LUMEN-GS: TOWARDS LIGHT-DISENTANGLED AND INTRINSIC-AWARE GAUSSIAN SPLATTING FOR FEED-FORWARD SPARSE-VIEW STREET RECONSTRUCTION

Anonymous authors
Paper under double-blind review

ABSTRACT

Feed-forward reconstruction models have recently advanced spatial intelligence and simulation but mainly predict color radiance fields, treating scenes as non-interactive assets without exploring intrinsic properties and lighting representations for controllable editing. Bridging this gap requires a feed-forward paradigm that jointly represents intrinsic properties and outdoor illumination, instead of the hours of per-scene optimization required by iterative inverse rendering. Therefore, we present Lumen-GS, to our knowledge the first feed-forward reconstruction system that integrates a physically based rendering (PBR) pipeline for street scenes. In a single forward pass, it jointly predicts intrinsic primitives, sunlight and global environment radiance within a sunlight-integrated representation from sparse-view images at a single timestamp, supporting reconstruction, relighting, and harmonization. Lumen-GS aggregates multi-view features via a Hybrid Ring representation, with sunlight direction from GPS/IMU metadata. It further introduces a Physically-Aligned Decoder (PAD), whose intrinsic prediction heads are jointly supervised by teacher pseudo-labels and Physically-Based Rendering to improve disentanglement. Results on nuScenes show that Lumen-GS remains competitive with the strongest feed-forward baselines in novel-view synthesis and with iterative inverse-rendering methods in PBR quality, while supporting object insertion, harmonization, and data augmentation for downstream perception. These results demonstrate the potential of Lumen-GS for lighting-controllable street-scene simulation.

1 INTRODUCTION

3D scene reconstruction is a fundamental task in computer vision and graphics, underpinning applications such as autonomous driving simulation, virtual reality, and digital twins. Recent advances in feed-forward reconstruction methods (Chen et al., 2024; Charatan et al., 2024; Miao et al., 2025; Yang et al., 2025; Wei et al., 2025; Xu et al., 2025; Shi et al., 2026) enable scene geometry and appearance to be inferred directly from multi-view images through a single network forward pass. As a result, these methods have substantially accelerated progress in spatial intelligence and simulation.

Current feed-forward reconstruction methods predominantly produce pixel- or voxel-aligned color radiance fields, modeling appearance as a mapping from features to color values. Although effective for photorealistic novel-view synthesis, they often fail to capture intrinsic properties that are independent of viewpoint and illumination. As a result, these representations remain entangled with appearance, limiting their ability to support downstream tasks such as relighting, object harmonization, and weather simulation, and ultimately constraining their generalization and practical utility.

Recent intrinsic scene decomposition methods (Jiang et al., 2024; Kaleta et al., 2025; Shi et al., 2025; Li et al., 2025; Zhang et al., 2023a; Lin et al., 2025a; Gao et al., 2024) predominantly cast the task as iterative inverse rendering under a physically based rendering (PBR) model, fitting material parameters through physics-governed optimization. This typically takes hours per scene, making large-scale data accumulation and real-time applications impractical. Combining the efficiency of feed-forward models with inverse rendering promises to overcome these limitations, yet two challenges remain. First, real street datasets provide neither ground truth for intrinsic properties such as normals, albedo, and roughness, nor for illumination parameters such as sun intensity.

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.

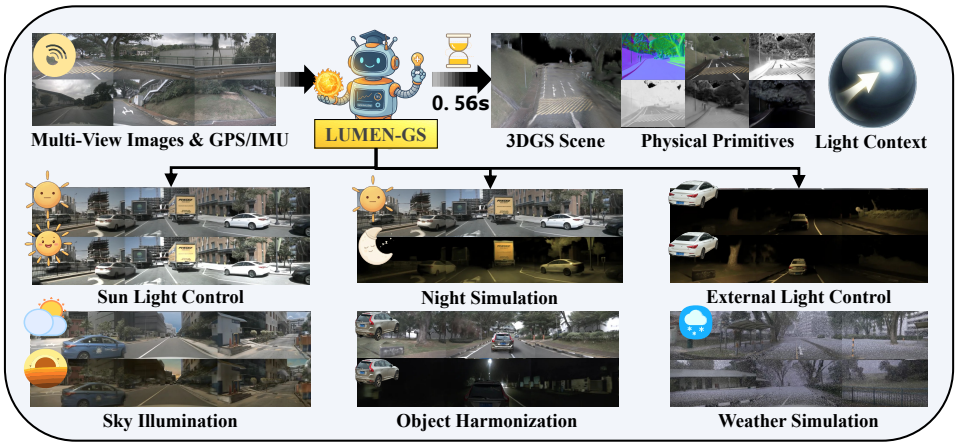

Figure 1: Given surround-view images at a single timestamp and the corresponding GPS/IMU data, our model predicts the intrinsic primitives and environmental light context of the scene. By further adjusting the lighting parameters, the model can support various downstream tasks.

Feed-forward models must therefore recover these properties in the absence of physically grounded supervision. Second, a sparse-view input (particularly images at single timestamp) leaves a severe ambiguity between geometry and lighting (Noras et al., 2026).

To address these challenges, we introduce **Lumen-GS**, a light-disentangled Gaussian Splatting model that takes *surround-view images at a single timestamp* and compresses hours of per-scene optimization into a 0.56-second forward pass, jointly predicting intrinsic primitives and illumination parameters. We represent the scene with 3D Gaussian Splatting (3DGS) primitives augmented with physically meaningful properties and a light context, disentangling outdoor illumination. To restore the missing physically grounded supervision, we supervise the expert heads with pseudo-labels from pretrained teachers and a geometry-derived visibility constraint. To resolve the ambiguity induced by such limited observations, we further introduce a Physically-Aligned Decoder (PAD), which injects the physical constraints between material and illumination properties into the decoder architecture and feature interactions, so that both rasterizers jointly supervise the expert heads, improving material–lighting disentanglement. Experiments show that Lumen-GS remains competitive with the strongest feed-forward baselines in novel-view synthesis and with iterative inverse-rendering methods in PBR rendering quality, and supports object insertion and harmonization, while also improving detector performance. Our main contributions are summarized as follows:

- We propose Lumen-GS, a light-disentangled and intrinsic-aware Gaussian Splatting model for feed-forward sparse-view street-scene reconstruction. Lumen-GS predicts geometry, intrinsic primitives, and outdoor illumination parameters in a single forward pass, enabling several downstream tasks.
- We introduce the Physically-Aligned Decoder (PAD), which incorporates material–lighting physical structure into latent-space decoding and jointly supervises the expert heads with teacher pseudo-labels and both rasterizers, improving material–lighting disentanglement.
- Experimental results demonstrate that Lumen-GS remains competitive with the strongest feed-forward baselines in novel-view synthesis and with iterative inverse-rendering methods in PBR rendering quality, while supporting object insertion, harmonization, and data augmentation.

