-world applications (Jin et al., 2023; Liang et al., 2024b).

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.

Beyond optimization-based 3D reconstruction methods, feedforward models can infer geometry and texture from multi-view images in a single forward pass, substantially reducing the inference cost (Chen et al., 2024a; Jiang et al., 2025). Several pioneering works directly extend this feedforward paradigm to inverse rendering. For instance, MAIR (Choi et al., 2023; 2025) introduces a scene-level multi-view inverse rendering pipeline to utilize multi-view stereo (MVS) depth as well as scene context to estimate SVBRDF. More recently, LIRM (Li et al., 2025) presents a feedforward transformer that jointly reconstructs high-quality shape, materials, and view-dependent radiance fields. Similarly, MVInverse (Wu et al., 2026) introduces a feedforward multi-view inverse rendering framework that directly predicts spatially varying scene properties from sequences of RGB images within a single forward pass.

Despite improved efficiency, existing feedforward inverse rendering methods still suffer two major limitations. *First*, these methods are optimized in a supervised manner and rely on synthetic data or pseudo labels generated by teacher models for training. As a result, the high data collection cost makes them difficult to scale up to real-world scenarios. *Second*, these methods couple material recovery with the geometry used for rendering. When a predicted scaffold is reused without subsequent geometry refinement, errors in projected Gaussian footprints and visibility can affect the predicted PBR parameters. Overall, performing unsupervised inverse rendering without reconstructing a complete relightable representation from scratch is non-trivial and remains underexplored.

To address the aforementioned two limitations, we propose an unsupervised inverse rendering framework compatible with existing feedforward Gaussian splatting methods. In particular, our framework enables existing feedforward methods to recover intrinsic properties without constructing an additional relightable 3D representation from scratch. Instead of predicting explicit per-Gaussian PBR parameters before splatting (pre-splat prediction), our method predicts view-independent material latents on the reusable feedforward scaffold. Then, these latents are splatted onto the image plane and decoded into PBR maps (post-splat decoding). In this way, existing feedforward 3D methods can be efficiently adapted to inverse rendering in an unsupervised manner.

In summary, our main contributions are as follows:

- We introduce an unsupervised post-training paradigm that adapts pretrained feedforward 3D Gaussian models to inverse rendering and relighting. The backbone and prediction heads are frozen with only a learnable adapter being applied on top the predicted Gaussian scaffold.
- We analyze the propagation of geometry errors from Gaussian scaffold to PBR maps and develop a post-splat decoding paradigm for compensation. Specifically, the adapter first transforms backbone features to view-independent material latents in 3D space and decodes PBR parameters conditioned on scene memory after splatting.
- We conduct experiments across seven feedforward Gaussian backbones and the results show our paradigm enables PBR reconstruction with only 1.5%–7.0% additional inference parameters relative to the original backbones. Both quantitative and qualitative evaluation demonstrate the superior performance of our method.

2 Related Works

2.1 Feedforward 3D Reconstruction

Conventional novel-view synthesis and 3D reconstruction methods typically rely on calibrated cameras and then perform per-scene optimization to recover geometry and appearance (Mildenhall et al., 2020; Kerbl et al., 2023). Such pipelines introduce substantial computational cost and limit practical scalability. Recent feedforward visual geometry models reduce this dependence by predicting scene structure directly from images. DUSt3R and MASt3R recover dense 3D point maps and matching-aware geometry from image pairs (Wang et al., 2024; Leroy et al., 2024), while VGGT predicts cameras, depth, point maps, and tracks within a unified transformer framework (Wang et al., 2025a). PI3 further adopts a permutation-equivariant architecture to predict affine-invariant camera poses and scale-invariant local point maps without designating a reference frame (Wang et al., 2025b).

Building on these geometry priors, feedforward reconstruction has evolved from geometry prediction toward ready-to-render Gaussian scene representations. pixelSplat and MVSplat predict 3D Gaussians from sparse posed or paired views for efficient novel-view synthesis (Charatan et al., 2024; Chen et al.,

2024a), while Splatt3R and NoPoSplat relax the dependence on calibrated camera poses for sparse unposed inputs (Smart et al., 2024; Ye et al., 2025b). MVSplat360, DepthSplat, and AnySplat further extend this line to wider-view coverage, depth-aware prediction, and unconstrained unposed image collections (Chen et al., 2024b; Xu et al., 2025; Jiang et al., 2025). Despite remarkable progress, feedforward methods are primarily trained for novel-view synthesis and lack the capability to regress physical properties.

2.2 Inverse Rendering

Inverse rendering recovers geometry, materials, and illumination from images, which is an ill-posed problem under unknown lighting (Li et al., 2020; Wang et al., 2021; Lin et al., 2025). NeRFactor, PhySG, nvdiffrec, TensoIR, and PBR-NeRF optimize neural fields, meshes, or material fields with physical rendering models to obtain relightable scene representations (Zhang et al., 2021b;a; Munkberg et al., 2022; Jin et al., 2023; Wu et al., 2025). GS-IR, GaussianShader, GIR, and GUS-IR recover materials and illumination with 3D Gaussians (Liang et al., 2024b; Jiang et al., 2024; Shi et al., 2025; Liang et al., 2025b), while GI-GS combines deferred shading and lightweight path tracing to model global illumination (Chen et al., 2025). However, these methods require costly per-scene optimization. In addition, for GS-IR and GI-GS, explicit PBR parameters are estimated per Gaussian before splatting, then rasterized into maps for shading. Such pre-splat parameterization makes material buffers sensitive to residual geometry and visibility errors.

Feedforward inverse rendering combines learned geometry priors with material decomposition. MAIR and MAIR++ use multi-view RGB images, MVS depth, and scene context to estimate scene materials and illumination (Choi et al., 2023; 2025). LIRM primarily targets object-centric reconstruction of shape, materials, and view-dependent radiance fields (Li et al., 2025). MVInverse uses cross-view attention to predict intrinsic property maps for the input views (Wu et al., 2026), and it combines these maps with external geometry reconstruction for novel view synthesis instead of producing a unified 3D scene representation that can be queried from novel viewpoints. Different from the aforementioned methods, we post-train a PBR adapter on existing feedforward Gaussian representations through multi-view RGB reconstruction.

3 Preliminaries

3.1 3D Gaussian splatting

3D Gaussian splatting (3DGS) represents a scene with anisotropic Gaussian primitives and renders them through differentiable splatting (Kerbl et al., 2023; Charatan et al., 2024). Given posed images $\mathcal{I}_{\text{posed}} = \{(I_v, T_v, K_v)\}_{v=1}^{V}$, the n-th Gaussian is defined as $\mathcal{G}_n = (\mu_n, \Sigma_n, o_n, \mathbf{h}_n)$, where μ_n, Σ_n, o_n, and $\mathbf{h}_n$ denote its center, covariance, opacity, and attributes, respectively. For target view v, its compositing weight at pixel u is

$$\bar{w}_{n,v}(u) = o_n \mathcal{N}_n^{(v)}(u) \prod_{k<n} \Big(1 - o_k \mathcal{N}_k^{(v)}(u)\Big). \tag{1}$$

RGB images, features, depth maps, and material buffers share the same geometry and visibility support (Chen et al., 2024a;b; Liang et al., 2024b; Chen et al., 2025).

3.2 Deferred Inverse Rendering in 3DGS

3DGS-based inverse rendering augments each Gaussian with geometric attributes and PBR parameters and applies deferred shading to the rasterized buffers (Liang et al., 2024b; Jiang et al., 2024). Let $a_n \in \mathbb{R}^3$ denote albedo and define the 8-D PBR parameter tuple $\mathbf{p}_n = (\mathbf{n}_n, a_n, r_n, m_n) \in \mathbb{R}^8$. With depth d_n, the full per-Gaussian G-buffer tuple is $\mathbf{q}_n = (d_n, \mathbf{p}_n)$. The target-view G-buffer is

$$\mathcal{B}_v(u) = \sum_{n=1}^{N} \bar{w}_{n,v}(u)\mathbf{q}_n = \big(d_v(u), \mathbf{n}_v(u), a_v(u), r_v(u), m_v(u)\big). \tag{2}$$

The rendered color is then obtained by

$$\hat{I}_v(u) = S_{\text{pbr}}(\mathcal{B}_v(u), \mathcal{E}, \boldsymbol{\omega}_o(u, v)), \tag{3}$$

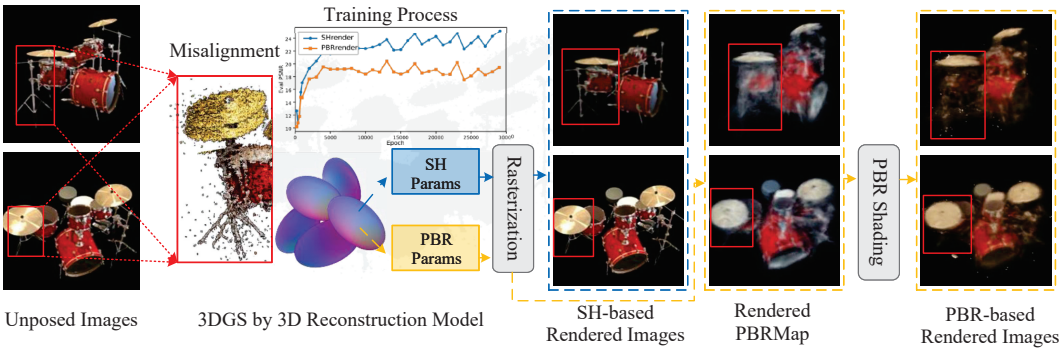


Figure 1: Motivation of framework. Near geometric misalignment, SH rendering retains higher reconstruction quality, whereas PBR parameter estimation yields blurred maps and degraded renderings.

where S_{pbr} is a differentiable PBR shader, $\mathcal{E}$ is the environment illumination, and $\boldsymbol{\omega}_o$ is the outgoing view direction (Cook & Torrance, 1982; Karis, 2013).

4 METHOD

4.1 MOTIVATION

Training a feed-forward inverse-rendering network from scratch requires dense PBR annotations at scale, which are expensive to acquire. We instead adapt pretrained feed-forward 3DGS models by freezing their original prediction pathways and introducing a scene-conditioned PBR adapter. To study the effect of residual geometry under a fixed Gaussian scaffold, let $\mathbf{r}$ denote the stacked multi-view RGB residual, and its first-order decomposition around a reference solution is

$$\mathbf{r} = A_g e_g + A_p \Delta p + A_{\mathcal{E}} \Delta \mathcal{E} + R_2, \tag{4}$$

Appendix D derives Eq. (4) from a first-order expansion of the differentiable renderer. Here, e_g is the scaffold geometry error, A_g, A_p, and $A_{\mathcal{E}}$ are the local image-space Jacobians with respect to geometry, pixel-aligned PBR maps, and environment illumination, Δp and $\Delta \mathcal{E}$ are the corresponding changes in PBR parameters and illumination, and R_2 collects higher-order terms. Under this fixed-geometry condition, material and illumination updates can reduce only the component of $A_g e_g$ that lies in the joint local range of A_p and $A_{\mathcal{E}}$.

Geometry errors perturb projections and alpha-compositing weights (Kerbl et al., 2023; Radl et al., 2024), affecting PBR maps and rendering. Figure 1 illustrates the deviation between SH rendering and PBR rendering. As we can see, both routes predict per-Gaussian attributes before splatting (pre-splat), one predicts SH coefficients, and the other predicts explicit PBR parameters that are rasterized into maps for shading. SH route directly fits appearance, allowing partial compensation for geometry errors. In contrast, PBR route aggregates explicit parameters before nonlinear shading, potentially losing information needed to distinguish mixtures with different shading responses. To remedy this, our post-training strategy therefore splats and aggregates per-Gaussian material latents, then decodes PBR maps with scene context (post-splat decoding). Section 4.4 analyzes this bottleneck.

4.2 OUR FRAMEWORK

As illustrated in Fig. 2, we adapt a pretrained feedforward 3DGS model to inverse rendering from a set of context images. The framework comprises multi-view Gaussian encoding, scene latent generation, scene-conditioned post-training adaptation, and PBR shading.

Multi-view Gaussian Encoding. The pretrained 3DGS model extracts multi-layer tokens with cross-view context and predicts initial Gaussian positions and attributes. These tokens support material feature prediction, while the initial Gaussians provide a unified 3D representation. During post-training, the pretrained backbone and its geometry and camera heads remain frozen, while the added modules are optimized using multi-view context images as supervision.

Scene Latent Generation. We construct a scene memory from multi-view tokens to provide global and local latents for material and illumination prediction. The global scene latent conditions

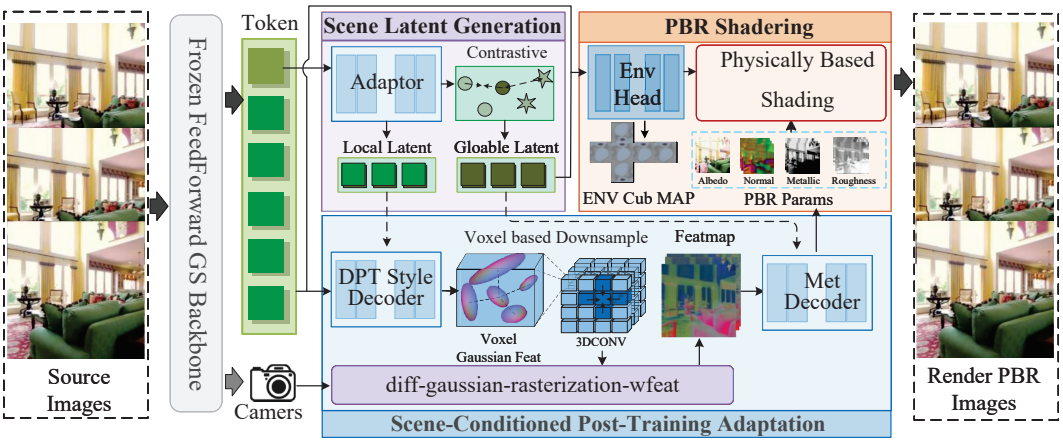


Figure 2: Overview of the framework. Scene latent generation provides global and local scene context. Scene-conditioned post-training adaptation fuses multi-view features and predicts PBR parameters. PBR shading combines PBR parameters and illumination for image rendering and relighting.

material decoding and illumination estimation, while the local latents condition dense material feature prediction for their respective context views. The illumination predictor also accesses the complete scene memory to incorporate multi-view information. A scene-level contrastive regularizer encourages consistent and discriminative global representations, with an exponential moving average (EMA) technique facilitating stable optimization. The training objective is described in Section 4.3.

Scene-Conditioned Post-Training Adaptation. We learn material features on the Gaussian representation supplied by the pretrained model. On top of multi-level image features, a separate DPT decoder combines these multi-view tokens, input-image cues, and local scene latents to predict dense material features. Voxel fusion consolidates material features from different views at shared 3D locations, establishing a common representation for adaptation. Sparse 3D convolutions then propagate the fused information across occupied voxels and their neighbors, allowing Gaussian attributes and material features to be refined with local 3D context. Afterwards, target-view feature maps are produced by splatting the fused material features onto the image plane. A shared material decoder then predicts PBR parameters from these maps, using the global scene latent to interpret the aggregated pixel features in their scene context. The computations for scene latent generation and post-training adaptation are given in Appendix G, with the EMA and contrastive formulations in Appendix I.2.

PBR Shading. The illumination predictor combines target-view feature maps with the complete scene memory and the global scene condition to predict a shared environment cubemap. A differentiable PBR shader then uses the predicted PBR parameters and illumination to render target images. Multi-view RGB reconstruction supervises the adapter through this rendering process.