2 Related Work

Optimization-based Inverse Rendering. Inverse rendering disentangles 3D geometry, material properties, and environmental illumination from 2D observations. Early implicit methods (Zhang et al., 2023b; Wu et al., 2023; Jin et al., 2023; Yao et al., 2022; Zhang et al., 2023a) incorporate physically based rendering (PBR) into neural fields (Mildenhall et al., 2021), while recent explicit representations such as Gaussian Splatting (Kerbl et al., 2023) improve rasterization efficiency. GIR (Shi et al., 2025) combines the shortest eigenvector of the 3D Gaussian covariance matrix with directional masking for accurate normal estimation and scene factorization. ReCap (Li et al., 2025) leverages cross-environment captures and optimizes multiple unknown lighting conditions as a multi-task problem to recover shared material properties. R3DG (Gao et al., 2024) decomposes BRDFs on

Gaussian point clouds with ray tracing, while GI-GS (Chen et al., 2025a) models global illumination on Gaussians via deferred-shading path tracing. For outdoor scenes, LumiGauss (Kaleta et al., 2025) builds on 2D Gaussian Splatting (2DGS) (Huang et al., 2024) and integrates Precomputed Radiance Transfer (PRT) with spherical harmonics, enabling realistic shadow modeling and compatibility with traditional graphics engines. DiffusionRenderer (Liang et al., 2025) further introduces diffusion priors for inverse rendering and relighting. UrbanIR (Lin et al., 2025a), InvRGB+L (Chen et al., 2025b), and OSDR-GS (Feng et al., 2025b) extend relighting to large-scale outdoor and driving scenes, modeling the interactions among sunlight, sky illumination, and urban geometry. Our method inherits their sunlight/sky lighting and intrinsic modeling, but learns it in a feed-forward manner.

Feed-forward Reconstruction. To overcome the bottleneck of per-scene optimization, recent work shifted to feed-forward architectures. Several works (Fei et al., 2024; Chen et al., 2025c; Wang et al., 2024; 2025; Min et al., 2024; Liu et al., 2025) directly regress 3D representations from input views. PixelSplat (Charatan et al., 2024) uses epipolar geometry and pixel-aligned features for single-pass 3D Gaussian inference. MVSplat (Chen et al., 2024) constructs 3D cost volumes via plane sweeping to improve cross-view consistency. DepthSplat (Xu et al., 2025) incorporates depth priors to improve geometry and convergence. For street-view scenes, Omni-Scene (Wei et al., 2025) proposes a dual-branch framework that combines pixel- and volume-based Gaussians via TPV (tri-view plane) (Huang et al., 2023). EVolSplat (Miao et al., 2025) predicts volume-based Gaussians with sparse convolutions, handling distant views and sky regions. XYZCylinder (Yu et al., 2025) lifts multi-view features onto a unified cylinder for camera-compatible Gaussian prediction. To model dynamic elements, STORM (Yang et al., 2025) uses a Transformer to infer spatio-temporal representations, including 3D Gaussians and their velocities. UniSplat (Shi et al., 2026) introduces a 3D latent scaffold to unify spatial and temporal feature fusion. Although these models reconstruct geometry and radiance quickly, feed-forward inverse rendering and relighting remain challenging. We bridge this gap by unifying the two in one pass, reconstructing light-interactive scenes.

3 METHOD

In this section, we detail the proposed 3D feed-forward model with disentanglement of light and intrinsic primitives for sparse-view street scenes. As illustrated in Fig. 2, the model takes multi-view driving images from a single timestamp as input and efficiently aggregates scene features (Sec. 3.2) by constructing a Hybrid Ring representation. By decoding intrinsic primitives, light parameters (Sec. 3.3), and rendering with a Physically-Based Rendering (PBR) pipeline (Sec. 3.1), we achieve explicit disentanglement of geometry, materials, and environmental lighting.

3.1 SUNLIGHT-INTEGRATED PBR AND PHYSICAL 3DGS REPRESENTATION

Unlike indoor and object-level settings, outdoor environments necessitate the modeling of strong directional sunlight. Since conventional PBR formulations do not explicitly account for such highly directional illumination, we adopt the sun-modeling methodology proposed in InvRGB+L (Chen et al., 2025b) together with the Gaussian representation introduced in R3DG (Gao et al., 2024). Formally, the sunlight-integrated PBR formulation is given in Eq. (1). We further implement a differentiable CUDA kernel of this PBR pipeline, which accelerates computation while reducing memory consumption (see appendix A.6).

$$\mathbf{r}_o(\boldsymbol{\omega}_o, \boldsymbol{\omega}_{\text{sun}}, \mathbf{x}) = \underbrace{\int_\Omega \mathbf{f}(\boldsymbol{\omega}_o, \boldsymbol{\omega}_i, \mathbf{x})\, \mathbf{r}_{\text{env}}(\boldsymbol{\omega}_i, \mathbf{x})\, (\boldsymbol{\omega}_i \cdot \mathbf{n})_+ \, d\boldsymbol{\omega}_i}_{\text{Environment illumination term}} + \underbrace{\mathbf{f}(\boldsymbol{\omega}_o, \boldsymbol{\omega}_{\text{sun}}, \mathbf{x})\, \mathbf{r}_{\text{sun}}(\mathbf{x})\, (\boldsymbol{\omega}_{\text{sun}} \cdot \mathbf{n})_+}_{\text{Direct sunlight term}}. \quad (1)$$

Here, $\mathbf{r}_o(\boldsymbol{\omega}_o, \boldsymbol{\omega}_{\text{sun}}, \mathbf{x})$ denotes the outgoing radiance at surface point $\mathbf{x} \in \mathbb{R}^3$ along the viewing direction $\boldsymbol{\omega}_o \in \mathbb{R}^3$ and sun direction $\boldsymbol{\omega}_{\text{sun}}$. The environment radiance is defined as $\mathbf{r}_{\text{env}}(\boldsymbol{\omega}_i, \mathbf{x}) = \mathbf{v}_{\text{global}}(\boldsymbol{\omega}_i, \mathbf{x})\, \mathbf{r}_{\text{global}}(\boldsymbol{\omega}_i) + \mathbf{r}_{\text{local}}(\mathbf{x})$, where $\mathbf{v}_{\text{global}}(\boldsymbol{\omega}_i, \mathbf{x})$ denotes the global environment visibility at $\mathbf{x}$ along direction $\boldsymbol{\omega}_i \in \mathbb{R}^3$, $\mathbf{r}_{\text{global}}(\boldsymbol{\omega}_i)$ denotes the scene-level global environment radiance, and $\mathbf{r}_{\text{local}}(\mathbf{x})$ denotes the point-dependent local illumination residual. The sunlight contribution is defined as $\mathbf{r}_{\text{sun}}(\mathbf{x}) = v_{\text{sun}}(\mathbf{x}, \boldsymbol{\omega}_{\text{sun}})\, r_{\text{sun}}$, where $v_{\text{sun}}(\mathbf{x})$ is the sunlight visibility at $\mathbf{x}$ and $r_{\text{sun}} \in \mathbb{R}^3$ is the sunlight intensity. $\mathbf{f}(\boldsymbol{\omega}_o, \boldsymbol{\omega}_i, \mathbf{x})$ is the BRDF that relates incoming radiance from $\boldsymbol{\omega}_i$ to outgoing radiance along $\boldsymbol{\omega}_o$. We use the GGX-based microfacet model (Walter et al., 2007; Karis & Games, 2013) to represent specular reflection. Here, Ω denotes the hemisphere centered on the normal $\mathbf{n} \in \mathbb{R}^3$

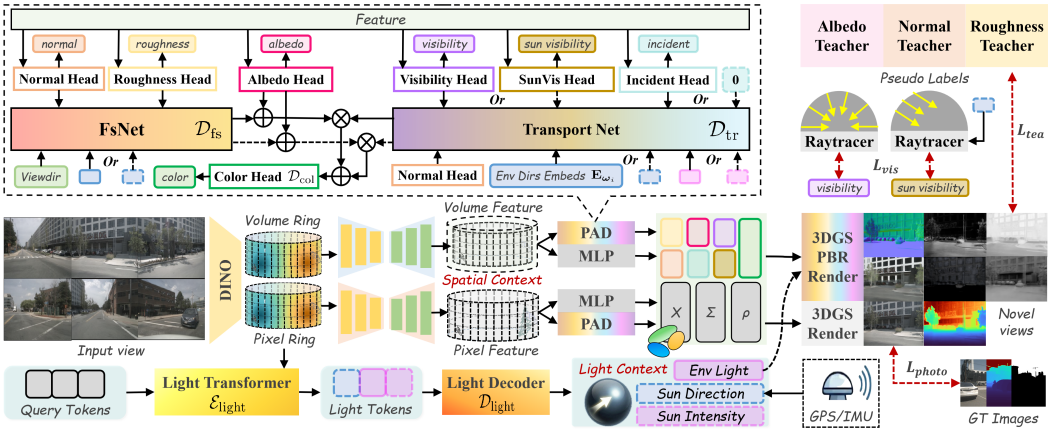

Figure 2: **Overview of Lumen-GS.** The model comprises spatial feature and light context aggregation, geometric and intrinsic properties decoding, and teacher-guided rendering supervision.

and $(\cdot)_+ = max(\cdot, 0)$ denotes the clamped dot product. To reduce disentanglement ambiguity and improve training stability, we replace the spherical harmonics-based environmental illumination with 3-channel RGB radiance.

We employ 3D Gaussian Splatting (3DGS) (Kerbl et al., 2023) primitives to parameterize scene in 3D space. In the standard 3DGS formulation, each Gaussian primitive is parameterized by its center $\mathbf{X}$, covariance $\mathbf{\Sigma}$, opacity ρ, and appearance color $\mathbf{C}$. Building upon these canonical properties, we further introduce physically meaningful properties for relightable rendering. Specifically, each Gaussian is represented as $G = \{\mathbf{X}, \mathbf{\Sigma}, \rho, \mathbf{C}, \mathbf{n}, \mathbf{C}_{\text{B}}, \mathbf{r}_{\text{local}}, R, \mathbf{v}_{\text{global}}, v_{\text{sun}}\}$.

The properties in our method are normal $\mathbf{n}$, albedo $\mathbf{C}_{\text{B}} \in \mathbb{R}^3$, roughness $R \in \mathbb{R}$, local illumination residual $\mathbf{r}_{\text{local}} \in \mathbb{R}^3$, global environment visibility $\mathbf{v}_{\text{global}} \in \mathbb{R}^{16}$ (Third-order spherical harmonic), and sunlight visibility $v_{\text{sun}} \in \mathbb{R}$. Per-scene methods also attach intrinsic and illumination attributes to Gaussians (Shi et al., 2025; Chen et al., 2025a); we instead predict them all feed-forward for outdoor scenes under sunlight and sky. Among them, $\mathbf{r}_{\text{local}}$, $\mathbf{v}_{\text{global}}$, and v_{sun} are point-dependent intrinsic properties stored in each Gaussian primitive, whereas $\mathbf{r}_{\text{global}}$ and r_{sun} are scene-level illumination properties. These additional properties enable each primitive to encode both geometric and physically based reflectance information, thereby supporting explicit sunlight-aware relighting in outdoor scenes.

3.2 Context Feature Aggregation

Spatial Context Aggregation. Given a set of multi-view driving images, our goal is to aggregate spatial features while preserving illumination cues, so as to disentangle intrinsic scene properties. Inspired by Omni-Scene (Wei et al., 2025) and XYZCylinder (Yu et al., 2025), we represent the scene using both pixel-based and volume-based features. In addition, we incorporate light context to guide subsequent decoding and improve illumination disentanglement. Specifically, the n-th view within the multi-view configuration can be parameterized as $\mathcal{C}_n = \{\mathbf{I}_n, f_n, \mathbf{P}_n, \mathbf{W}_n^e, \mathbf{c}_n^e\}$, where $\mathbf{I}_n \in \mathbb{R}^{3\times H\times W}$ is the image captured with the vertical Field of View (FoV) f_n, and $\mathbf{P}_n \in \mathbb{R}^{4\times4}$, $\mathbf{W}_n^e \in \mathbb{R}^{4\times4}$, and $\mathbf{c}_n^e \in \mathbb{R}^3$ denote the camera intrinsic matrix, the extrinsic transformation from the camera coordinate system to the ego coordinate system, and the camera position in the ego-vehicle coordinate system, respectively.

We leverage DINOv2 (Oquab et al., 2024) to extract image features $\mathbf{F}_n \in \mathbb{R}^{D\times H_f\times W_f}$ and a metric depth estimator to predict a depth map $\mathbf{D}_n \in \mathbb{R}^{1\times H\times W}$. Following the projection methodology proposed by (Yu et al., 2025), we project and aggregate these multi-view representations onto a ring plane. The pixel-ring feature $\mathbf{F}_{\text{pix}} \in \mathbb{R}^{D_p\times H_u\times W_u}$ and volume-ring feature $\mathbf{F}_{\text{vol}} \in \mathbb{R}^{D_v\times H_u\times W_u}$ are processed by two UNet-based networks and then sampled to produce the aggregated features $\hat{\mathbf{F}}_{\text{pix}} \in \mathbb{R}^{D_p\times N_p}$ and $\hat{\mathbf{F}}_{\text{vol}} \in \mathbb{R}^{D_v\times N_v}$.