4.3 Training Objectives

PBR Image Reconstruction. We supervise PBR rendering with a masked L1–SSIM loss (Liang et al., 2025b):

$$\mathcal{L}_{\text{PBR}} = (1 - \lambda_{\text{ssim}})\mathcal{L}_1^M\left(\hat{I}_v^{\text{pbr}}, I_v\right) + \lambda_{\text{ssim}}\mathcal{L}_{\text{SSIM}}^M\left(\hat{I}_v^{\text{pbr}}, I_v\right). \tag{5}$$

Material and Normal Regularization. We apply edge-aware smoothness to the predicted material maps and normals. We also enforce consistency between shading normals and depth-derived normals:

$$\mathcal{L}_{\text{struct}} = \lambda_{\text{mat}}\mathcal{L}_{\text{matTV}} + \lambda_{\text{ncons}}\mathcal{L}_{\text{ncons}} + \lambda_{\text{ntv}}\mathcal{L}_{\text{nTV}}. \tag{6}$$

Here, $\mathcal{L}_{\text{matTV}}$ regularizes albedo, roughness, and metallicity, $\mathcal{L}_{\text{ncons}}$ enforces normal consistency, and $\mathcal{L}_{\text{nTV}}$ smooths normals within valid interior regions.

Diffuse–Specular Shading Balance. To reduce ambiguity between diffuse and specular shading (Chung et al., 2024; Wu et al., 2025), we regularize their image-space contributions:

$$\mathcal{L}_{\text{shade}} = \frac{\lambda_{\log}}{N_\Omega + \epsilon} \sum_v \sum_{u \in \Omega_v} \eta_v^d(u) \times \left| \log\big(1 + \ell_v^d(u) + \ell_v^s(u)\big) - \log(1 + y_v(u)) \right| + \lambda_{\text{ratio}} \frac{\sum_v \sum_{u \in \Omega_v} \ell_v^s(u)}{\sum_v \sum_{u \in \Omega_v} \ell_v^d(u) + \epsilon}. \tag{7}$$

Here, $\ell_v^d(u)$ and $\ell_v^s(u)$ are scalar diffuse and specular intensities, $y_v(u)$ is a target-intensity proxy, $\eta_v^d(u)$ is the diffuse fraction, Ω_v is the valid coverage region, and $N_\Omega = \sum_v |\Omega_v|$. The coefficients $\lambda_{\log}$ and λ_{ratio} are loss weights, and $\epsilon > 0$ provides numerical stability. The first term matches total shading to the target in diffuse-dominant regions, while the second suppresses excessive specular energy. Exact reductions are provided in Appendix I.

Global Scene Latent Contrastive Learning. We regularize global scene latents using a scene-level InfoNCE objective (He et al., 2020), which is detailed in Appendix I.2.

Overall, the training objective is $\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{PBR}} + \mathcal{L}_{\text{struct}} + \mathcal{L}_{\text{shade}} + \lambda_{\text{scene}}\mathcal{L}_{\text{scene}}$.

4.4 Geometry-to-PBR Propagation and Post-Splat Decoding

Geometry errors change Gaussian blending weights and hence the mixtures used for material prediction. Let $\mathbf{w}(g)$ collect the compositing weights and $\mathbf{M}$ denote the predicted PBR maps. Holding contributor attributes, decoder parameters, and scene conditioning fixed, the local response in a differentiable neighborhood with unchanged contributors and depth order is

$$\delta\mathbf{M} = \mathbf{J}_{\mathbf{M},\mathbf{w}}\mathbf{J}_{\mathbf{w},g}\,\delta g + o(\|\delta g\|). \tag{8}$$

Appendix D derives this geometry-to-material path and Appendix F separates coverage changes from weight redistribution and gives local response bounds.

For simplicity of analysis, we study residual geometry errors locally, keeping the contributors, depth order, and total coverage fixed while redistributing blending weights. Pre-splat blends explicit PBR parameters, whereas post-splat blends latent features before decoding. Under nonlinear shading, mixtures with identical PBR means may produce different averaged shading responses (Pharr et al., 2024). The shared 8-D statistic T_8 cannot distinguish such mixtures, but additional feature statistics can retain the differences (Wagstaff et al., 2022; Amir et al., 2023). To characterize this capacity, consider an extended statistic T_+ that retains T_8. Let $\mathcal{R}^*_{\text{PBR}}(T)$ denote the minimum expected squared RGB error over valid material decoders, with the scene conditions and PBR shader held fixed. Then

$$T_8 = \Pi T_+ \quad \Longrightarrow \quad \mathcal{R}^*_{\text{PBR}}(T_+) \le \mathcal{R}^*_{\text{PBR}}(T_8). \tag{9}$$

Extracting T_8 from T_+ and applying the original decoder leaves its prediction unchanged, while the additional statistics allow different material decisions. Appendix E proves this relation and gives a GGX example showing how a retained roughness moment improves rendering. Section 5.4 evaluates the trained models through decoder-placement ablation.

5 Experiments

5.1 Experimental Setup

Datasets and implementation details. Since our approach adapts pretrained feedforward Gaussian backbones through post-training, we use RealEstate10K (RE10K) (Zhou et al., 2018), a real-scene benchmark used by feedforward Gaussian reconstruction methods (Chen et al., 2024a; Jiang et al., 2025), to assess cross-backbone adaptation. We use 6,128 scenes for training and randomly sample 200 scenes from the test split for evaluation. Training context views follow AnySplat's temporal sampling protocol (Jiang et al., 2025). Each backbone is initialized from its pretrained checkpoint, and then a separate PBR adapter is trained. Architecture, rendering, regularization, optimization, precision, and hardware details are provided in Appendix H.

Backbones and metrics. For cross-backbone evaluation, we integrate our adapter with different feedforward Gaussian pipelines, including AnySplat (Jiang et al., 2025), Depth Anything 3 (DA3) (Lin

Table 1: Native-versus-PBR target-view reconstruction quality, backbone parameters, and added inference parameters across seven feedforward Gaussian backbones on RealEstate10K. We report PSNR, SSIM, and LPIPS under the Multi Views and Sparse Views protocols.

Backbone	Venue	Variant	Params (M)	Multi Views			Sparse Views		
				PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
Pose-Free Context-Camera Integrations									
DA3	ICLR	Vanilla	1689.85	25.31	0.848	0.168	N/A	N/A	N/A
504 × 504	2026	**Ours-PBR**	**+42.00**	**25.55**	**0.860**	**0.158**	N/A	N/A	N/A
AnySplat	TOG	Vanilla	1190.74	25.32	0.849	0.168	19.80	0.680	0.274
448 × 448	2025	**Ours-PBR**	**+47.76**	**25.60**	**0.861**	**0.159**	**20.12**	**0.683**	**0.271**
SPFSplat	ICCV	Vanilla	616.02	22.95	0.781	0.165	**23.93**	**0.799**	**0.153**
256 × 256	2025	**Ours-PBR**	**+43.06**	**23.83**	**0.790**	**0.154**	23.15	0.788	0.178
GT Context-Camera Integrations									
C3G	CVPR	Vanilla	1013.14	19.69	0.611	**0.295**	19.80	0.618	**0.290**
224 × 224	2026	**Ours-PBR**	**+15.49**	**19.98**	**0.623**	0.327	**20.01**	**0.629**	0.325
FLARE	CVPR	Vanilla	1424.56	N/A	N/A	N/A	21.06	0.700	0.311
256 × 256	2025	**Ours-PBR**	**+36.38**	N/A	N/A	N/A	**23.20**	**0.793**	**0.260**
NoPoSplat	ICLR	Vanilla	612.00	18.66	0.595	0.310	20.46	**0.741**	**0.189**
256 × 256	2025	**Ours-PBR**	**+43.06**	**20.37**	**0.657**	**0.295**	**21.05**	0.713	0.228
YoNoSplat	ICLR	Vanilla	964.98	22.95	0.781	0.165	21.47	0.712	0.207
224 × 224	2026	**Ours-PBR**	**+43.76**	**23.83**	**0.790**	**0.154**	**21.79**	**0.713**	**0.206**

Vanilla and Ours-PBR denote the native appearance and PBR-adapted outputs, respectively. Pose-Free integrations obtain context-camera poses from the feedforward backbone during training and testing, whereas GT integrations use ground-truth context-camera poses in both stages. Bold marks the better metric within each matched pair, and gray N/A denotes an unavailable result.



Figure 3: Sparse-view qualitative comparison on RE10K with two input views. Matched crops compare Vanilla and Ours-PBR across four backbones, the target RGB image marks the crop region. Full-scene comparisons under both protocols appear in Appendix A.

et al., 2026), SPFSplat (Huang & Mikolajczyk, 2025), C3G (An et al., 2026), FLARE (Zhang et al., 2025), NoPoSplat (Ye et al., 2025b), and YoNoSplat (Ye et al., 2025a). We evaluate target-view reconstruction with PSNR, SSIM (Wang et al., 2004), and LPIPS (Zhang et al., 2018). RE10K provides neither ground-truth illumination nor images of the same scene under specified target lighting, so we assess relighting qualitatively, as in the real-scene relighting visualizations of GI-GS (Chen et al., 2025).

5.2 Cross-Backbone Adaptation and Appearance Comparison

We train a separate adapter for each backbone with 10–20 context views and three target views, except for FLARE, whose setup requires sparse-view training. We then compare target-view PBR renderings with the native 3DGS-based renderings under multi-view and sparse-view evaluation protocols. These protocols use 10–20 and two context views, respectively, and each evaluates three target views (Jiang et al., 2025; Ye et al., 2025b). Within each backbone and view setting, Vanilla and Ours-PBR use the same target frames and evaluation protocol.

Reconstruction fidelity. Table 1 evaluates whether PBR adaptation preserves each backbone's native rendering quality. Across the 12 comparable backbone–protocol pairs, Ours-PBR achieves higher PSNR in 11 pairs and higher SSIM in 10 pairs, with setting-equal average gains of approximately $+0.59$ dB and $+0.015$, respectively. The average gains remain positive under both protocols. Training

the SPFSplat adapter with 10–20 context views rather than two may contribute to its lower sparse-view reconstruction scores relative to its Vanilla renderer. C3G's limited Gaussian count (An et al., 2026) may constrain the spatial detail available for PBR shading, despite its PSNR and SSIM gains. The matched comparisons in Figure 3 and Appendix A further show that Ours-PBR preserves the scene layout, dominant colors, and major object boundaries of the native renderings. Overall, our framework equips seven feedforward Gaussian backbones with decomposed PBR representations while largely preserving their target-view reconstruction quality.

Adaptation overhead. Across the seven backbones, the adapter adds 15.49M–47.76M inference parameters, corresponding to 1.5%–7.0% of the original backbone parameters. Table 1 reports the overhead for each backbone. By reusing the Gaussian scaffold and feature representations of pretrained reconstruction models, our method enables PBR rendering and relighting with a small parameter overhead.

5.3 Qualitative Relighting on Real-World Scenes

Figure 4 compares our SPFSplat PBR variant with GI-GS (Chen et al., 2025), GS-IR (Liang et al., 2024b), LumiGauss (Kaleta et al., 2025), and R3GW (Corona et al., 2026) on a held-out outdoor RE10K scene under shared Studio and Sunset HDR environment maps. Appendix B provides the full indoor and outdoor comparisons with source-light observations and reconstructions. These results assess illumination response and scene-content retention under target lighting.

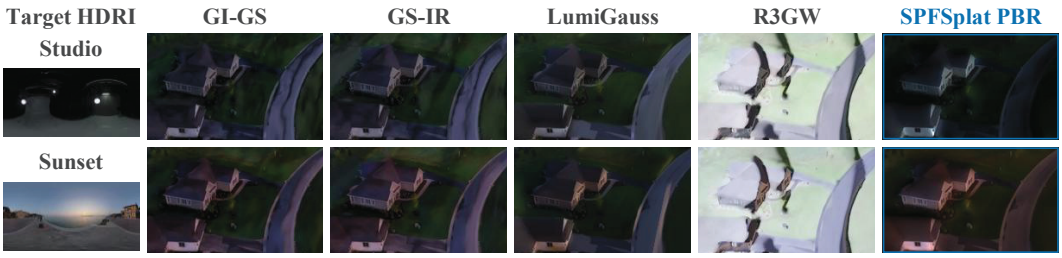


Figure 4: Qualitative relighting of an outdoor RE10K scene under Studio and Sunset illumination. The first column shows the target HDRI inputs; all methods use the same image crop. Full indoor and outdoor comparisons, including source-light reconstructions, appear in Appendix B.

Across both scenes, the SPFSplat PBR variant changes the lighting condition from the dim Studio setting to the warmer Sunset environment while retaining the indoor cabinet boundaries and outdoor road layout. The baselines also respond, with varying smoothing, contrast, and exposure. LumiGauss and R3GW are cross-domain diagnostics indoors because both target outdoor captures. Overall, these real-world relighting results indicate that our method equips pretrained feed-forward Gaussian backbones with PBR capability, without retraining them from scratch.

5.4 Ablation Study

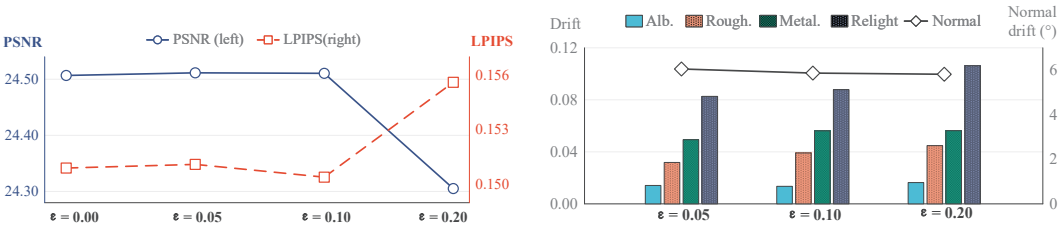
We conduct two ablation experiments. Table 2 isolates the effects of post-splat decoding and the scene latent on PBR reconstruction with a frozen Gaussian scaffold. Direct-PBR and Bottleneck-Direct serve as pre-splat approaches, with explicit PBR parameters predicted for each Gaussian before splatting. In contrast, ours variants instead decode PBR maps from splatted latent features, following the post-splat route. Figure 5 measures how geometry perturbations on Gaussian centers affect Direct-PBR predictions. Panel 5a reports the source-light RGB response, and panel 5b reports PBR parameters and relighting drift, comparing each adapter on its training geometry with the unperturbed run. Appendix C defines the metrics and reports the exact measurements.

Component ablation. Bottleneck-Direct remains close to Direct-PBR (24.49/0.812/0.148 vs. 24.51/0.810/0.151) despite introducing a 64-D latent. A 64-D latent alone does not change the rasterized statistic because it is decoded to an explicit 8-D PBR tuple before splatting. Keeping the 64-D representation through splatting and decoding it in image space, still without the scene latent, raises PSNR to 24.62 dB with only 0.054M added parameters. This gain therefore comes from

Table 2: Component ablation on RealEstate10K. The variants isolate latent width, post-splat decoding, and the scene latent on a frozen YoNoSplat scaffold.

Variant	8-D head	64-D latent	Pre-splat	Post-splat	Scene latent	Added Params (M)	Clean source-light RGB		
							PSNR↑	SSIM↑	LPIPS↓
Direct-PBR	✓	–	✓	×	×	ref.	24.51	0.810	0.151
Bottleneck-Direct	×	✓	✓	×	×	+0.009	24.49	0.812	0.148
Ours w/o Scene Latent	×	✓	×	✓	×	+0.054	24.62	0.812	0.148
Ours (Full)	×	✓	×	✓	✓	+11.566	**24.82**	**0.813**	**0.146**

Random context–target sampling can produce values different from the YoNoSplat results in Table 1.