Light Context Aggregation. We employ a Transformer-based encoder $\mathcal{E}_{\text{light}}$ that conditions on the pixel-ring feature $\mathbf{F}_{\text{pix}}$ to transform randomly initialized query tokens $\mathcal{T}_{\text{init}}$ into lighting features $\mathbf{F}_{\text{light}}$. These features are then fed into a lightweight decoder $\mathcal{D}_{\text{light}}$, which predicts the light context

$\{\mathbf{r}_{\text{global}}, r_{\text{sun}}\}$. Because the frequent absence of illumination cues under overcast and occluded weather conditions makes lighting estimation ill-posed, we infer the normalized solar direction $\boldsymbol{\omega}_{\text{sun}}$ of the current scene from GPS/IMU data as helpful prior.

3.3 PHYSICALLY-ALIGNED DECODER (PAD)

Traditional decoding paradigms in feed-forward reconstruction models typically rely on Dense Prediction Transformer (DPT) (Ranftl et al., 2021) or lightweight MLPs, assigning an independent prediction branch to each target attribute. By predicting these properties independently, such methods ignore the physical correlations among geometry, material, and illumination, which often leads to suboptimal disentanglement. To overcome this limitation, we propose PAD that explicitly exploits the physical constraints between material and illumination to decode the intrinsic properties for G.

We retain separate decoders for the geometry properties of the Gaussian primitives, using MLPs to predict the position $\mathbf{X}$, covariance $\mathbf{\Sigma}$, and opacity ρ. The appearance color $\mathbf{C}$ is not decoded independently, but is instead produced by the PAD. We further introduce expert heads for the intrinsic properties of the Gaussian primitives. We feed the aggregated features $\hat{\mathbf{F}}_{\text{pix}}$ and $\hat{\mathbf{F}}_{\text{vol}}$ into expert heads corresponding to individual intrinsic properties, producing the decoded outputs $\mathbf{n}$, $\mathbf{C}_{\text{B}}$, $\mathbf{r}_{\text{local}}$, R, $\mathbf{v}_{\text{global}}$, and v_{sun}, together with the intermediate latent features $\mathbf{F}_n$, $\mathbf{F}_{\text{B}}$, $\mathbf{F}_{\text{local}}$, $\mathbf{F}_R$, $\mathbf{F}_{\text{vis}}$, and $\mathbf{F}_{\text{sunvis}}$, each of which is a d-dimensional feature vector. These latent features are then fed into the PAD to decode the final appearance color $\mathbf{C}$. After decoding all Gaussian primitives, we assemble the final scene representation as $\mathcal{G} = \{G_i\}_{i=1}^{N_p+N_v}$.

The discrete PBR formulation is given in Eq. (2). It computes the interactions between the diffuse-specular BRDF term and the transport term for environmental illumination $\mathbf{M}_{\text{env}}, \mathbf{T}_{\text{env}}$ and sunlight $\mathbf{M}_{\text{sun}}, \mathbf{T}_{\text{sun}}$, respectively, and then combines the two contributions.

$$\begin{aligned}\mathbf{r}_o(\boldsymbol{\omega}_o, \boldsymbol{\omega}_{\text{sun}}, \mathbf{x}) \approx \sum\nolimits_{k=1}^{N} \frac{2\pi}{N} \Big[\mathbf{f}(\boldsymbol{\omega}_o, \boldsymbol{\omega}_k, \mathbf{x})\big(\mathbf{v}_{\text{global}}(\boldsymbol{\omega}_k, \mathbf{x})\mathbf{r}_{\text{global}}(\boldsymbol{\omega}_k) + \mathbf{r}_{\text{local}}(\mathbf{x})\big)(\boldsymbol{\omega}_k \cdot \mathbf{n})_+\Big] \\ + \mathbf{f}(\boldsymbol{\omega}_o, \boldsymbol{\omega}_{\text{sun}}, \mathbf{x})\, v_{\text{sun}}(\mathbf{x}, \boldsymbol{\omega}_{\text{sun}})\, r_{\text{sun}}\, (\boldsymbol{\omega}_{\text{sun}} \cdot \mathbf{n})_+ = \langle \mathbf{M}_{\text{env}}, \mathbf{T}_{\text{env}} \rangle + \mathbf{M}_{\text{sun}} \cdot \mathbf{T}_{\text{sun}}.\end{aligned} \tag{2}$$

To inject this physical structure at low cost, we perform neural PBR rendering in the latent space (Eq. (3)). For environmental illumination, we construct two subnetworks $\mathcal{D}_{\text{fs}}^{\text{env}}$ and $\mathcal{D}_{\text{tr}}^{\text{env}}$ to model the BRDF term and the light transport term, respectively. To avoid Monte Carlo integration over the hemisphere, we encode incident directions implicitly as $\mathbf{E}_{\boldsymbol{\omega}_i}$; both subnetworks consume this encoding with the corresponding physical features to predict the latent BRDF distribution $\mathbf{H}_{\text{M}}^{\text{env}}$ and transport distribution $\mathbf{H}_{\text{T}}^{\text{env}}$. For sunlight, we use decoders $\mathcal{D}_{\text{fs}}^{\text{sun}}$ and $\mathcal{D}_{\text{tr}}^{\text{sun}}$ with identical architectures but separate parameters to predict $\mathbf{H}_{\text{M}}^{\text{sun}}$ and $\mathbf{H}_{\text{T}}^{\text{sun}}$. The split is needed because the environment decoders predict features over N sampled directions, whereas the sunlight decoders predict along the single direction embedded as $\mathbf{E}_{\text{sun}}$.

$$\begin{aligned}\mathbf{H}_{\text{M}}^{\text{env}} &= \mathbf{F}_{\text{B}} + \mathcal{D}_{\text{fs}}^{\text{env}}(\mathbf{E}_{\boldsymbol{\omega}_i}, \mathbf{F}_n, \mathbf{F}_R, \mathbf{E}_{\text{view}}), \quad \mathbf{H}_{\text{T}}^{\text{env}} = \mathcal{D}_{\text{tr}}^{\text{env}}(\mathbf{E}_{\boldsymbol{\omega}_i}, \mathbf{F}_n, \mathbf{F}_{\text{vis}}, \mathbf{F}_{\text{local}}, \mathbf{F}_{\text{light}}), \\ \mathbf{H}_{\text{M}}^{\text{sun}} &= \mathbf{F}_{\text{B}} + \mathcal{D}_{\text{fs}}^{\text{sun}}(\mathbf{E}_{\text{sun}}, \mathbf{F}_n, \mathbf{F}_R, \mathbf{E}_{\text{view}}), \quad \mathbf{H}_{\text{T}}^{\text{sun}} = \mathcal{D}_{\text{tr}}^{\text{sun}}(\mathbf{E}_{\text{sun}}, \mathbf{F}_n, \mathbf{F}_{\text{sunvis}}, \vec{\mathbf{0}}, \mathbf{F}_{\text{light}}).\end{aligned} \tag{3}$$

Since $\mathbf{H}$ already aggregates the multi-directional information, we couple the BRDF and transport latents with an elementwise product and map the result to the Gaussian color along $\mathbf{E}_{\text{view}}$ with a prediction head: $\mathbf{C} = \mathcal{D}_{\text{col}}(\mathbf{H}_{\text{M}}^{\text{env}} \odot \mathbf{H}_{\text{T}}^{\text{env}} + \mathbf{H}_{\text{M}}^{\text{sun}} \odot \mathbf{H}_{\text{T}}^{\text{sun}})$.

3.4 TRAINING OBJECTIVES

In addition to the RGB image $\hat{\mathbf{I}}$, depth map $\hat{\mathbf{D}}$, and opacity map $\hat{\mathbf{O}}$ rendered by the standard 3DGS rasterizer Π, for each novel view $\mathcal{C}$, the rendered PBR image $\hat{\mathbf{I}}_{\text{pbr}}$, normal map $\hat{\mathbf{I}}_n$, albedo map $\hat{\mathbf{I}}_{\text{B}}$, and roughness map $\hat{\mathbf{I}}_{\text{R}}$ are computed as shown in Eq. (4):

$$\hat{\mathbf{I}}_{\text{pbr}}, \hat{\mathbf{I}}_n, \hat{\mathbf{I}}_{\text{B}}, \hat{\mathbf{I}}_{\text{R}} = \Pi_{\mathcal{C}}^{\text{pbr}}\big(\mathbf{C}_{\text{B}},\, R,\, \mathbf{n},\, \boldsymbol{\omega}_o,\, \mathbf{r}_{\text{local}},\, \mathbf{r}_{\text{global}},\, \mathbf{v}_{\text{global}},\, \boldsymbol{\omega}_{\text{sun}},\, r_{\text{sun}},\, v_{\text{sun}}\big). \tag{4}$$

Here, Π^{pbr} denotes the PBR Gaussian rasterizer. Let the photometric, material, and visibility losses be denoted by $\mathcal{L}_{\text{photo}}$, $\mathcal{L}_{\text{tea}}$, and $\mathcal{L}_{\text{vis}}$ respectively; they are computed as shown in Eq. (5), and the range masking applied to $\mathcal{L}_{\text{vis}}$ is given in appendix A.2. Here $\mathcal{L}_{\text{msg}}$ is the Multi-Scale Gradient loss in (Li & Snavely, 2018) and V_{ray} is the ray tracer for the geometry-derived visibility constraint.

$$\begin{aligned}\mathcal{L}_{\text{photo}} &= \mathcal{L}_1 + \lambda_{\text{lpips}}\mathcal{L}_{\text{LPIPS}} + \lambda_{\text{ssim}}\mathcal{L}_{\text{SSIM}}, \quad \mathcal{L}_{\text{tea}} = \mathcal{L}_1 + \lambda_{\text{msg}}\mathcal{L}_{\text{msg}}, \\ \mathcal{L}_{\text{vis}} &= \mathbb{E}_{\boldsymbol{\omega}\sim\Omega,\, G\sim\mathcal{G}}\left[\|\mathbf{v}_{\text{global}}(\boldsymbol{\omega}, \mathbf{x}_G) - V_{\text{ray}}(G, \boldsymbol{\omega})\|\right] + \mathbb{E}_{G\sim\mathcal{G}}\left[\|v_{\text{sun}}(G) - V_{\text{ray}}(G, \boldsymbol{\omega}_{\text{sun}})\|\right].\end{aligned} \tag{5}$$

Table 1: **Quantitative comparison on nuScenes benchmark.** We mark whether a method explicitly models illumination disentanglement with ✅. The **best**, second-best, and third-best results in each column are highlighted in color. * denotes a metric taken from the corresponding paper.

Models	Light Disentangled	nuScenes					Efficiency	
		PSNR↑	LPIPS↓	SSIM↑	PCC↑	D-RMSE↓	Infer FPS↑	Infer Mem.(GB)↓
PixelSplat	❌	21.33	0.376	0.607	0.077	12.78	2.03	31.41
MVSplat	❌	21.87	0.342	0.621	0.201	12.13	4.93	16.10
DrivingRecon	❌	21.09	0.411	0.639	0.633	7.82	1.83	5.65
EVolSplat	❌	22.15	0.348	0.686	0.818	6.66	6.40	5.69
DepthSplat	❌	23.19	0.339	0.675	0.431	10.04	**7.19**	7.44
DrivingForward	❌	23.84	0.280	0.739	0.437	15.40	3.94	**5.45**
STORM	❌	24.04	0.261	0.719	0.829	5.95	2.61	18.46
Omni-Scene	❌	24.11	0.242	0.734	0.816	5.80	2.51	8.34
UniSplat	❌	25.37*	0.246*	**0.765***	-	-	1.50	13.07
Lumen-GS (Ours)	✅	**25.51**	**0.221**	0.754	**0.888**	**5.41**	1.79	9.15

The final objective function is given in Eq. (6). Here, $\mathbf{O}$, $\mathbf{D}$, $\mathbf{I}_n$, $\mathbf{I}_\mathrm{B}$, and $\mathbf{I}_\mathrm{R}$ are teacher pseudo-labels (Lai et al., 2024; Wang et al., 2026; Wu et al., 2025; Sun et al., 2025); appendix A.3 states which teacher supplies each map. The two terms of the extra regularizer $\mathcal{L}_\mathrm{reg}$, together with the range masking of $\mathcal{L}_\mathrm{vis}$, are given in appendix A.2, where all loss weights are also listed.

$$\begin{aligned}\mathcal{L} = {} & \mathcal{L}_\mathrm{photo}(\hat{\mathbf{I}}, \mathbf{I}) + \lambda_\mathrm{pbr}\mathcal{L}_\mathrm{photo}(\hat{\mathbf{I}}_\mathrm{pbr}, \mathbf{I}) + \lambda_\mathrm{depth}\mathcal{L}_1(\hat{\mathbf{D}}, \mathbf{D}) + \lambda_\mathrm{mask}\mathcal{L}_1(\hat{\mathbf{O}}, \mathbf{O}) \\ & + \lambda_\mathrm{tea}\mathcal{L}_\mathrm{tea}(\hat{\mathbf{I}}_n, \mathbf{I}_n) + \lambda_\mathrm{tea}\mathcal{L}_\mathrm{tea}(\hat{\mathbf{I}}_\mathrm{B}, \mathbf{I}_\mathrm{B}) + \lambda_\mathrm{tea}\mathcal{L}_\mathrm{tea}(\hat{\mathbf{I}}_\mathrm{R}, \mathbf{I}_\mathrm{R}) + \lambda_\mathrm{vis}\mathcal{L}_\mathrm{vis}(\mathcal{G}, \boldsymbol{\omega}_\mathrm{sun}) + \lambda_\mathrm{reg}\mathcal{L}_\mathrm{reg}.\end{aligned} \tag{6}$$