(a) Source-light RGB

(b) PBR Params drift

Figure 5: Response of Direct-PBR to controlled center perturbations. Source-light PSNR and LPIPS remain nearly unchanged through $\epsilon = 0.10$, then degrade at $\epsilon = 0.20$ (a). PBR params and relighting predictions on G_ϵ drift from the $\epsilon = 0$ adapter's predictions on G_0, with normal drift measured in degrees on the right axis (b). Appendix Table 3 reports the exact measurements.

post-splat decoding rather than latent width alone. Adding the scene latent further improves all three metrics to 24.82 dB, 0.813, and 0.146, showing that scene-level information complements post-splat material decoding and illumination estimation.

Motivation experiment. Figure 5 compares source-light reconstruction stability with the sensitivity of PBR predictions to geometry perturbations. Under mild perturbations ($0 < \epsilon \leq 0.10$), source-light PSNR and SSIM change by less than 0.01 dB and less than 0.001, with nonzero PBR and relighting drifts. When $\epsilon = 0.10$, endpoint albedo L1, roughness RMSE, and relighting LPIPS are 0.014, 0.039, and 0.088, respectively. When $\epsilon = 0.20$, relighting drift reaches 0.106, while source-light PSNR falls to 24.31 dB and LPIPS rises to 0.156. Consequently, the final PBR predictions can differ under perturbed training and evaluation geometry even when source-light reconstruction remains nearly unchanged. Appendix Table 3 reports the complete perturbation sweep at full precision.

6 Conclusion

We presented an unsupervised post-training framework for inverse rendering that reuses pretrained feedforward 3D Gaussian models without training a new reconstruction network from scratch. The original backbone and prediction heads remain frozen, while the added modules learn to predict materials and illumination solely from multi-view RGB images. Our analysis of geometry-to-PBR propagation motivates decoding PBR parameters from aggregated Gaussian features. Experiments across seven backbones show that our adapters largely preserve target-view reconstruction quality under multi-view and sparse-view settings, with only 1.5%–7.0% additional inference parameters relative to the original backbones. Ablations show that post-splat decoding and the scene latent improve PBR reconstruction.

AI use statement

We used AI for language polishing and grammar correction of the manuscript. All content has been reviewed and verified by the authors.

Reproducibility statement

We provide the information needed for reproduction in the main paper and the appendix, the data and evaluation protocol are described in Sec. 5.1, and the model, training, and optimization details are given in Appendix G, Appendix H, and Appendix I. In addition, the supplementary material provides test samples together with their corresponding relighting results.

References

Tal Amir, Steven J. Gortler, Ilai Avni, Ravina Ravina, and Nadav Dym. Neural injective functions for multisets, measures and graphs via a finite witness theorem. In *Advances in Neural Information Processing Systems*, volume 36, pp. 42516–42551, 2023. doi: 10.52202/075280-1845. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/84b686f7cc7b7751e9aaac0da74f755a-Abstract-Conference.html.

Honggyu An, Jaewoo Jung, Mungyeom Kim, Chaehyun Kim, Minkyeong Jeon, Jisang Han, Kazumi Fukuda, Takuya Narihira, Hyunah Ko, Junsu Kim, Sunghwan Hong, Yuki Mitsufuji, and Seungryong Kim. Learning compact 3D representations from feed-forward novel view synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 363–373, 2026. URL https://openaccess.thecvf.com/content/CVPR2026/html/An_Learning_Compact_3D_Representations_from_Feed-Forward_Novel_View_Synthesis_CVPR_2026_paper.html.

Jonathan T. Barron and Jitendra Malik. Shape, illumination, and reflectance from shading. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(8):1670–1687, 2015. doi: 10.1109/TPAMI.2014.2377712. URL https://doi.org/10.1109/TPAMI.2014.2377712.

David Charatan, Sizhe Li, Andrea Tagliasacchi, and Vincent Sitzmann. pixelSplat: 3D gaussian splats from image pairs for scalable generalizable 3D reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 19457–19467, 2024. doi: 10.1109/CVPR52733.2024.01840. URL https://doi.org/10.1109/CVPR52733.2024.01840.

Hongze Chen, Zehong Lin, and Jun Zhang. GI-GS: Global illumination decomposition on Gaussian splatting for inverse rendering. In *International Conference on Learning Representations*, 2025. doi: 10.48550/arXiv.2410.02619. URL https://arxiv.org/abs/2410.02619.

Yuedong Chen, Haofei Xu, Chuanxia Zheng, Bohan Zhuang, Marc Pollefeys, Andreas Geiger, Tat-Jen Cham, and Jianfei Cai. MVSplat: Efficient 3D gaussian splatting from sparse multi-view images. In *European Conference on Computer Vision*, volume 15079 of *Lecture Notes in Computer Science*, pp. 370–386, 2024a. doi: 10.1007/978-3-031-72664-4_21. URL https://doi.org/10.1007/978-3-031-72664-4_21.

Yuedong Chen, Chuanxia Zheng, Haofei Xu, Bohan Zhuang, Andrea Vedaldi, Tat-Jen Cham, and Jianfei Cai. MVSplat360: Feed-forward 360 scene synthesis from sparse views. In *Advances in Neural Information Processing Systems*, volume 37, pp. 107064–107086, 2024b. doi: 10.52202/079017-3399. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/c196239c5f9481e0db2755f31fe4585f-Abstract-Conference.html.

JunYong Choi, SeokYeong Lee, Haesol Park, Seung-Won Jung, Ig-Jae Kim, and Junghyun Cho. MAIR: Multi-view attention inverse rendering with 3D spatially-varying lighting estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8392–8401, 2023. doi: 10.1109/CVPR52729.2023.00811. URL https://doi.org/10.1109/CVPR52729.2023.00811.

JunYong Choi, SeokYeong Lee, Haesol Park, Seung-Won Jung, Ig-Jae Kim, and Junghyun Cho. MAIR++: Improving multi-view attention inverse rendering with implicit lighting representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 47(6):5076–5093, 2025. doi: 10.1109/TPAMI.2025.3548679. URL https://doi.org/10.1109/TPAMI.2025.3548679.

Hoon-Gyu Chung, Seokjun Choi, and Seung-Hwan Baek. Differentiable point-based inverse rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4399–4409, 2024. doi: 10.1109/CVPR52733.2024.00421. URL https://doi.org/10.1109/CVPR52733.2024.00421.

Robert L. Cook and Kenneth E. Torrance. A reflectance model for computer graphics. *ACM Transactions on Graphics (TOG)*, 1(1):7–24, 1982. doi: 10.1145/357290.357293.

Margherita Lea Corona, Wieland Morgenstern, Peter Eisert, and Anna Hilsmann. R3GW: Relightable 3D Gaussians for outdoor scenes in the wild. In *Proceedings of the 21st International Conference on Computer Vision Theory and Applications – Volume 3: VISAPP*, pp. 432–443. SCITEPRESS, 2026. doi: 10.5220/0014332200004084. URL https://doi.org/10.5220/0014332200004084.

Thomas M. Cover and Joy A. Thomas. *Elements of Information Theory*. John Wiley & Sons, 2 edition, 2006. doi: 10.1002/047174882X. URL https://doi.org/10.1002/047174882X.

Alban Gauthier, Robin Faury, Jérémy Levallois, Théo Thonat, Jean-Marc Thiery, and Tamy Boubekeur. MIPNet: Neural normal-to-anisotropic-roughness MIP mapping. *ACM Transactions on Graphics*, 41(6), 2022. doi: 10.1145/3550454.3555487. URL https://doi.org/10.1145/3550454.3555487.

Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9729–9738, 2020. doi: 10.1109/CVPR42600.2020.00975. URL https://doi.org/10.1109/CVPR42600.2020.00975.

Ranran Huang and Krystian Mikolajczyk. No pose at all: Self-supervised pose-free 3D gaussian splatting from sparse views. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 27947–27957, 2025. doi: 10.1109/ICCV51701.2025.02594. URL https://doi.org/10.1109/ICCV51701.2025.02594.

Lihan Jiang, Yucheng Mao, Linning Xu, Tao Lu, Kerui Ren, Yichen Jin, Xudong Xu, Mulin Yu, Jiangmiao Pang, Feng Zhao, Dahua Lin, and Bo Dai. AnySplat: Feed-forward 3D gaussian splatting from unconstrained views. *ACM Transactions on Graphics*, 44(6):1–16, 2025. doi: 10.1145/3763326. URL https://doi.org/10.1145/3763326.

Yingwenqi Jiang, Jiadong Tu, Yuan Liu, Xifeng Gao, Xiaoxiao Long, Wenping Wang, and Yuexin Ma. GaussianShader: 3D gaussian splatting with shading functions for reflective surfaces. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5322–5332, 2024. doi: 10.1109/CVPR52733.2024.00509. URL https://doi.org/10.1109/CVPR52733.2024.00509.

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, 2023. doi: 10.1109/CVPR52729.2023.00024. URL https://doi.org/10.1109/CVPR52729.2023.00024.

Haian Jin, Yuan Li, Fujun Luan, Yuanbo Xiangli, Sai Bi, Kai Zhang, Zexiang Xu, Jin Sun, and Noah Snavely. Neural gaffer: Relighting any object via diffusion. In *Advances in Neural Information Processing Systems*, volume 37, pp. 141129–141152, 2024. doi: 10.52202/079017-4481. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/ff7373914a96956f2a7cacbdf3b0b8d8-Abstract-Conference.html.

Joanna Kaleta, Kacper Kania, Tomasz Trzciński, and Marek Kowalski. LumiGauss: Relightable Gaussian splatting in the wild. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 1–10, 2025. doi: 10.1109/WACV61041.2025.00011. URL https://doi.org/10.1109/WACV61041.2025.00011.

Brian Karis. Real shading in Unreal Engine 4. In *SIGGRAPH Physically Based Shading Course*, 2013.

Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3D Gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics*, 42(4), 2023. doi: 10.1145/3592433. URL https://arxiv.org/abs/2308.04079.

Shuichang Lai, Letian Huang, Jie Guo, Kai Cheng, Bowen Pan, Xiaoxiao Long, Jiangjing Lyu, Chengfei Lv, and Yanwen Guo. GlossyGS: Inverse rendering of glossy objects with 3D gaussian splatting. *IEEE Transactions on Visualization and Computer Graphics*, 31(10):7478–7491, 2025. doi: 10.1109/TVCG.2025.3547063. URL https://doi.org/10.1109/TVCG.2025.3547063.

Vincent Leroy, Yohann Cabon, and Jerome Revaud. Grounding image matching in 3D with MASt3R. In *European Conference on Computer Vision*, volume 15130 of *Lecture Notes in Computer Science*, pp. 71–91, 2024. doi: 10.1007/978-3-031-73220-1_5. URL https://doi.org/10.1007/978-3-031-73220-1_5.

Zhengqin Li, Mohammad Shafiei, Ravi Ramamoorthi, Kalyan Sunkavalli, and Manmohan Chandraker. Inverse rendering for complex indoor scenes: Shape, spatially-varying lighting and SVBRDF from a single image. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 2475–2484, 2020. doi: 10.1109/CVPR42600.2020.00255. URL https://doi.org/10.1109/CVPR42600.2020.00255.

Zhengqin Li, Dilin Wang, Ka Chen, Zhaoyang Lv, Thu Nguyen-Phuoc, Milim Lee, Jia-Bin Huang, Lei Xiao, Yufeng Zhu, Carl S. Marshall, Yuheng Ren, Richard Newcombe, and Zhao Dong. LIRM: Large inverse rendering model for progressive reconstruction of shape, materials and view-dependent radiance fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*

(CVPR), pp. 505–517, 2025. doi: 10.1109/CVPR52734.2025.00056. URL https://openaccess.thecvf.com/content/CVPR2025/html/Li_LIRM_Large_Inverse_Rendering_Model_for_Progressive_Reconstruction_of_Shape_CVPR_2025_paper.html.

Ruofan Liang, Zan Gojcic, Merlin Nimier-David, David Acuna, Nandita Vijaykumar, Sanja Fidler, and Zian Wang. Photorealistic object insertion with diffusion-guided inverse rendering. In *European Conference on Computer Vision*, volume 15119 of *Lecture Notes in Computer Science*, pp. 446–465, 2024a. doi: 10.1007/978-3-031-73030-6_25. URL https://doi.org/10.1007/978-3-031-73030-6_25.

Ruofan Liang, Zan Gojcic, Huan Ling, Jacob Munkberg, Jon Hasselgren, Chih-Hao Lin, Jun Gao, Alexander Keller, Nandita Vijaykumar, Sanja Fidler, and Zian Wang. DiffusionRenderer: Neural inverse and forward rendering with video diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 26069–26080, 2025a. doi: 10.1109/CVPR52734.2025.02428. URL https://doi.org/10.1109/CVPR52734.2025.02428.

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, 2024b. doi: 10.1109/CVPR52733.2024.02045. URL https://doi.org/10.1109/CVPR52733.2024.02045.

Zhihao Liang, Hongdong Li, Kui Jia, Kailing Guo, and Qi Zhang. GUS-IR: Gaussian splatting with unified shading for inverse rendering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 47(10): 8364–8378, 2025b. doi: 10.1109/TPAMI.2025.3578416. URL https://doi.org/10.1109/TPAMI.2025.3578416.

Chih-Hao Lin, Jia-Bin Huang, Zhengqin Li, Zhao Dong, Christian Richardt, Tuotuo Li, Michael Zollhöfer, Johannes Kopf, Shenlong Wang, and Changil Kim. IRIS: Inverse rendering of indoor scenes from low dynamic range images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 465–474, 2025. doi: 10.1109/CVPR52734.2025.00052. URL https://doi.org/10.1109/CVPR52734.2025.00052.

Haotong Lin, Sili Chen, Jun Hao Liew, Donny Y. Chen, Zhenyu Li, Yang Zhao, Sida Peng, Hengkai Guo, Xiaowei Zhou, Guang Shi, Jiashi Feng, and Bingyi Kang. Depth anything 3: Recovering the visual space from any views. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=yirunib818.

Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. NeRF: Representing scenes as neural radiance fields for view synthesis. In *European Conference on Computer Vision*, volume 12346 of *Lecture Notes in Computer Science*, pp. 405–421, 2020. doi: 10.1007/978-3-030-58452-8_24. URL https://doi.org/10.1007/978-3-030-58452-8_24.

Jacob Munkberg, Jon Hasselgren, Tianchang Shen, Jun Gao, Wenzheng Chen, Alex Evans, Thomas Müller, and Sanja Fidler. Extracting triangular 3D models, materials, and lighting from images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8280–8290, 2022. doi: 10.1109/CVPR52688.2022.00810. URL https://doi.org/10.1109/CVPR52688.2022.00810.

Matt Pharr, Bartłomiej Wroński, Marco Salvi, and Marcos Fajardo. Filtering after shading with stochastic texture filtering. *Proceedings of the ACM on Computer Graphics and Interactive Techniques*, 7(1), 2024. doi: 10.1145/3651293. URL https://doi.org/10.1145/3651293.

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):1–17, 2024. doi: 10.1145/3658187. URL https://doi.org/10.1145/3658187.

René Ranftl, Alexey Bochkovskiy, and Vladlen Koltun. Vision transformers for dense prediction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 12179–12188, 2021. doi: 10.1109/ICCV48922.2021.01196. URL https://openaccess.thecvf.com/content/ICCV2021/html/Ranftl_Vision_Transformers_for_Dense_Prediction_ICCV_2021_paper.html.

Yahao Shi, Yanmin Wu, Chenming Wu, Xing Liu, Chen Zhao, Haocheng Feng, Jian Zhang, Bin Zhou, Errui Ding, and Jingdong Wang. GIR: 3D gaussian inverse rendering for relightable scene factorization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 1–12, 2025. doi: 10.1109/TPAMI.2025.3575937. URL https://doi.org/10.1109/TPAMI.2025.3575937. Early access.