4 Experiments

4.1 Basic Settings

Baselines. We benchmark against recent state-of-the-art models in feed-forward reconstruction (Szymanowicz et al., 2024; Yu et al., 2024; Charatan et al., 2024; Chen et al., 2024; Lu et al., 2024; Miao et al., 2025; Xu et al., 2025; Tian et al., 2025; Gieruc et al., 2024; Yang et al., 2025; Wei et al., 2025; Shi et al., 2026), scene relighting (Chen et al., 2025b; Lin et al., 2025a), object relighting (Jiang et al., 2024; Shi et al., 2025; Li et al., 2025; Kaleta et al., 2025) and harmonization (Ke et al., 2022; Guerreiro et al., 2023; Zhang et al., 2025b; Brooks et al., 2023; Zhou et al., 2024).

Metrics. We evaluate rendering quality using PSNR, LPIPS, and SSIM. Albedo and roughness accuracy $\mathbf{C}_\mathrm{B}$-RMSE and R-RMSE are measured by RMSE against teacher-predicted references that are shared by all methods but are not real ground truth (appendix A.3), while normals use mean angular error $\mathbf{n}$-Mean and depth uses PCC (Pearson correlation coefficient) between depth maps (Wei et al., 2025; Xiong et al., 2023), and D-RMSE to calculate error between rendered and LiDAR depth. Efficiency is assessed by inference time and memory. Harmonization quality is measured with FID, KID, and CLIP-score, and detection performance with precision, recall, and mAP.

Implementation details. We train on 8 A800 80GB GPUs for approximately 6 days at batch size 8, using AdamW with a cosine-annealed learning rate for 100K iterations; feature-aggregation hyperparameters follow (Yu et al., 2025), and appendix A.2 and appendix A.6 list all remaining values. All teacher pseudo-labels are precomputed offline.

4.2 Main Results

We evaluate Lumen-GS from four aspects: feed-forward sparse-view reconstruction, PBR relighting, object insertion, and downstream night-time and snow weather data augmentation. The results demonstrate both strong rendering quality and practical utility for controllable simulation.

Feed-forward sparse-view reconstruction. Tab. 1 compares Lumen-GS with representative feed-forward reconstruction methods on nuScenes (Caesar et al., 2020). Despite additionally predicting intrinsic properties and illumination parameters, Lumen-GS remains competitive with the strongest radiance-field baseline, UniSplat (Shi et al., 2026). Compared with UniSplat, Lumen-GS improves PSNR by +0.14 dB and reduces LPIPS by 0.025, while also achieving the highest depth correlation among all methods with a +0.059 PCC gain over the second-best result. Although highly optimized real-time models such as DepthSplat (Xu et al., 2025) and EVolSplat (Miao et al., 2025) achieve

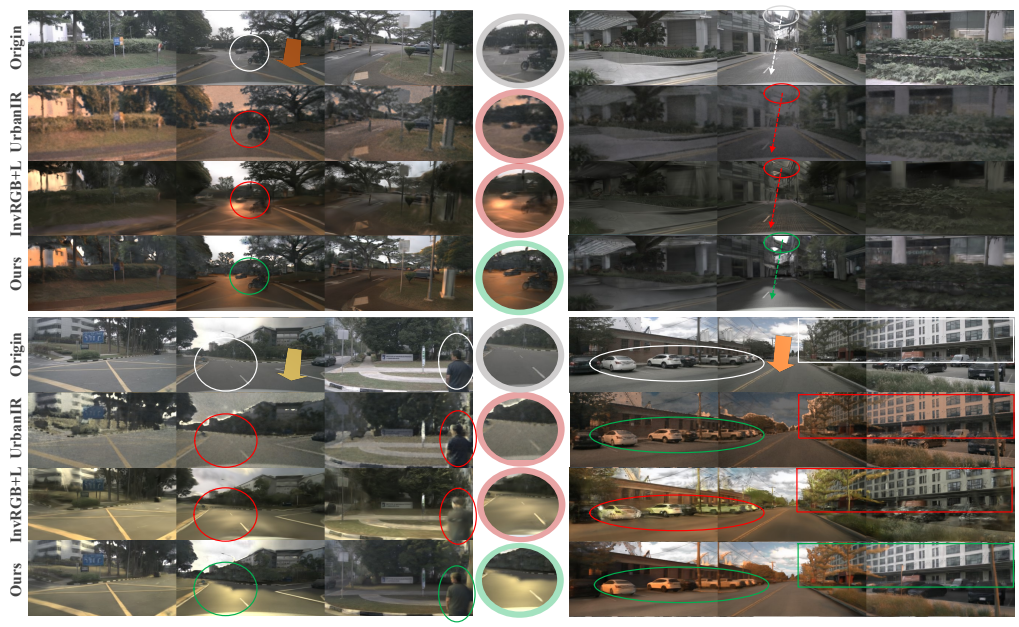

Figure 3: **Qualitative comparison of scene relighting.** UrbanIR and InvRGB+L are optimized per scene on the full scene bin (*dense*), with the same GPS/IMU sun prior as ours; Lumen-GS infers from one timestamp. Our relighting shows smoother illumination transitions and more plausible lighting.

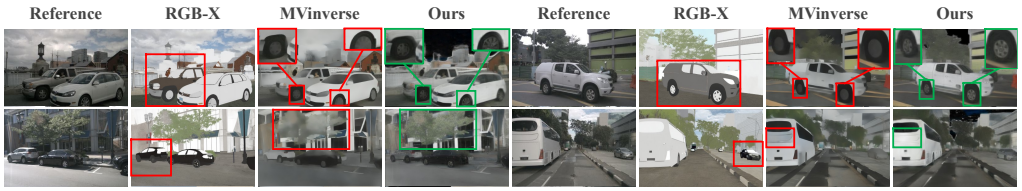

Figure 4: **Albedo prediction results.** Our model avoids color inconsistency and over-smoothing.

higher FPS, they are substantially weaker in reconstruction quality and do not provide disentangled lighting or material outputs (absolute speed and memory are reported per method in Tab. 1). As Fig. 15 shows, our model demonstrates superior reconstruction quality. Lumen-GS is therefore the only method in this comparison that simultaneously supports sparse-view feed-forward reconstruction and explicit light manipulation, which is essential for relighting and simulation-oriented applications.

Table 2: **Quantitative comparison of scene relighting results.** Dense baselines are reported as upper-bound references. Rankings are computed only among the sparse-input settings. $\mathbf{n}$-Mean, $\mathbf{C}_B$-RMSE, and R-RMSE denote normal mean angular error, albedo RMSE, and roughness RMSE.

Model	Setting	RGB branch			PBR branch						PCC↑	D-RMSE↓	Time↓
		PSNR↑	LPIPS↓	SSIM↑	PSNR↑	LPIPS↓	SSIM↑	$\mathbf{n}$-Mean↓	$\mathbf{C}_B$-RMSE↓	R-RMSE↓			
UrbanIR	D	21.53	0.373	0.791	14.19	0.432	0.680	12.49	0.256	–	0.268	10.78	∼5h
InvRGB+L	D	26.86	0.218	0.834	21.96	0.332	0.697	15.15	0.068	0.070	0.717	6.45	∼7h
UrbanIR	S	13.04	0.676	0.375	12.93	0.677	0.352	34.91	0.156	–	0.140	15.67	∼5h
InvRGB+L	S	23.31	0.249	0.721	20.42	0.342	0.636	15.58	**0.085**	0.082	0.530	11.23	∼7h
MVInv.+DA3	S	17.73	0.493	0.471	12.67	0.612	0.389	28.77	0.144	0.128	0.606	9.22	2s
Ours	S	**25.51**	**0.221**	**0.754**	**22.55**	**0.269**	**0.715**	**13.50**	0.088	**0.076**	**0.888**	**5.41**	**0.56s**

PBR relighting and intrinsic rendering. We compare Lumen-GS with outdoor street-scene relighting methods under two settings in Tab. 2. In the *dense* setting (D) each baseline is optimized per scene on all frames of the scene bin. In the *sparse* setting (S) it is optimized or run in inference on the same single-timestamp surround-view input available to Lumen-GS. For fairness, all baselines are additionally provided with the same GPS/IMU-derived ground-truth sun direction instead of estimating it during optimization. As Fig. 12 shows, these iterative baselines degrade steadily once observations are removed. We also combine a strong material-estimation model, MVInverse (Wu et al., 2025), with the feed-forward reconstruction model DA3 (Lin et al., 2025b); this composition is rendered with the illumination predicted by our model, so that the comparison isolates geometry and material

from lighting (protocol in appendix A.3). As Tab. 2 illustrates, Lumen-GS surpasses both dense baselines on all three PBR-rendering metrics (PBR-PSNR 22.55 vs 21.96 for the strongest bound, InvRGB+L-dense) while trailing that bound on RGB fidelity and intrinsic errors—the expected price of single-timestamp input and a 0.56s forward pass. Under the sparse setting, it delivers the strongest RGB and PBR rendering quality. Qualitatively, our relighting is also more spatially coherent and physically plausible: Fig. 3 shows smoother illumination transitions, cast light that follows scene geometry, and fewer specular artifacts; Fig. 4 compares the predicted albedo maps.

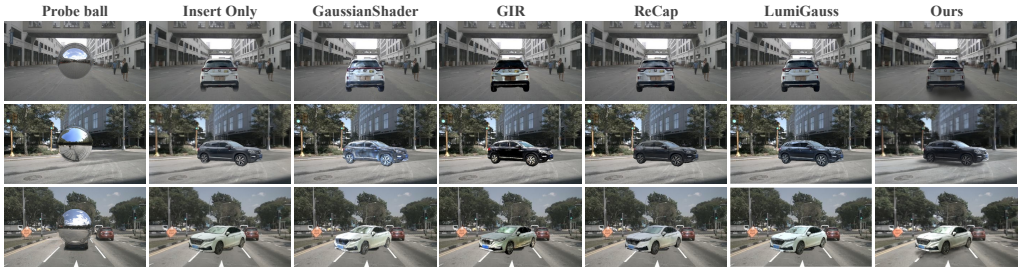

Figure 5: **Qualitative comparison of object insertion.** Our feed-forward intrinsic scene representation enables inserted assets to be rendered with lighting that is consistent with the target scene.

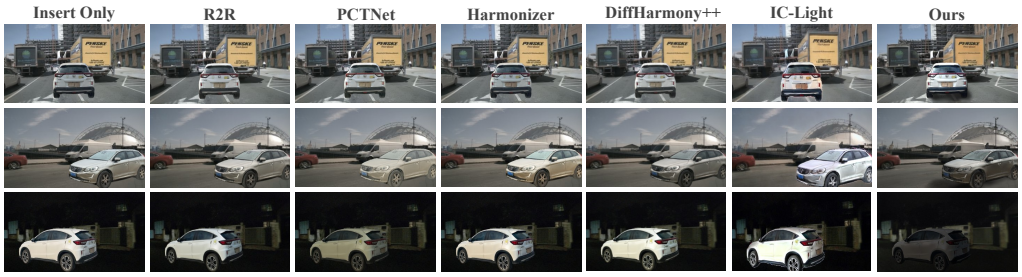

Figure 6: **Comparison with 2D harmonization baselines.** Direct image harmonization often struggles to preserve physically plausible shadows and highlights in open-world street scenes, whereas our 3D PBR pipeline yields more coherent composites.

Table 3: Quantitative comparison of different harmonization models; the IC-light result follows Zhang et al. (2025a).

Model	FID ↓	KID ↓	CLIP Score ↑
Insert	124.22	62.31	21.48
Harmonizer	116.17	50.83	21.51
PCTNet	114.92	50.72	21.45
InstructPix2Pix	115.87	63.77	21.44
DiffHarmony++	119.63	55.63	21.03
R2R	118.13	56.77	21.17
IC-light	121.32	59.97	21.54
Lumen-GS (Ours)	**108.63**	**47.51**	**22.16**

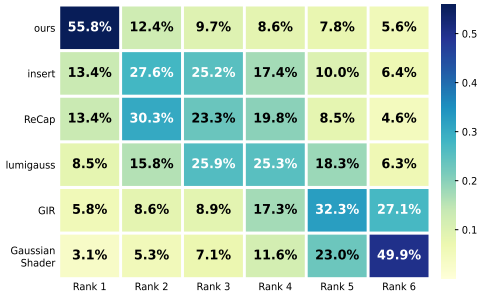

Figure 7: User study results for relighting-based object insertion quality.

Object insertion with relighting. The light-disentangled representation also enables sparse-view object insertion. Existing pipelines typically estimate an environment map from the input image and use it to relight a pasted asset. For comparison, we bake vehicles from the 3DRealCar (Du et al., 2025) dataset into the representations of GIR (Shi et al., 2025), GaussianShader (Jiang et al., 2024), ReCap (Li et al., 2025), and LumiGauss (Kaleta et al., 2025). We use the pretrained DiffusionLight-Turbo (Chinchuthakun et al., 2026) to predict a single-frame environment probe for relighting. Since these baselines cannot recover a complete 3D scene from our sparse input, the rendered vehicles are composited by alpha blending. For Lumen-GS, we adapt an iterative baking procedure to align external assets with our lighting representation and insert the vehicles directly into the reconstructed scene. As shown in Fig. 5, our pipeline yields more realistic object-scene integration, especially in highlights, shadows, and local illumination. An anonymized user study with 96 participants over 40 scenes (Fig. 7; protocol and statistics in appendix A.4) ranks the six insertions of each scene by

perceived integration: over the 3793 usable rankings, Lumen-GS is ranked first in 55.8%, against 13.4% for the best baseline, with a mean rank of 2.17 against 2.93.