Brandon Smart, Chuanxia Zheng, Iro Laina, and Victor Adrian Prisacariu. Splatt3R: Zero-shot gaussian splatting from uncalibrated image pairs. *arXiv preprint arXiv:2408.13912*, 2024. doi: 10.48550/arXiv.2408.13912. URL https://arxiv.org/abs/2408.13912.

Edward Wagstaff, Fabian B. Fuchs, Martin Engelcke, Michael A. Osborne, and Ingmar Posner. Universal approximation of functions on sets. *Journal of Machine Learning Research*, 23(151):1–56, 2022. URL https://www.jmlr.org/papers/v23/21-0730.html.

Bruce Walter, Stephen R. Marschner, Hongsong Li, and Kenneth E. Torrance. Microfacet models for refraction through rough surfaces. In Jan Kautz and Sumanta Pattanaik (eds.), *Proceedings of the Eurographics Symposium on Rendering*, pp. 195–206. The Eurographics Association, 2007. doi: 10.2312/EGWR/EGSR07/195-206. URL https://doi.org/10.2312/EGWR/EGSR07/195-206.

Jianyuan Wang, Minghao Chen, Nikita Karaev, Andrea Vedaldi, Christian Rupprecht, and David Novotny. VGGT: Visual geometry grounded transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5294–5306, 2025a. doi: 10.1109/CVPR52734.2025.00499. URL https://doi.org/10.1109/CVPR52734.2025.00499.

Shuzhe Wang, Vincent Leroy, Yohann Cabon, Boris Chidlovskii, and Jerome Revaud. DUSt3R: Geometric 3D vision made easy. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 20697–20709, 2024. doi: 10.1109/CVPR52733.2024.01956. URL https://doi.org/10.1109/CVPR52733.2024.01956.

Yifan Wang, Jianjun Zhou, Haoyi Zhu, Wenzheng Chang, Yang Zhou, Zizun Li, Junyi Chen, Jiangmiao Pang, Chunhua Shen, and Tong He. π^3: Permutation-equivariant visual geometry learning. *arXiv preprint arXiv:2507.13347*, 2025b. doi: 10.48550/arXiv.2507.13347. URL https://arxiv.org/abs/2507.13347.

Zhou Wang, Alan C. Bovik, Hamid R. Sheikh, and Eero P. Simoncelli. Image quality assessment: From error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4):600–612, 2004. doi: 10.1109/TIP.2003.819861. URL https://doi.org/10.1109/TIP.2003.819861.

Zian Wang, Jonah Philion, Sanja Fidler, and Jan Kautz. Learning indoor inverse rendering with 3D spatially-varying lighting. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 12538–12547, 2021. URL https://openaccess.thecvf.com/content/ICCV2021/html/Wang_Learning_Indoor_Inverse_Rendering_With_3D_Spatially-Varying_Lighting_ICCV_2021_paper.html.

Sean Wu, Shamik Basu, Tim Broedermann, Luc Van Gool, and Christos Sakaridis. PBR-NeRF: Inverse rendering with physics-based neural fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10974–10984, 2025. doi: 10.1109/CVPR52734.2025.01025. URL https://doi.org/10.1109/CVPR52734.2025.01025.

Xiangzuo Wu, Chengwei Ren, Jun Zhou, Xiu Li, and Yuan Liu. MVInverse: Feed-forward multiview inverse rendering in seconds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 37343–37357, 2026. URL https://openaccess.thecvf.com/content/CVPR2026/html/Wu_MVInverse_Feed-forward_Multiview_Inverse_Rendering_in_Seconds_CVPR_2026_paper.html.

Haofei Xu, Songyou Peng, Fangjinhua Wang, Hermann Blum, Daniel Barath, Andreas Geiger, and Marc Pollefeys. DepthSplat: Connecting gaussian splatting and depth. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 16453–16463, 2025. doi: 10.1109/CVPR52734.2025.01534. URL https://doi.org/10.1109/CVPR52734.2025.01534.

Botao Ye, Boqi Chen, Haofei Xu, Daniel Barath, and Marc Pollefeys. YoNoSplat: You only need one model for feedforward 3D gaussian splatting. *arXiv preprint arXiv:2511.07321*, 2025a. doi: 10.48550/arXiv.2511.07321. URL https://doi.org/10.48550/arXiv.2511.07321.

Botao Ye, Sifei Liu, Haofei Xu, Xueting Li, Marc Pollefeys, Ming-Hsuan Yang, and Songyou Peng. No pose, no problem: Surprisingly simple 3D gaussian splats from sparse unposed images. In *The Thirteenth International Conference on Learning Representations*, 2025b. doi: 10.48550/arXiv.2410.24207. URL https://openreview.net/forum?id=P4o9akekdf.

Zheng Zeng, Valentin Deschaintre, Iliyan Georgiev, Yannick Hold-Geoffroy, Yiwei Hu, Fujun Luan, Ling-Qi Yan, and Miloš Hašan. RGB<->X: Image decomposition and synthesis using material- and lighting-aware diffusion models. In *ACM SIGGRAPH Conference Papers*, pp. 1–11, 2024. doi: 10.1145/3641519.3657445. URL https://doi.org/10.1145/3641519.3657445.

Kai Zhang, Fujun Luan, Qianqian Wang, Kavita Bala, and Noah 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*, pp. 5453–5462, 2021a. doi: 10.1109/CVPR46437.2021.00541. URL https://doi.org/10.1109/CVPR46437.2021.00541.

Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 586–595, 2018. doi: 10.1109/CVPR.2018.00068. URL https://doi.org/10.1109/CVPR.2018.00068.

Shangzhan Zhang, Jianyuan Wang, Yinghao Xu, Nan Xue, Christian Rupprecht, Xiaowei Zhou, Yujun Shen, and Gordon Wetzstein. FLARE: Feed-forward geometry, appearance and camera estimation from uncalibrated sparse views. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 21936–21947, 2025. doi: 10.1109/CVPR52734.2025.02043. URL https://doi.org/10.1109/CVPR52734.2025.02043.

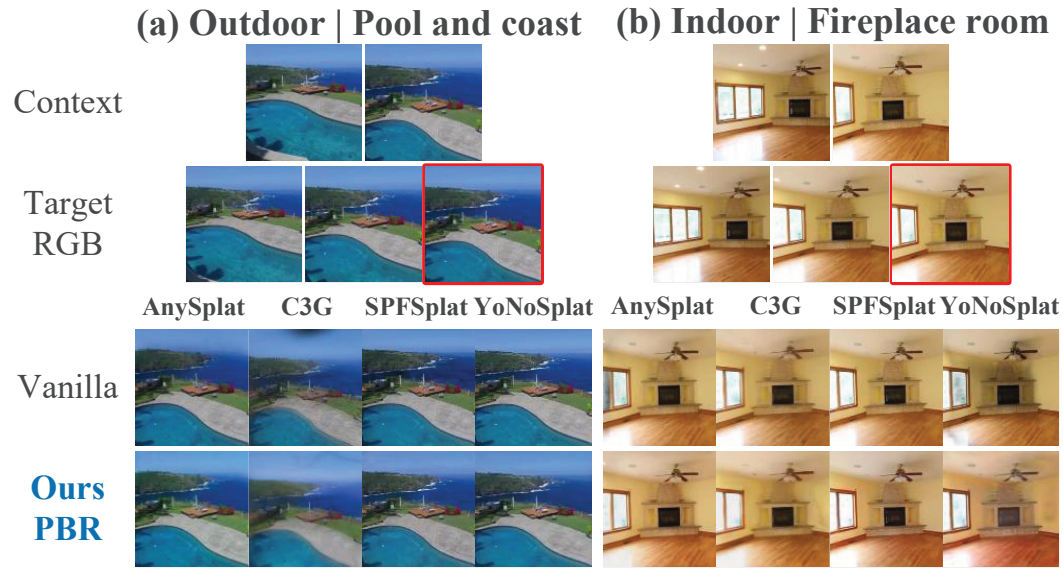
Xiuming Zhang, Pratul P. Srinivasan, Boyang Deng, Paul Debevec, William T. Freeman, and Jonathan T. Barron. NeRFactor: Neural factorization of shape and reflectance under an unknown illumination. *ACM Transactions on Graphics*, 40(6):1–18, 2021b. doi: 10.1145/3478513.3480496. URL https://doi.org/10.1145/3478513.3480496.

Tinghui Zhou, Richard Tucker, John Flynn, Graham Fyffe, and Noah Snavely. Stereo magnification: Learning view synthesis using multiplane images. *ACM Transactions on Graphics*, 37(4):1–12, 2018. doi: 10.1145/3197517.3201323. URL https://doi.org/10.1145/3197517.3201323.

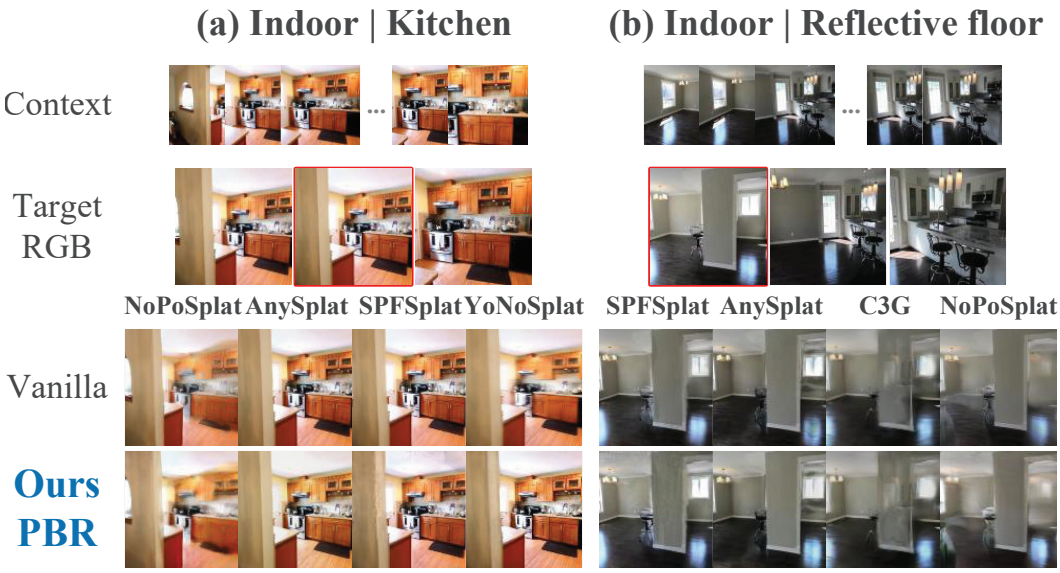
Yang Zhou, Tao Huang, Ravi Ramamoorthi, Pradeep Sen, and Ling-Qi Yan. Appearance-preserving scene aggregation for level-of-detail rendering. *ACM Transactions on Graphics*, 44(1):1–23, 2025. doi: 10.1145/3708343. URL https://doi.org/10.1145/3708343.

A Additional Cross-Backbone Qualitative Results

Figure 6 presents full-scene comparisons under the sparse-view and multi-view protocols. Figures 7 and 8 provide additional matched examples using the same display conventions.



(A) Sparse-view: 2 contexts, 3 targets.



(B) Multi-view: 10–20 contexts, 3 targets.

Figure 6: Full-scene cross-backbone qualitative comparison under sparse-view (A) and multi-view (B) protocols. Red outlines mark evaluated targets, and each method column compares matched Vanilla and Ours-PBR outputs. Ellipses in panel (B) denote unshown context views.

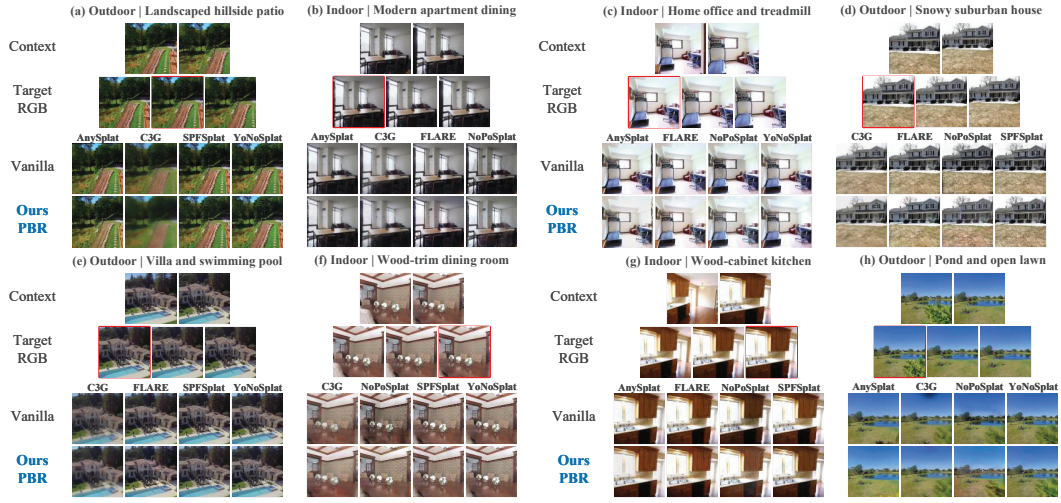


Figure 7: Additional cross-backbone qualitative results under the sparse-view protocol (2 context views and 3 held-out target views). Red outlines mark the evaluated target, and each method column compares matched Vanilla SH and Ours-PBR outputs within the same backbone.

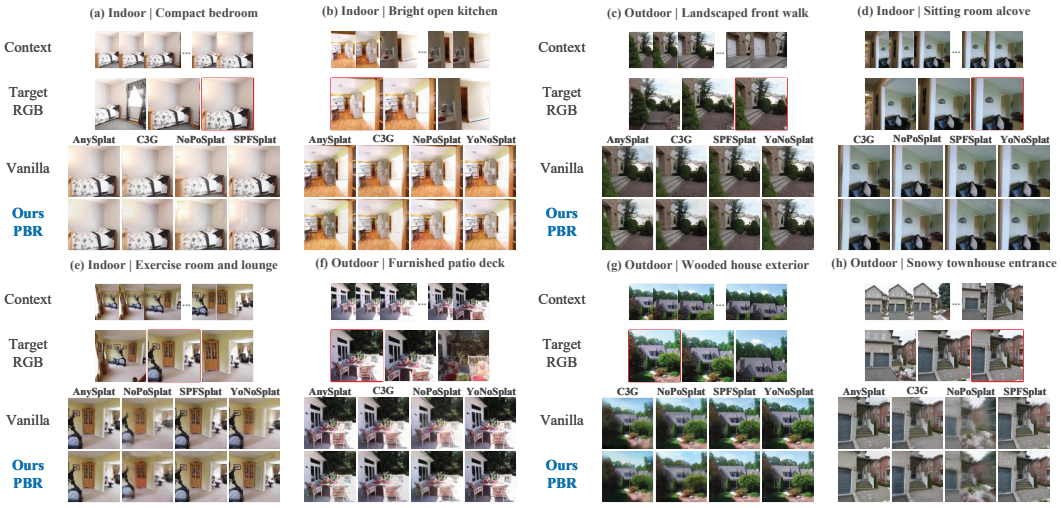
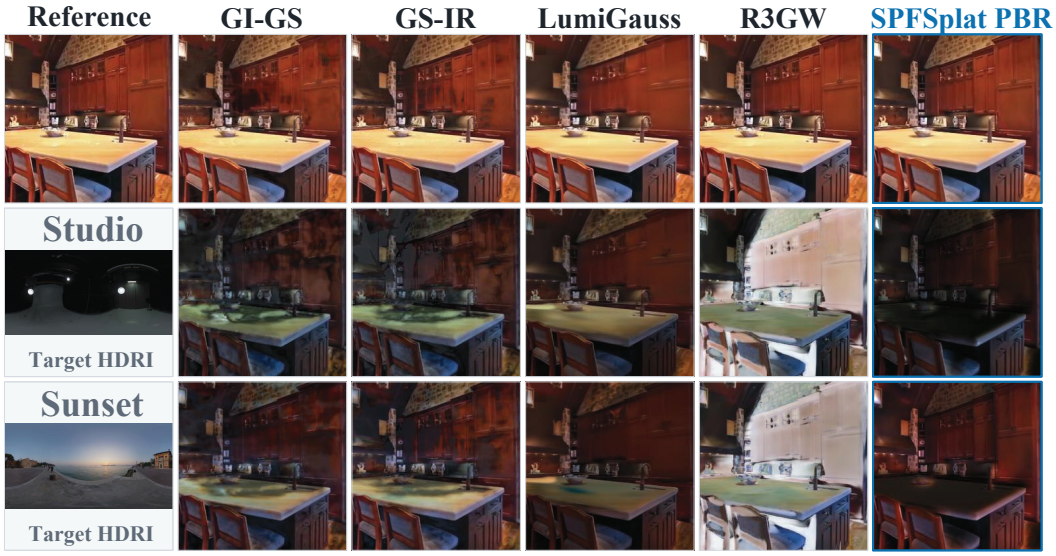


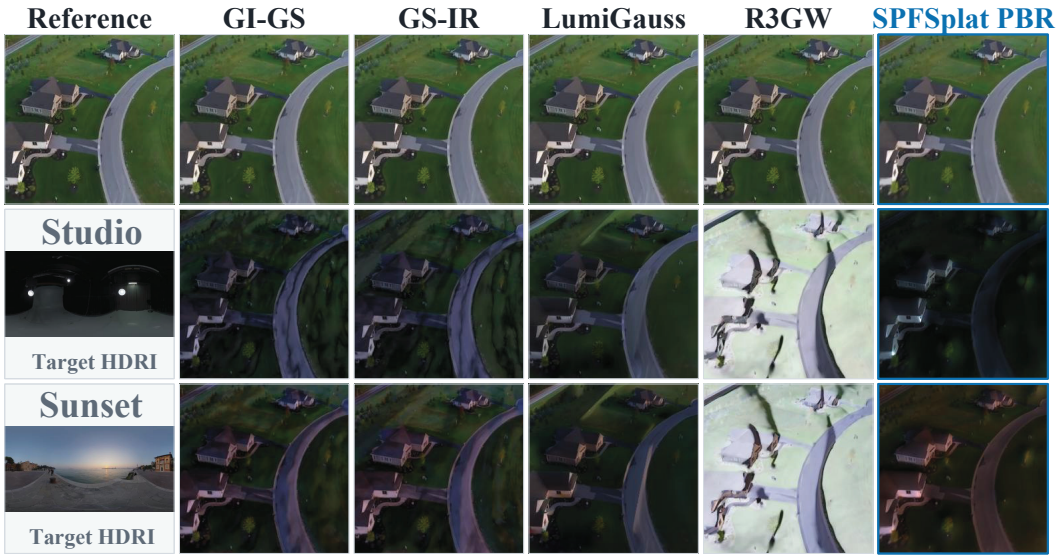
Figure 8: Additional cross-backbone qualitative results under the multi-view protocol (10–20 context views and 3 target views). Red outlines mark the evaluated target, each ellipsis denotes context views omitted from the montage, and each method column compares matched Vanilla SH and Ours-PBR outputs within the same backbone.

B Additional Relighting Results

Figure 9 provides the full indoor and outdoor comparisons accompanying Figure 4, including source-light observations and reconstructions together with Studio and Sunset relighting.



(A) Indoor kitchen.



(B) Outdoor residential scene.

Figure 9: Qualitative relighting on held-out RE10K views. Columns compare the source-light reference with GI-GS, GS-IR, LumiGauss, R3GW, and SPFSplat PBR. The top row contains the source-light observation and reconstructions, the Studio and Sunset rows show target HDRI inputs in the first column and the corresponding relit outputs. The HDRI thumbnails are lighting inputs, not target-light image ground truth.

C Ablation Metrics and Controlled Perturbations

For each scene, let d_s be the median nearest-neighbor distance between Gaussian centers in the clean scaffold G_0. We draw one scene-level noise field $\boldsymbol{\xi}_n \sim \mathcal{N}(0, \mathbf{I}_3)$, subtract its mean, normalize it to unit RMS, and reuse it at every perturbation level:

$$\boldsymbol{\mu}_n(\epsilon) = \boldsymbol{\mu}_n(0) + \epsilon d_s \boldsymbol{\xi}_n. \tag{10}$$

Each ϵ uses a separately trained PBR adapter while the corresponding G_ϵ remains frozen. The measured Δ_{proj} is the mean additional displacement, in pixels, between the projections of matched centers from G_ϵ and G_0 over the evaluated views.

For endpoint evaluation, we render each adapter's predicted PBR parameters on its corresponding geometry G_ϵ and compare them with the $\epsilon = 0$ adapter's predictions on G_0. We denote the resulting albedo, roughness, metallicity, and world-space unit normals by A_ϵ, R_ϵ, M_ϵ, and N_ϵ, respectively. Both endpoints use the same cameras, and material drift is measured on the intersection Ω of their valid raster masks. Albedo is aligned by one RGB-shared scale per scene,

$$c^* = \arg\min_c \|cA_\epsilon - A_0\|^2_{2,\Omega}, \tag{11}$$

after which the reported drifts are

$$\begin{aligned} D_{\text{alb}} &= \text{mean}_\Omega\, |c^* A_\epsilon - A_0|, \\ D_{\text{rough}} &= \text{RMSE}_\Omega(R_\epsilon, R_0), \qquad D_{\text{metal}} = \text{RMSE}_\Omega(M_\epsilon, M_0), \\ D_{\text{normal}} &= \frac{180}{\pi}\, \text{mean}_\Omega \arccos\big(\text{clamp}(N_\epsilon^\top N_0, -1, 1)\big)\,. \end{aligned} \tag{12}$$

For endpoint relighting, we shade each endpoint under a fixed bank $\mathcal{H}$ of five HDR environments and average full-frame prediction-to-prediction LPIPS:

$$D_{\text{relight}} = \frac{1}{|\mathcal{H}|} \sum_{h \in \mathcal{H}} \text{LPIPS}\big(I^{\text{rel}}_{\epsilon,h}, I^{\text{rel}}_{0,h}\big)\,. \tag{13}$$

Table 3: Exact measurements for the controlled center-perturbation experiment summarized in Figure 5. Endpoint drift compares the adapter trained on G_ϵ and evaluated on G_ϵ with the matched adapter trained and evaluated on G_0.

Path	ϵ	Added Δ_{proj} (px)	Source-light RGB		Endpoint drift from the matched $\epsilon = 0$ run				
			PSNR↑	LPIPS-VGG↓	ΔAlb. L1↓	ΔRough. RMSE↓	ΔMetal. RMSE↓	ΔNormal deg.↓	ΔRelight LPIPS↓
Direct-PBR	0.00	0.000	24.5067	0.1509	ref.	ref.	ref.	ref.	ref.
	0.05	0.234	24.5114	0.1511	0.0141	0.0318	0.0493	6.050	0.0827
	0.10	0.468	24.5105	0.1504	0.0135	0.0391	0.0562	5.866	0.0878
	0.20	0.936	24.3052	0.1556	0.0164	0.0448	0.0563	5.814	0.1062

D Local RGB Residual Decomposition

Let $\mathcal{R}(g, p, \mathcal{E})$ denote the stacked RGB output of the differentiable renderer as a function of the Gaussian scaffold g, pixel-aligned PBR parameters p, and environment illumination $\mathcal{E}$. Around a reference state $(g^\star, p^\star, \mathcal{E}^\star)$, define $e_g = g - g^\star$, $\Delta p = p - p^\star$, and $\Delta\mathcal{E} = \mathcal{E} - \mathcal{E}^\star$. A first-order expansion gives

$$\begin{aligned} \mathbf{r} &= \mathcal{R}(g^\star + e_g, p^\star + \Delta p, \mathcal{E}^\star + \Delta\mathcal{E}) - \mathcal{R}(g^\star, p^\star, \mathcal{E}^\star) \\ &= A_g e_g + A_p \Delta p + A_\mathcal{E} \Delta\mathcal{E} + R_2, \end{aligned} \tag{14}$$

where A_g, A_p, and $A_\mathcal{E}$ are the corresponding renderer Jacobians at the reference state and R_2 contains higher-order terms. This is Eq. (4).

The map perturbation contains a geometry-mediated component that can be factored more explicitly. Let $\mathbf{w}(g)$ stack the relevant pixelwise Gaussian compositing weights and let $\mathbf{M}_k(g)$ be the PBR

map predicted by route $k \in \{d, f\}$ while its learned contributor attributes, decoder parameters, and scene condition are held fixed. To isolate the material-mediated rendering path, also hold illumination, viewing direction, and all other nonmaterial shader inputs fixed. Define $\delta\mathbf{M}_k = \mathbf{M}_k(g^\star + e_g) - \mathbf{M}_k(g^\star)$ and define $\delta\hat{\mathbf{I}}_k$ analogously for the corresponding rendered output. Within a neighborhood with unchanged active contributors and compositing order, the chain rule, with Jacobians evaluated at the reference state, gives

$$\begin{aligned} \delta\mathbf{M}_k &= \mathbf{J}_{\mathbf{M}_k,\mathbf{w}}\mathbf{J}_{\mathbf{w},g}e_g + o(\|e_g\|), \\ \delta\hat{\mathbf{I}}_k &= \mathbf{J}^{(k)}_{S_{\mathrm{pbr}},\mathbf{M}}\delta\mathbf{M}_k + o(\|\delta\mathbf{M}_k\|). \end{aligned} \tag{15}$$

Here, $\mathbf{J}^{(k)}_{S_{\mathrm{pbr}},\mathbf{M}}$ is the PBR-shader Jacobian evaluated at route k's reference map. Identifying $\delta\mathbf{M}_k$ with its stacked map vector, Eq. (15) gives the geometry-mediated component of Δp and its contribution $A_p\delta\mathbf{M}_k$ to Eq. (14). This term is distinct from the partial geometry response $A_g e_g$, which holds the pixel-aligned PBR map fixed, and from additional map changes caused by training the PBR parameter predictor. If the illumination head also depends on rasterized features, $\Delta\mathcal{E}$ admits an analogous geometry-mediated component, we isolate the material path here. Changes in the active contributors or their order form boundaries between such differentiable neighborhoods.

When the scaffold is frozen, $b_g = A_g e_g$ is fixed during adapter training. Neglecting R_2, write $C = [A_p, A_{\mathcal{E}}]$ and $\boldsymbol{\xi} = (\Delta p^\top, \Delta\mathcal{E}^\top)^\top$. The local first-order photometric response solves

$$\min_{\boldsymbol{\xi}} \|b_g + C\boldsymbol{\xi}\|_2^2, \qquad C\boldsymbol{\xi}^\star = -P_{\mathrm{col}(C)}b_g, \tag{16}$$

where $P_{\mathrm{col}(C)}$ is the orthogonal projector onto the column space of C. Thus, the component of the geometry-induced residual in the joint material–illumination Jacobian range can be reduced by updates to the predicted PBR parameters and illumination. The fitted RGB correction is fixed by this joint range, but its split between material and illumination can be non-unique when their Jacobian ranges overlap, the adapter parameterization, regularization, and optimizer select a particular solution.

E Full Analysis of the Post-Splat PBR Bottleneck

E.1 Raw Contributor Moments and Route Equivalence

Fix a pixel and the front-to-back compositing order. Let $x_n \in \mathcal{X}$ denote the state of its n-th Gaussian contributor, and let δ_{x_n} be the Dirac measure at that state. The contributors define the finite positive measure

$$\nu = \sum_{n=1}^{N} \bar{w}_n \delta_{x_n}, \qquad s = \nu(\mathcal{X}) = \sum_n \bar{w}_n > 0, \qquad \widehat{\nu} = \frac{\nu}{s}. \tag{17}$$

For an embedding $\phi : \mathcal{X} \to \mathbb{R}^d$, raw feature splatting is the neural moment

$$T_\phi(\nu) = \int_{\mathcal{X}} \phi(x)\, d\nu(x) = \sum_n \bar{w}_n \phi(x_n) = s\, \mathbb{E}_{\widehat{\nu}}[\phi(x)]. \tag{18}$$

This is the weighted-measure generalization of a sum-pooled set representation (Amir et al., 2023; Wagstaff et al., 2022). The raw moment scales linearly with the total contributor weight, whereas its normalized counterpart is invariant to this scaling:

$$T_\phi(c\nu) = cT_\phi(\nu), \qquad \frac{T_\phi(c\nu)}{c\,s} = \frac{T_\phi(\nu)}{s}. \tag{19}$$

Moreover, the learned embedding family can allocate one coordinate to $\phi_0(x) = 1$. With $\phi = [1, \widetilde{\phi}]$, the raw feature explicitly contains $[s, s\mathbb{E}_{\widehat{\nu}}[\widetilde{\phi}]]$, jointly retaining coverage and contributor composition.

Let $g_D : \mathcal{X} \to \mathbb{R}^8$ predict a per-Gaussian PBR tuple including its per-Gaussian range constraints, let $B_{\mathrm{pre}} : \mathcal{X} \to \mathbb{R}^{64}$ produce a pre-splat latent, and let $D_{\mathrm{pre}} : \mathbb{R}^{64} \to \mathbb{R}^8$ be its pointwise PBR head with the corresponding constraints. Define $g_{\mathrm{dir}} = g_D$ and $g_{\mathrm{pre}} = D_{\mathrm{pre}} \circ B_{\mathrm{pre}}$. Let Ψ_d and

Ψ_{pre} collect any remaining post-raster normalization, such as unit-normal normalization. The two early-bottleneck paths are

$$\begin{aligned}\mathbf{M}^{\mathrm{dir}}(\nu) &= \Psi_d(T_{g_D}(\nu))\,,\\ \mathbf{M}^{\mathrm{pre}}(\nu) &= \Psi_{\mathrm{pre}}\big(T_{g_{\mathrm{pre}}}(\nu)\big)\,.\end{aligned} \tag{20}$$

Thus, the pre-splat bottleneck remains a Direct predictor with a reparameterized per-Gaussian head. Its 64-D code is decoded to an 8-D tuple before the moment is formed, so the statistic exposed after rasterization is still an explicit 8-D PBR moment.

E.2 Moment Fibers and Irreducible PBR-Map Error

For route $\kappa \in \{\mathrm{dir}, \mathrm{pre}\}$, let g_κ denote its per-contributor mapping and define the rasterized statistic $T_8^\kappa = T_{g_\kappa}$. Its fiber at $t \in \mathbb{R}^8$ is

$$\mathcal{F}_8^\kappa(t) = \{\nu : T_8^\kappa(\nu) = t\}. \tag{21}$$

Let $\mathbf{M}^\star(\nu) \in \mathbb{R}^8$ denote the desired pixel-aligned PBR tuple for contributor measure ν. Every prediction that factors through T_8^κ is constant on each fiber. Suppose two contributor measures lie in the same fiber but require different targets, i.e., $T_8^\kappa(\nu_1) = T_8^\kappa(\nu_2)$ and $\mathbf{M}^\star(\nu_1) \neq \mathbf{M}^\star(\nu_2)$. Under an equiprobable two-state distribution, the optimal common prediction is their midpoint, yielding

$$\inf_{\mathbf{m}\in\mathbb{R}^8} \frac{1}{2}\sum_{i=1}^{2} \|\mathbf{m} - \mathbf{M}^\star(\nu_i)\|_2^2 = \frac{1}{4}\|\mathbf{M}^\star(\nu_1) - \mathbf{M}^\star(\nu_2)\|_2^2 > 0. \tag{22}$$

This lower bound is caused by a non-injective statistic: once the two mixtures share T_8^κ, neither a fixed post-raster map nor the common PBR renderer can produce different outputs for the two cases.

E.3 Retained Statistics and Rendering Error

For either early route $\kappa \in \{\mathrm{dir}, \mathrm{pre}\}$, write $T_8^\kappa = T_{g_\kappa}$. Two mixtures that produce the same T_8^κ are indistinguishable to any decoder that receives this statistic. To compare the aggregation forms, fix g_κ and consider the 64-D feature map

$$\begin{aligned}&\phi_\kappa^+(x) = [1, g_\kappa(x), \chi_\kappa(x)],\\ &T_+^\kappa(\nu) = [s, T_8^\kappa(\nu), T_{\chi_\kappa}(\nu)], \qquad T_8^\kappa = \Pi_\kappa T_+^\kappa,\end{aligned} \tag{23}$$

where $\chi_\kappa : \mathcal{X} \to \mathbb{R}^{55}$ supplies additional moments and Π_κ selects the eight explicit PBR channels. This construction retains the chosen early statistic while allowing other mixture information to reach the decoder. We suppress κ below.

Let Y be the reference RGB image and let $\boldsymbol{\omega}$ collect illumination, viewing directions, and other nonmaterial shader inputs. Let z_s denote the global scene latent, whose construction is given in Appendix G.1. For a statistic T, define

$$\mathcal{R}^*_{\mathrm{PBR}}(T) = \inf_D \mathbb{E}\left[\|S_{\mathrm{pbr}}(D(T; z_s); \boldsymbol{\omega}) - Y\|_2^2\right], \tag{24}$$

where D ranges over measurable mappings to valid PBR parameter maps. Here T collects the rasterized statistics of all target pixels, and Π applies the channel selection at each pixel. This formulation includes decoders with spatial normalization. The comparison uses the same joint distribution of scenes and target views, the same scene condition, and the same shader inputs.

For any decoder D_8 receiving T_8, define

$$D_+(T_+; z_s) = D_8(\Pi T_+; z_s) = D_8(T_8; z_s). \tag{25}$$

The first step extracts the original eight channels, the second applies the original decoder. Both routes therefore produce identical PBR maps and rendered images. Taking the infimum over decoders gives

$$\mathcal{R}^*_{\mathrm{PBR}}(T_+) \le \mathcal{R}^*_{\mathrm{PBR}}(T_8), \tag{26}$$

which proves Eq. (9). The extra channels can also support predictions that depend on moments absent from T_8.

The retained information can be quantified using the data-processing inequality (Cover & Thomas, 2006). With common conditions $C = (z_s, \boldsymbol{\omega})$ and finite mutual information, the chain rule gives

$$\begin{aligned} I(Y; T_+ \mid C) &= I(Y; T_8, T_+ \mid C) \\ &= I(Y; T_8 \mid C) + I(Y; T_+ \mid T_8, C). \end{aligned} \tag{27}$$

The last term measures information about the rendering target that remains in T_+ after T_8 is known.

For squared-error RGB prediction, this information has a corresponding risk decomposition. Assume $\mathbb{E}\|Y\|_2^2 < \infty$, let $m_T = \mathbb{E}[Y \mid T, C]$, and define $\mathcal{R}^*_{\mathrm{RGB}}(T) = \inf_\Gamma \mathbb{E}\|Y - \Gamma(T, C)\|_2^2$ over measurable RGB predictors. Conditional expectation minimizes this loss, and

$$\mathcal{R}^*_{\mathrm{RGB}}(T_8) - \mathcal{R}^*_{\mathrm{RGB}}(T_+) = \mathbb{E}\|m_{T_+} - m_{T_8}\|_2^2 \geq 0. \tag{28}$$

To obtain the identity, expand $Y - m_{T_8} = (Y - m_{T_+}) + (m_{T_+} - m_{T_8})$; the two terms have zero expected inner product. The reduction is strict when the conditional means differ with positive probability. Equation (26) establishes the ordering with the PBR output constraints, while this identity isolates the contribution of retained information.

A collision of statistics also gives a direct rendering-error lower bound. Consider two equally likely mixtures with identical T_8 and common conditions, but different reference responses $Y_1 \neq Y_2$. For a pixelwise example, hold all other pixels' inputs and target responses fixed. Any decoder receiving T_8 produces the same rendered response $\widehat{Y}$ in both cases, so

$$\begin{aligned} \frac{1}{2}\sum_{i=1}^{2} \|\widehat{Y} - Y_i\|_2^2 &= \left\|\widehat{Y} - \frac{Y_1 + Y_2}{2}\right\|_2^2 + \frac{1}{4}\|Y_1 - Y_2\|_2^2 \\ &\geq \frac{1}{4}\|Y_1 - Y_2\|_2^2 > 0. \end{aligned} \tag{29}$$

The GGX example below shows how an additional moment removes this ambiguity and yields a valid PBR parameter with the required rendering response.

Using the extended statistic T_+^κ in Eq. (23), the PBR-map risk has a corresponding decomposition. For squared PBR-map loss, define $R^\star(T) = \inf_\Gamma \mathbb{E}\|\mathbf{M}^\star - \Gamma(T)\|_2^2$, where the infimum is over measurable decoders Γ. The conditional-variance identity gives

$$R^\star(T_8^\kappa) - R^\star(T_+^\kappa) = \mathbb{E}\left\|\mathbb{E}[\mathbf{M}^\star \mid T_+^\kappa] - \mathbb{E}[\mathbf{M}^\star \mid T_8^\kappa]\right\|_2^2 \geq 0. \tag{30}$$

The inequality is strict exactly when an extra retained moment changes the conditional mean of the target PBR map.

E.4 Post-Aggregation Interactions

For a per-contributor mapping $\gamma : \mathcal{X} \to \mathbb{R}^m$, the early statistic $A_\gamma(\nu) = \int \gamma(x)\, d\nu(x)$ is additive in the contributor measure. For fixed scene condition z_s, write $\rho(\mathbf{z}) = D_{\mathrm{mat}}(\mathbf{z}; z_s)$. A post-splat predictor then has the sum-decomposable form $F(\nu) = \rho(T_\phi(\nu))$. Define its centered interaction between two contributor groups as

$$\mathcal{I}_F(\nu_1, \nu_2) = F(\nu_1 + \nu_2) - F(\nu_1) - F(\nu_2) + F(0). \tag{31}$$

Additivity gives $\mathcal{I}_{A_\gamma} \equiv 0$. Let $\mathbf{z}_i = T_\phi(\nu_i)$. For a twice differentiable output channel ρ_k with Hessian H_{ρ_k}, the post-splat interaction is exactly

$$\mathcal{I}_{F_k}(\nu_1, \nu_2) = \int_0^1 \int_0^1 \mathbf{z}_1^\top H_{\rho_k}(\tau \mathbf{z}_1 + \sigma \mathbf{z}_2)\mathbf{z}_2 \, d\tau \, d\sigma. \tag{32}$$

Equivalently, for two distinct Gaussian features $\mathbf{f}_i, \mathbf{f}_j$ and perturbation directions $\mathbf{d}, \mathbf{e}$,

$$\mathrm{D}^2_{\mathbf{f}_i, \mathbf{f}_j} F_k[\mathbf{d}, \mathbf{e}] = \bar{w}_i \bar{w}_j \, \mathbf{d}^\top H_{\rho_k}(T_\phi(\nu))\mathbf{e}, \tag{33}$$

whereas the mixed derivative of the raw additive statistic is zero. A non-affine post-aggregation map can combine retained moments through these interactions. A nonlinear map applied after an early

statistic can also introduce interactions, but it assigns the same output to mixtures that share that statistic. For an affine map $\rho(\mathbf{z}) = A\mathbf{z} + \mathbf{b}$, the raw aggregation instead satisfies

$$\rho\left(\sum_n \bar{w}_n \mathbf{f}_n\right) - \sum_n \bar{w}_n \rho(\mathbf{f}_n) = (1-s)\mathbf{b}. \tag{34}$$

Thus, affine decoding commutes with aggregation at unit coverage or with zero bias.

Nonlinear decoding determines how retained moments interact, while the feature map determines which moments remain available. These roles are complementary: a later nonlinearity can use the information in its input, but cannot distinguish mixtures already mapped to the same statistic.

The raw-statistic distinction already appears in one dimension. At unit coverage, take $f_1 = 0$, $f_2 = 1$, weights $(t, 1-t)$, and $\rho(z) = z^2$. Post-splat decoding returns $(1-t)^2$, whereas the raw pre-splat statistic for fixed per-contributor outputs q_1 and q_2 is the affine mixture $tq_1 + (1-t)q_2$. A subsequent nonlinear map can transform this scalar statistic but cannot recover mixture information that it does not contain. Latent width is therefore not itself the source of the distinction, it determines how many mixture statistics survive until the nonlinearity is applied.

E.5 Why Explicit PBR Means Are Insufficient

PBR shading depends nonlinearly on normal and roughness, so averaging these parameters generally differs from averaging their shaded responses (Pharr et al., 2024). With the remaining shader inputs denoted by $\boldsymbol{\omega}$, an effective PBR tuple can be defined by

$$\mathbf{p}_{\text{eff}}(\nu) \in \arg\min_{\mathbf{p}\in\mathcal{P}} \mathbb{E}_{\boldsymbol{\omega}} \left\| S_{\text{pbr}}(\mathbf{p};\boldsymbol{\omega}) - \int S_{\text{pbr}}(g_D(x);\boldsymbol{\omega})\, d\widehat{\nu}(x) \right\|_2^2, \tag{35}$$

where $\mathcal{P}$ is the valid PBR parameter domain. The dependence between material channels under filtering is also central to learned SVBRDF filtering (Gauthier et al., 2022) and appearance-preserving scene aggregation (Zhou et al., 2025).

For example, fix all material channels except roughness r, and set $\bar{r} = \mathbb{E}_{\widehat{\nu}}[r]$. If S_{pbr} is three times differentiable over the roughness interval, with $\|\partial_r^3 S_{\text{pbr}}\|_2 \le K$, Taylor expansion gives

$$\begin{aligned} \mathbb{E}_{\widehat{\nu}}[S_{\text{pbr}}(r;\boldsymbol{\omega})] - S_{\text{pbr}}(\bar{r};\boldsymbol{\omega}) &= \frac{1}{2}\partial_r^2 S_{\text{pbr}}(\bar{r};\boldsymbol{\omega})\,\mathrm{Var}_{\widehat{\nu}}(r) + \varepsilon_3, \\ \|\varepsilon_3\|_2 &\le \frac{K}{6}\mathbb{E}_{\widehat{\nu}}[|r-\bar{r}|^3]. \end{aligned} \tag{36}$$

The first-order term vanishes because $\mathbb{E}_{\widehat{\nu}}[r - \bar{r}] = 0$. Thus, the variance of roughness contributes to the pixel response through the curvature of the shader. This makes the missing distribution information relevant to rendering.

Consider a GGX example at unit coverage, with a fixed metallic material, no diffuse component, and illumination and viewing directions aligned with the surface normal. The GGX normal distribution and Smith masking term satisfy

$$D_{\text{GGX}}(\alpha, 1) = \frac{1}{\pi\alpha^2}, \qquad G = 1, \tag{37}$$

where $\alpha > 0$ is the microfacet width (Walter et al., 2007). Parameterize $\alpha = r^\gamma$ with $\gamma > 0$ and absorb the fixed illumination and Fresnel factors into a nonzero RGB vector $\mathbf{k}$. The linear-RGB response is then

$$S_{\text{pbr}}(r) = \mathbf{k}\, h(r), \qquad h(r) = r^{-2\gamma}. \tag{38}$$

Both $\gamma = 1$ and $\gamma = 2$ satisfy this construction.

Choose $d > 0$ and three contributors with roughness values $(a-d, a, a+d)$ inside the valid roughness range and identical remaining PBR parameters. Give them weights $(t, 1-2t, t)$ for $0 < t < 1/2$. The contributor set and coverage stay fixed, and their mean roughness is

$$\mathbb{E}_t[r] = t(a-d) + (1-2t)a + t(a+d) = a. \tag{39}$$

The complete explicit PBR statistic is therefore constant in t. In contrast, the shaded mixture has response $Y(t) = \mathbf{k}\,\mu_h(t)$, where

$$\mu_h(t) = h(a) + t\,[h(a-d) + h(a+d) - 2h(a)]\,. \tag{40}$$

Since $h''(r) = 2\gamma(2\gamma+1)r^{-2\gamma-2} > 0$, the bracket is positive. Any two distinct values of t have the same T_8 but different target responses, giving the positive lower bound in Eq. (29).

If an additional feature coordinate stores $\chi_1(r) = h(r)$, its aggregate retains $\mu_h(t)$. A post-splat decoder can then output

$$\begin{aligned} r_{\text{eff}}(t) &= \mu_h(t)^{-1/(2\gamma)} \in [a-d, a+d], \\ S_{\text{pbr}}(r_{\text{eff}}(t)) &= \mathbf{k}\,\mu_h(t) = Y(t). \end{aligned} \tag{41}$$

Thus, for this pair of mixtures under the fixed illumination, the extended statistic admits zero rendering error while the explicit PBR mean has a positive minimum error. The example connects a specific retained statistic to a valid material prediction and an improved rendering response.

As a microfacet-width example, the distributions $\zeta_1 = \delta_{0.5}$ and $\zeta_2 = \frac{1}{2}\delta_{0.2} + \frac{1}{2}\delta_{0.8}$ have the same mean α, but their mean lobe peaks are $4/\pi$ and $13.28125/\pi$, respectively. An explicit first moment maps both cases to the same mean width, whereas a learned moment can retain dispersion and normal–roughness correlation. Glossy inverse rendering likewise observes that shading and alpha blending do not commute (Lai et al., 2025).

At a cross-surface overlap, let the visible material be $\mathbf{p}_A$ while an erroneously overlapping contributor carries $\mathbf{p}_B$ with weight $\epsilon \in [0,1]$. Direct-PBR produces

$$\bar{\mathbf{p}}_\epsilon = (1-\epsilon)\mathbf{p}_A + \epsilon\mathbf{p}_B, \qquad \|\bar{\mathbf{p}}_\epsilon - \mathbf{p}_A\|_2 = \epsilon\|\mathbf{p}_B - \mathbf{p}_A\|_2, \tag{42}$$

thereby coupling visibility weights to material interpolation. Features that retain contributor identity can support a nonlinear material-selection rule after aggregation. As in the roughness example, the useful additional information is the information needed to distinguish mixtures with different target materials or rendering responses.

This analysis characterizes the information available after a contributor mixture has been formed. It establishes a refinement of the Direct-PBR statistic and a larger class of post-aggregation functions, rather than an unconditional ordering of the errors learned by two finite networks. The perturbation-specific comparison requires the local sensitivity analysis below.

F Geometry-to-PBR Propagation Through Gaussian Compositing

F.1 Raw Splatting Decomposition

We now derive the compositing-mediated geometry-to-PBR response for raw alpha-premultiplied feature splatting and its Direct-PBR counterpart. Fix a valid target pixel and suppress its view and pixel indices. Let

$$\mathbf{w} = [\bar{w}_1, \ldots, \bar{w}_N]^\top, \qquad s = \mathbf{1}^\top\mathbf{w} > 0, \qquad \boldsymbol{\pi} = \frac{\mathbf{w}}{s}, \tag{43}$$

where $\bar{w}_n$ is the accumulated front-to-back compositing weight from Eq. (1). Hence $\mathbf{1}^\top\boldsymbol{\pi} = 1$. For the local feature matrix $\mathbf{F} = [\mathbf{f}_1, \ldots, \mathbf{f}_N]$, with $\mathbf{f}_n = \phi(x_n)$ in the notation of Appendix E, define

$$\mathbf{h} = \mathbf{F}\boldsymbol{\pi}, \qquad \widetilde{\mathbf{f}} = \mathbf{F}\mathbf{w} = s\mathbf{h}. \tag{44}$$

Thus, $\widetilde{\mathbf{f}} = T_\phi(\nu)$ links the matrix and measure formulations.

Within a differentiable neighborhood that preserves the active contributors and compositing order, hold $\mathbf{F}$, z_s, and the trained material decoder fixed. Differentiating Eq. (43) gives

$$\mathbf{J}_{s,g} = \mathbf{1}^\top\mathbf{J}_{\mathbf{w},g}, \qquad \mathbf{J}_{\boldsymbol{\pi},g} = \frac{1}{s}\left(\mathbf{I} - \boldsymbol{\pi}\mathbf{1}^\top\right)\mathbf{J}_{\mathbf{w},g}, \qquad \mathbf{1}^\top\mathbf{J}_{\boldsymbol{\pi},g} = 0. \tag{45}$$

where $\mathbf{I}$ is the identity matrix of the appropriate size. The product rule then yields

$$\begin{aligned} \mathbf{J}_{\widetilde{\mathbf{f}},g} &= \mathbf{h}\,\mathbf{J}_{s,g} + s\mathbf{F}\mathbf{J}_{\boldsymbol{\pi},g} \\ &= \mathbf{h}\,\mathbf{J}_{s,g} + s\big(\mathbf{F} - \mathbf{h}\mathbf{1}^\top\big)\mathbf{J}_{\boldsymbol{\pi},g}. \end{aligned} \tag{46}$$

The second equality follows because subtracting any shared vector from all columns of $\mathbf{F}$ leaves the redistribution response unchanged. Define $\mathbf{M}_f = D_{\mathrm{mat}}(\widetilde{\mathbf{f}}; z_s)$. Composing its decoder Jacobian with Eq. (46) gives

$$\mathbf{J}_{\mathbf{M}_f,g} = \mathbf{J}_{D_{\mathrm{mat}},\widetilde{\mathbf{f}}} \left[\mathbf{h}\, \mathbf{J}_{s,g} + s\big(\mathbf{F} - \mathbf{h}\mathbf{1}^\top\big)\mathbf{J}_{\boldsymbol{\pi},g} \right]. \tag{47}$$

F.2 Finite Redistribution Response

Consider two normalized weight vectors $\boldsymbol{\pi}$ and $\boldsymbol{\pi}'$ over the same local features and with the same coverage s. Define

$$d_{\mathrm{TV}}(\boldsymbol{\pi}', \boldsymbol{\pi}) = \frac{1}{2}\|\boldsymbol{\pi}' - \boldsymbol{\pi}\|_1, \qquad \mathrm{diam}(\mathbf{F}) = \max_{i,j} \|\mathbf{f}_i - \mathbf{f}_j\|_2. \tag{48}$$

The positive and negative parts of $\boldsymbol{\pi}' - \boldsymbol{\pi}$ have equal mass d_{TV}. The result is immediate when this mass is zero, otherwise, dividing their feature-weighted sums by d_{TV} yields two points in the convex hull of $\{\mathbf{f}_n\}$, which gives

$$\|\widetilde{\mathbf{f}}' - \widetilde{\mathbf{f}}\|_2 \le s\, d_{\mathrm{TV}}(\boldsymbol{\pi}', \boldsymbol{\pi})\ \mathrm{diam}(\mathbf{F}). \tag{49}$$

If the material decoder is locally L_D-Lipschitz over this segment with z_s fixed, then

$$\|\mathbf{M}'_f - \mathbf{M}_f\|_2 \le L_D s\, d_{\mathrm{TV}}(\boldsymbol{\pi}', \boldsymbol{\pi})\ \mathrm{diam}(\mathbf{F}). \tag{50}$$

Coverage variation is represented separately by the first term of Eq. (46).

F.3 Direct-PBR Counterpart

Let $\mathbf{P} = [\mathbf{p}_1, \ldots, \mathbf{p}_N]$ collect the explicit, per-Gaussian PBR tuples blended by the Direct-PBR path, including its pre-raster output constraints. In the measure notation, $\mathbf{p}_n = g_D(x_n)$ and hence $\mathbf{P}\mathbf{w} = T_{g_D}(\nu)$. Its raw rasterized buffer is

$$\widetilde{\mathbf{p}} = \mathbf{P}\mathbf{w} = s\bar{\mathbf{p}}, \qquad \bar{\mathbf{p}} = \mathbf{P}\boldsymbol{\pi}. \tag{51}$$

Holding $\mathbf{P}$ fixed gives the corresponding compositing Jacobian

$$\mathbf{J}_{\widetilde{\mathbf{p}},g} = \bar{\mathbf{p}}\, \mathbf{J}_{s,g} + s\big(\mathbf{P} - \bar{\mathbf{p}}\mathbf{1}^\top\big)\mathbf{J}_{\boldsymbol{\pi},g}. \tag{52}$$

Any deterministic post-raster normalization is represented by $\mathbf{M}_d = \Psi_d(\widetilde{\mathbf{p}})$, with $\mathbf{J}_{\mathbf{M}_d,g} = \mathbf{J}_{\Psi_d,\widetilde{\mathbf{p}}}\mathbf{J}_{\widetilde{\mathbf{p}},g}$. At fixed coverage,

$$\|\widetilde{\mathbf{p}}' - \widetilde{\mathbf{p}}\|_2 \le s\, d_{\mathrm{TV}}(\boldsymbol{\pi}', \boldsymbol{\pi})\ \mathrm{diam}(\mathbf{P}). \tag{53}$$

If Ψ_d is locally L_{Ψ_d}-Lipschitz over the corresponding segment, then

$$\|\mathbf{M}'_d - \mathbf{M}_d\|_2 \le L_{\Psi_d} s\, d_{\mathrm{TV}}(\boldsymbol{\pi}', \boldsymbol{\pi})\ \mathrm{diam}(\mathbf{P}). \tag{54}$$

Thus, the two paths share the geometry-to-weight response when evaluated on the same scaffold. At fixed coverage, their material responses are controlled by $L_D\,\mathrm{diam}(\mathbf{F})$ and $L_{\Psi_d}\,\mathrm{diam}(\mathbf{P})$, respectively. These are route-specific upper bounds, comparing them alone does not order the realized perturbations.

F.4 Directional Comparison and Shading Response

For a unit perturbation direction $\mathbf{u}_g$, define

$$\mathbf{v}_k(\mathbf{u}_g) = \mathbf{J}_{\mathbf{M}_k,g}\mathbf{u}_g, \qquad k \in \{d, f\}. \tag{55}$$

The first-order expansion along $g_t = g + t\mathbf{u}_g$ is

$$\mathbf{M}_k(g_t) - \mathbf{M}_k(g) = t\mathbf{v}_k(\mathbf{u}_g) + o(|t|). \tag{56}$$

Consequently, if $\|\mathbf{v}_f(\mathbf{u}_g)\|_2 < \|\mathbf{v}_d(\mathbf{u}_g)\|_2$, then there exists $t_0 > 0$ such that the feature route has smaller PBR-map drift for every $0 < |t| < t_0$ along this direction. Equations (47) and (52) show that this strict condition compares the realized latent-decoder response with the realized explicit-PBR response, not merely their worst-case upper bounds.

With illumination, viewing direction, and all other nonmaterial shader inputs fixed, the corresponding material-mediated rendered-image response is

$$\hat{\mathbf{I}}_k(g_t) - \hat{\mathbf{I}}_k(g) = t\, \mathbf{J}^{(k)}_{S_{\mathrm{pbr}},\mathbf{M}}\mathbf{J}_{\mathbf{M}_k,g}\mathbf{u}_g + o(|t|). \tag{57}$$

The shader therefore maps geometry-induced PBR drift into RGB space.

G ARCHITECTURE DETAILS

This appendix specifies scene latent generation and the multilevel-feature instantiation of the post-training adapter in Section 4.2. Backbone-specific feature interfaces follow Appendix H. Each backbone provides initial Gaussian positions and attributes together with intermediate features. Its original prediction pathways remain frozen, while the adapter learns scene conditioning, feature and attribute refinement, material decoding, and illumination estimation.

G.1 SCENE LATENT GENERATION

Consider a scene with V context views. Let $D_v^{(\ell)}$ denote the tokens of view v at backbone layer ℓ, and let $D_v \in \mathbb{R}^{N\times C}$ denote its final-layer tokens. Here, N is the number of tokens per view and C is their feature dimension. We suppress the scene index below. LayerNorm and GroupNorm are denoted by LN and GN, respectively, and affine maps act along the feature dimension.

The scene generator pools tokens within each view and applies a shared two-layer MLP:

$$\begin{aligned} \bar{h}_v &= \frac{1}{N}\sum_{j=1}^{N} D_{v,j}, \\ b_v &= \mathrm{GELU}\big(f_1(\mathrm{LN}_C(\bar{h}_v))\big), \qquad r_v = f_2(b_v), \\ f_1 &: \mathbb{R}^C \to \mathbb{R}^d, \qquad f_2 : \mathbb{R}^d \to \mathbb{R}^{(K+1)d}, \end{aligned} \tag{58}$$

where $d = 256$, $K = 128$, and both affine maps include biases. The output is organized into memory slots and normalized along each slot's feature dimension:

$$S_v = \mathrm{LN}_d\Big(\mathrm{reshape}_{(K+1)\times d}(r_v)\Big) \in \mathbb{R}^{(K+1)\times d}. \tag{59}$$

The first slot of each view contributes to the global scene latent. The remaining slots provide the corresponding view's local conditions:

$$\begin{aligned} g_v &= S_v[0], \qquad L_v = S_v[1:K+1] \in \mathbb{R}^{K\times d}, \\ z_s &= \frac{1}{V}\sum_{v=1}^{V} g_v, \\ \mathcal{M}_s &= \mathrm{Concat}_{\mathrm{slots}}(S_1, \ldots, S_V) \in \mathbb{R}^{V(K+1)\times d}. \end{aligned} \tag{60}$$

Dense feature prediction for view v uses L_v, and PBR parameter decoding and illumination estimation use z_s as their global condition. The illumination predictor additionally reads the complete scene memory $\mathcal{M}_s$.

G.2 LOCALLY CONDITIONED DENSE MATERIAL FEATURES

The independent residual DPT decoder follows the multiscale reassembly and fusion structure of DPT (Ranftl et al., 2021). For backbone b, let $\mathcal{L}_b = (\ell_{b,1}, \ell_{b,2}, \ell_{b,3}, \ell_{b,4})$ denote the four intermediate layers selected by its feature interface and ordered from fine to coarse. The layer indices are backbone-dependent and are selected from the available intermediate outputs of that backbone rather than fixed absolute depths. Let $\mathcal{R}_j$ reassemble, rescale, and project tokens from the j-th selected layer, with reassembly scales indexed from fine to coarse. Let $\mathcal{F}_j$ denote a DPT fusion unit containing residual convolutions and upsampling. The dense feature map is

$$\begin{aligned} A_{v,j} &= \mathcal{R}_j(D_v^{(\ell_{b,j})}), \qquad j = 1,2,3,4, \\ B_{v,4} &= \mathcal{F}_4(A_{v,4}), \\ B_{v,j} &= \mathcal{F}_j(B_{v,j+1}, A_{v,j}), \qquad j = 3,2,1, \\ U_v &= \mathrm{Up}_{H,W}(\mathrm{Conv}_{3\times 3}(B_{v,1})). \end{aligned} \tag{61}$$

Each reassembly module restores a spatial grid, applies a 1×1 projection and its scale transformation, and uses a 3×3 convolution to obtain a common feature width. A fusion unit processes the skip

feature with a residual convolutional block, adds the coarser input, and applies another residual block, twofold bilinear upsampling, and a 1×1 projection.

The dense output receives a convolutional residual and an image-conditioned residual:

$$\begin{aligned} U_{v,r} &= U_v + \tanh(a_r)\ \mathrm{Conv}^{(2)}_{3\times3}\Big(\mathrm{GELU}(\mathrm{Conv}^{(1)}_{3\times3}(U_v))\Big), \\ U_{v,i} &= U_{v,r} + \tanh(a_i)\,\phi_{\mathrm{img}}(I_v). \end{aligned} \tag{62}$$

Here, I_v is the normalized context image and ϕ_{img} contains three 3×3 convolutions with GELU activations. The dense features query the local scene memory through

$$\begin{aligned} Q_v &= \mathrm{Flatten}_{H,W}(\mathrm{Conv}_{1\times1}(U_{v,i})) \in \mathbb{R}^{HW\times d}, \\ A^{\mathrm{mem}}_v &= \mathrm{softmax}_K\left(\frac{Q_v L_v^\top}{\sqrt{d}}\right). \end{aligned} \tag{63}$$

The softmax is taken over the K local slots. An affine map f_o projects the retrieved memory to the dense feature width, and an affine map f_g produces a channel gate:

$$\begin{aligned} C^{\mathrm{mem}}_v &= \mathrm{Reshape}_{H,W}(f_o(A^{\mathrm{mem}}_v L_v)), \\ \eta_v &= \sigma\left(f_g\left(\frac{1}{K}\sum_{r=1}^{K} L_v[r]\right)\right), \\ U_{v,s} &= U_{v,i} + \tanh(a_s)\,\eta_v \odot C^{\mathrm{mem}}_v. \end{aligned} \tag{64}$$

The sigmoid is denoted by σ, and $\odot$ is elementwise multiplication with the gate broadcast over spatial positions. The scalar gates a_r, a_i, a_s are initialized to zero. The output head yields a material feature at each context pixel:

$$f_v = \mathrm{Conv}_{1\times1}(\mathrm{ReLU}(\mathrm{Conv}_{3\times3}(U_{v,s}))). \tag{65}$$

The material feature f_v has 64 channels and is the only dense adapter feature passed to the subsequent 3D refinement. Final PBR parameters are decoded from the rendered target-view features.

G.3 Multi-view Voxel Fusion

Index context pixels by i. Let p_i be the initial Gaussian center, ρ_i its original predicted attributes, and c_i the opacity logit within ρ_i. We associate the pixel's material feature f_i with these attributes and group candidates by position:

$$\begin{aligned} x_i &= [\rho_i; f_i], \\ r_i &= \mathrm{round}(p_i/h) \in \mathbb{Z}^3, \qquad h = 0.002, \\ \mathcal{V}_r &= \{i : r_i = r\}, \qquad \mathcal{A} = \{r : \mathcal{V}_r \neq \varnothing\}. \end{aligned} \tag{66}$$

The voxel width h is expressed in the current scene coordinate system, and $\mathcal{A}$ is the set of occupied voxels. The opacity logits determine the within-voxel fusion weights:

$$\begin{aligned} m_r &= \max_{j\in\mathcal{V}_r} c_j, \\ w_i &= \frac{\exp(c_i - m_r)}{\sum_{j\in\mathcal{V}_r}\exp(c_j - m_r) + \epsilon_v}, \qquad i \in \mathcal{V}_r, \quad \epsilon_v = 10^{-6}, \\ \bar{p}_r &= \sum_{i\in\mathcal{V}_r} w_i p_i, \qquad \bar{x}_r = \sum_{i\in\mathcal{V}_r} w_i x_i. \end{aligned} \tag{67}$$

Thus, $\bar{p}_r$ is the aggregated candidate position and $\bar{x}_r$ combines the associated Gaussian attributes and material features.

G.4 Sparse 3D Refinement

The sparse convolution uses the occupied voxel itself and its six axis-aligned neighbors. For hidden features H_r, layer j computes

$$\begin{aligned} \Delta &= \{(0,0,0), (\pm1,0,0), (0,\pm1,0), (0,0,\pm1)\}, \\ \mathrm{SC}_j(H)_r &= W_{j,\mathrm{pw}}\left(\sum_{\substack{\delta\in\Delta \\ r+\delta\in\mathcal{A}}} w_{j,\delta} \odot H_{r+\delta}\right) + b_{j,\mathrm{pw}}. \end{aligned} \tag{68}$$

Here, $w_{j,\delta}$ is a learned per-channel weight for offset δ, and $W_{j,\mathrm{pw}}$ mixes channels after neighborhood aggregation. Stacking the fused attributes as $\bar{X}$, the two convolutional layers and attribute residual are

$$\begin{aligned} H_0 &= \mathrm{LN}_0\big(\mathrm{GELU}(f_{\mathrm{in}}(\bar{X}))\big)\,, \\ H_1 &= \mathrm{ReLU}(\mathrm{LN}_1(\mathrm{SC}_1(H_0)))\,, \\ H_2 &= \mathrm{ReLU}(\mathrm{LN}_2(\mathrm{SC}_2(H_1)) + H_0)\,, \\ \widetilde{X} &= \bar{X} + \mathrm{GELU}(f_{\mathrm{out}}(H_2)). \end{aligned} \tag{69}$$

The input and output maps are affine, and the weights and bias of f_{out} are initialized to zero. Each refined vector $\widetilde{x}_r$ supplies Gaussian attributes and material features. Let $\widetilde{c}_r$, $\widetilde{s}_r$, and $\widetilde{q}_r$ denote its raw opacity, scale, and rotation parameters. Their rendering parameterization is

$$\begin{aligned} o_r &= \sigma(\widetilde{c}_r), \\ s_r &= \min(0.001\,\mathrm{softplus}(\widetilde{s}_r), 0.3)\,, \\ q_r^{\mathrm{rot}} &= \frac{\widetilde{q}_r}{\|\widetilde{q}_r\|_2 + 10^{-8}}, \\ \mu_r &= \bar{p}_r, \qquad \Sigma_r = R(q_r^{\mathrm{rot}})\,\mathrm{diag}(s_r^2)R(q_r^{\mathrm{rot}})^\top. \end{aligned} \tag{70}$$

The scale mapping acts componentwise, and $R(\cdot)$ converts a quaternion to a rotation matrix. Refinement updates opacity, scale, rotation, and material features while retaining the fused centers and occupied voxel set. With refined material features $\widetilde{f}_r$, the PBR representation is $\mathcal{G}_{\mathrm{PBR}} = \{(\mu_r, \Sigma_r, o_r, \widetilde{f}_r)\}_{r\in\mathcal{A}}$.

G.5 Post-Splat Material Decoding

For target camera π_t and pixel u, let $\mathcal{N}_r^{(t)}(u)$ be the projected Gaussian kernel, and let $j \prec_t r$ denote an earlier contributor in front-to-back compositing order. Following Eq. (1), the feature map is

$$\begin{aligned} \bar{w}_{r,t}(u) &= o_r \mathcal{N}_r^{(t)}(u) \prod_{j\prec_t r} \Big(1 - o_j \mathcal{N}_j^{(t)}(u)\Big)\,, \\ F^t(u) &= \sum_{r\in\mathcal{A}} \bar{w}_{r,t}(u)\,\widetilde{f}_r. \end{aligned} \tag{71}$$

The shared material decoder transforms the rendered features and applies a condition derived from the global scene latent:

$$\begin{aligned} Z^t &= \mathrm{ReLU}\big(\mathrm{GN}_8(\mathrm{Conv}_{1\times1}(F^t))\big)\,, \\ [\gamma_s; \beta_s] &= f_{m,2}(\mathrm{SiLU}(f_{m,1}(z_s)))\,, \\ \widetilde{Z}^t &= (1+\gamma_s)\odot Z^t + \beta_s. \end{aligned} \tag{72}$$

The shared convolution preserves 64 channels and uses eight-group GroupNorm. The affine maps have dimensions $f_{m,1} : \mathbb{R}^{256} \to \mathbb{R}^{128}$ and $f_{m,2} : \mathbb{R}^{128} \to \mathbb{R}^{128}$. The vectors $\gamma_s, \beta_s \in \mathbb{R}^{64}$ are broadcast over spatial positions and shared across the target views of the scene. Four 1×1 heads predict the PBR parameters:

$$\begin{aligned} A^t &= \sigma(\mathcal{C}_A(\widetilde{Z}^t)), \qquad N^t = \mathrm{norm}_{10^{-6}}(\mathcal{C}_N(\widetilde{Z}^t)), \\ R^t &= \sigma(\mathcal{C}_R(\widetilde{Z}^t)), \qquad M^t = \sigma(\mathcal{C}_M(\widetilde{Z}^t)), \\ P^t &= [A^t; N^t; R^t; M^t], \qquad R^t_{\mathrm{shade}} = 0.04 + 0.96R^t. \end{aligned} \tag{73}$$

Here, $\mathrm{norm}_\epsilon(x) = x/\max(\|x\|_2, \epsilon)$, and A^t, N^t, R^t, and M^t are albedo, scene-coordinate normals, roughness, and metallicity, respectively. Normals are normalized per pixel, and shading and material smoothness use R^t_{shade}.

Illumination interface. The illumination head combines the rendered target-view features with all slots in $\mathcal{M}_s$ and uses z_s for global conditioning. Its light queries produce a shared scene cubemap for the PBR shader. The query and cubemap dimensions are given in Appendix H.

H Additional Implementation Details

Cross-backbone integration. For the cross-backbone study, each backbone is initialized from its public checkpoint, set to evaluation mode, and excluded from gradient updates along its original geometry-reconstruction, camera-prediction, opacity, base-appearance, and Gaussian-scaffold pathways. We train a separate adapter for each backbone without transferring adapter weights. In the DA3 integration, the pretrained GS-DPT head predicts opacity, rotation, scale, and appearance from DA3 tokens, while unprojected depth provides Gaussian centers (Lin et al., 2026). The adapters share the same functional design, while their scene-memory extractors and latent-assignment heads follow the corresponding feature interfaces. In the executed integrations, AnySplat, DA3, and SPFSplat use pose-free context cameras, whereas C3G, FLARE, NoPoSplat, and YoNoSplat use ground-truth context-camera parameters. The target/render camera source follows each backbone-specific interface and is held fixed between Vanilla and Ours-PBR within every matched pair.

Adapter architecture. The material latent, global scene descriptor, and image-space material-decoder trunk have dimensions of 64, 256, and 64, respectively. Backbone-specific assignment heads map the available intermediate features to the shared 64-dimensional material interface. The shared decoder uses GroupNorm with eight groups and converts rasterized latent features into PBR parameters, with roughness restricted to $[0.04, 1.0]$. Unless explicitly marked as a Direct-PBR or pre-splat bottleneck ablation, all reported results use this Feature-Decoder path. The illumination predictor uses eight light queries and two four-head cross-attention layers to predict an HDR cubemap comprising six 256×256 faces.

For data sampling, each training scene contains RGB images and may optionally include PBR parameter maps for auxiliary supervision. Images and optional supervision maps are center-cropped within each batch to a common resolution divisible by 14. The background color is set to black, and the training seed is fixed to 114514 for synchronized distributed sampling.

For each matched comparison, Vanilla and Ours-PBR use the same evaluation resolution, valid-pixel mask, and exposure settings, without method-specific post-processing tuning. In the main PBR training configurations, SSAO and SSR are enabled, whereas tone mapping, gamma conversion, and random environment-map perturbations are disabled, deviations from these settings are reported only as ablations.

Optimization and compute. Following AnySplat (Jiang et al., 2025), we optimize each PBR adapter with AdamW using a peak learning rate of 2×10^{-4}, linear warmup for 1,000 iterations, and cosine decay, without gradient accumulation. All reported experiments run on a NVIDIA A100 PCIe (80 GB) GPU. We disable `torch.compile` for stability under dynamic batches, and clip the maximum gradient norm to 5.0. For the six shared-loss adapters, the masked PBR reconstruction uses $\lambda_{\text{ssim}} = 0.2$. The remaining configured loss weights are $\lambda_{\text{mat}} = 10^{-3}$, $\lambda_{\text{ncons}} = 0.2$, $\lambda_{\text{ntv}} = 1.0$, $\lambda_{\log} = 0.1$, $\lambda_{\text{ratio}} = 0.025$, and $\lambda_{\text{scene}} = 0.1$. The executed C3G run instead uses unit-weight PBR MSE and disables these additional structural, shading, and scene-contrastive terms. Material TV uses edge-aware finite differences between valid neighboring pixels. Normal consistency is evaluated where both Gaussian coverage and depth-derived normals are valid, while normal TV uses normalized normals and a 3×3-eroded coverage mask. For scene contrastive learning, the target scene-latent generator and projection head are updated from their online counterparts by EMA with momentum coefficient $\mu = 0.99$. We use contrastive temperature $\tau_c = 0.07$, a FIFO memory capacity of 4096 keys, maximum memory age $T_{\max} = 2000$ optimization steps, and age-decay constant $\tau_a = 1000$ steps. Each context view is retained independently with probability 0.5, and both sampled subsets are constrained to be nonempty. Optional material-map supervision is disabled by default. SH-RGB reconstruction, environment-map TV, the physical material prior, opacity losses, depth supervision, and backbone-normal supervision are not included in the default objective.

I Training-Objective Details

I.1 Diffuse–Specular Shading Statistics

For each supervised target view v, let $L_v^d(u), L_v^s(u) \in \mathbb{R}^3$ denote the diffuse and specular RGB contributions returned by the PBR shader at pixel u. We define

$$\begin{aligned} \ell_v^d(u) &= \mathrm{mean}_c\big([L_v^d(u)]_+\big)\,, \quad \ell_v^s(u) = \mathrm{mean}_c([L_v^s(u)]_+)\,, \\ y_v(u) &= \max_c([I_v(u)]_+)\,, \qquad \eta_v^d(u) = \frac{\ell_v^d(u)}{\max(\ell_v^d(u)+\ell_v^s(u),\epsilon)}. \end{aligned} \tag{74}$$

Here, $[x]_+ = \max(x, 0)$ is applied elementwise and $c \in \{R, G, B\}$ indexes the RGB channels. Thus, ℓ_v^d and ℓ_v^s are scalar diffuse and specular intensities, y_v is a maximum-channel target-intensity proxy, and η_v^d is the diffuse fraction.

Let $\Omega_v = \{u \mid M_v(u) = 1\}$ be the valid Gaussian-coverage region and $N_\Omega = \sum_v |\Omega_v|$, where all sums run over the supervised target views. The two terms combined in Eq. (7) are

$$\begin{aligned} \mathcal{L}_{\log} &= \frac{1}{N_\Omega + \epsilon} \sum_v \sum_{u\in\Omega_v} \eta_v^d(u)\,\big|\log\big(1+\ell_v^d(u)+\ell_v^s(u)\big) - \log(1+y_v(u))\big|\,, \\ \mathcal{L}_{\mathrm{ratio}} &= \frac{\sum_v \sum_{u\in\Omega_v} \ell_v^s(u)}{\sum_v \sum_{u\in\Omega_v} \ell_v^d(u) + \epsilon}. \end{aligned} \tag{75}$$

The same small positive constant ϵ is used for numerical stability. The first term is normalized by the number of valid pixels, without renormalizing the diffuse weight, while the second is computed after summing over all valid pixels.

I.2 Scene Contrastive Descriptors and Memory

Let E_θ and P_θ denote the online scene-latent generator and contrastive projector, with EMA counterparts E_ξ and P_ξ. The generator is defined in Appendix G.1. Scene contrastive learning uses the already contextualized backbone features $D = \{D_v\}_{v=1}^V$. The online and EMA target networks independently sample their view-retention masks. If a mask is empty, one random view is retained:

$$\begin{aligned} &\widetilde{m}_v^a \sim \mathrm{Bernoulli}(p), \qquad a \in \{q, k\}, \quad p = 0.5, \\ &m^a = \begin{cases} \widetilde{m}^a, & \sum_v \widetilde{m}_v^a > 0, \\ e_{v^*}, & \sum_v \widetilde{m}_v^a = 0, \end{cases} \qquad v^* \sim \mathrm{Uniform}\{1,\dots,V\}. \end{aligned} \tag{76}$$

The vector e_{v^*} has a single nonzero entry at v^*. These masks select the view memories after backbone feature extraction. The two global descriptors are

$$g^q = \frac{\sum_v m_v^q E_\theta(D)_v[0]}{\sum_v m_v^q}, \qquad g^k = \frac{\sum_v m_v^k E_\xi(D)_v[0]}{\sum_v m_v^k}. \tag{77}$$

The contrastive projector consists of LayerNorm and two affine maps with an intermediate GELU. At step n, the normalized query and key are

$$\begin{aligned} P_\theta(g) &= f_{p,2}(\mathrm{GELU}(f_{p,1}(\mathrm{LN}_d(g))))\,, \\ q_n &= \mathrm{norm}_{10^{-8}}(P_{\theta_n}(g^q)), \\ k_n &= \mathrm{sg}\big[\mathrm{norm}_{10^{-8}}(P_{\xi_n}(g^k))\big]\,. \end{aligned} \tag{78}$$

Both affine maps preserve $d = 256$ channels, $\mathrm{norm}_\epsilon(x) = x/\max(\|x\|_2, \epsilon)$, and sg denotes stop-gradient.

For the single-scene training step, the historical queue $\mathcal{Q}_n$ stores each key k_j with its scene identity y_j and generation step j. For the current scene y_n, the candidate indices, positive indices, and age weights are

$$\begin{aligned} &\mathcal{J}_n = \{n\} \cup \{j : (k_j, y_j, j) \in \mathcal{Q}_n,\ 0 \le n - j \le T_{\max}\}, \\ &\mathcal{P}_n = \{j \in \mathcal{J}_n : y_j = y_n\}, \\ &a_j = n - j, \qquad \omega_j = \exp(-a_j/\tau_a). \end{aligned} \tag{79}$$

Current and historical keys from the same scene are positive examples, valid historical keys from other scenes are negative examples. The scene-level contrastive objective is

$$\mathcal{L}_{\mathrm{scene},n} = -\log \frac{\sum_{j\in\mathcal{P}_n} \omega_j \exp(q_n^\top k_j/\tau_c)}{\sum_{j\in\mathcal{J}_n} \omega_j \exp(q_n^\top k_j/\tau_c)}. \tag{80}$$

This term enters the total objective in Section 4.3 with $\lambda_{\mathrm{scene}} = 0.1$. We use $\tau_c = 0.07$, $T_{\max} = 2000$, and $\tau_a = 1000$.

Following momentum contrast (He et al., 2020), the target parameters track the online parameters by EMA. After the optimizer produces θ_{n+1}, the update and queue insertion are

$$\begin{aligned} \xi_{n+1} &= \mu\xi_n + (1-\mu)\theta_{n+1}, \qquad \mu = 0.99, \\ \mathcal{Q}_{n+1} &= \mathrm{Tail}_{4096}(\mathcal{Q}_n \,\|\, (k_n, y_n, n)) \,. \end{aligned} \tag{81}$$

Here, θ and ξ include both the scene-latent generator and the projector, and Tail_{4096} retains the most recent 4096 records. The enqueued key k_n is the one computed with ξ_n during the current forward pass. The projected global descriptors serve the contrastive objective, while material and illumination prediction use the full-view online memory $\mathcal{M}_s$ and global latent z_s. Local latents receive task gradients through the prediction modules they condition.