Object insertion with 2D harmonization. We also compare against 2D harmonization baselines (Ke et al., 2022; Guerreiro et al., 2023; Zhang et al., 2025b; Brooks et al., 2023; Zhou et al., 2024), which refine a copy-paste composite with image-to-image networks. Fig. 6 shows that most of them struggle with open-world street scenes: they adjust color tone but rarely recover physically consistent shadows, reflections, and directional lighting. For quantitative evaluation, we randomly place several pre-baked 3DRealCar assets into nuScenes scenes under drivable-area (SegFormer (Xie et al., 2021)) and ground-plane constraints, yielding 1,200 rendered composites. We then sample 1,200 non-overlapping real nuScenes images and compute FID, KID, and CLIP Score. As reported in Tab. 3, Lumen-GS achieves the best generation quality and semantic alignment, improving over the strongest baseline by 6.29 in FID, 3.21 in KID, and 0.62 in CLIP Score. Each harmonization baseline is introduced individually in appendix A.1.

Table 4: Performance of YOLO variants w/ or w/o night car labels generated by our model.

Settings	YOLOv5n				YOLOv8n				YOLOv12n				YOLO-Master			
	P	R	mAP_{50}	$mAP_{50:95}$	P	R	mAP_{50}	$mAP_{50:95}$	P	R	mAP_{50}	$mAP_{50:95}$	P	R	mAP_{50}	$mAP_{50:95}$
w/o Gen	0.509	0.350	0.373	0.204	0.659	0.330	0.377	0.205	0.547	0.343	0.383	0.202	0.629	0.362	0.384	0.210
w/ Gen	**0.614**	**0.419**	**0.466**	**0.261**	**0.725**	**0.375**	**0.459**	**0.267**	**0.577**	**0.409**	**0.437**	**0.253**	**0.653**	**0.402**	**0.472**	**0.265**

Night-time and adverse weather detection augmentation. Unlike WeatherEdit (Qian et al., 2026), which models snow as appearance-entangled particles, we endow snow with intrinsic properties that interact with the illumination parameters predicted by Lumen-GS, leading to more realistic results. Fig. 17 in the appendix shows additional snow weather simulation results. We further test whether Lumen-GS benefits real-time 2D perception. Four vehicle detectors are trained on $1{,}500$ daytime and only 100 nighttime real nuImages frames so that night remains the rare class, and are then augmented with a further $1{,}500$ night images obtained by running Lumen-GS on each of those daytime frames and switching the reconstructed scene to night, the annotations being inherited from the daytime frame (Fig. 14 in the appendix); both conditions are scored on the same 192 *real* nighttime nuImages validation frames, so no rendered image enters the test set (protocol in appendix A.3). Tab. 4 shows that the generated data consistently improves all detectors. The gains are especially clear on mAP: mAP@50 improves by 0.054–0.093, while mAP@50:95 improves by 0.051–0.062 across the four YOLO variants. These improvements require no manual labeling, further supporting the practical value of Lumen-GS for controllable data augmentation in autonomous-driving perception.

Ablation of key designs of Lumen-GS. Removing any PAD decoder degrades both branches (PSNR -0.34 dB for w/o Pixel PAD, -0.36 dB for w/o Volume PAD, -0.55 dB when both are removed); the PAD effect is milder than removing the feature branches themselves (w/o pixel branch -2.67 dB, w/o volume branch -0.95 dB), which carry the larger share of rendering fidelity. Removing sun modeling degrades mainly the PBR branch, underscoring the necessity of directional-light modeling. Tab. 9 reports further ablations.

Table 5: Ablation of the PAD decoders, feature branches and sun modeling.

Model	PSNR↑	LPIPS↓	SSIM↑	PBR-PSNR↑	PBR-LPIPS↓	PBR-SSIM↑	PCC↑	D-RMSE↓	C_B-RMSE↓
w/o Pixel PAD	25.17	0.225	0.748	22.12	0.278	0.706	0.882	5.53	0.090
w/o Volume PAD	25.15	0.225	0.749	22.44	0.272	0.708	0.883	5.53	0.089
w/o All PAD	24.96	0.232	0.741	22.07	0.281	0.703	0.882	5.54	0.091
w/o Pixel branch	22.84	0.358	0.652	19.85	0.402	0.621	0.741	6.75	0.134
w/o Volume branch	24.56	0.264	0.728	21.68	0.294	0.692	0.865	6.01	0.102
w/o sun modeling	25.16	0.228	0.744	20.73	0.367	0.674	0.884	5.41	0.167
Lumen-GS	**25.51**	**0.221**	**0.754**	**22.55**	**0.269**	**0.715**	**0.888**	**5.41**	**0.088**

5 Conclusion

In this paper, we presented Lumen-GS, a feed-forward model for sparse-view street-scene reconstruction with explicit intrinsic decomposition and light disentanglement. Predicting geometry, intrinsics, and illumination in one forward pass enables physically based scene manipulation on top of efficient reconstruction. Experiments on nuScenes show that Lumen-GS remains competitive with the strongest feed-forward baselines in novel-view synthesis and with iterative inverse-rendering methods in PBR rendering quality, while supporting controllable object insertion, harmonization, and perception-oriented data augmentation toward lighting-controllable driving simulation.

AI USE STATEMENT

We used generative AI tools for language polishing of the manuscript and for assistance with routine code refactoring and LaTeX formatting. Off-the-shelf generative models (Gemini 3 Pro Image, DiT360, SD-Turbo) were used only as external input generators for the zero-shot evaluation; they are cited experimental material, not AI assistance in our research or writing workflow. All AI-assisted output was manually reviewed and verified by the authors, who take full responsibility for the final content of this work, including text, claims, and artifacts produced with the aid of generative AI.

ETHICS STATEMENT

This work targets street-scene reconstruction and simulation for autonomous driving, using the publicly available nuScenes dataset, which has been anonymized by its authors. The only human-subject data we collect are the anonymous preference rankings of the user study in appendix A.4, contributed voluntarily and without compensation by graduate students with a vision or graphics background; the instrument records only the ordering of six model renderings and contains no identifying, demographic or device data. Since realistic relit and inserted objects could in principle be misused to fabricate driving footage, we encourage watermarking and provenance tracking of synthesized data, and careful validation before any safety-critical use (see Appendix A.9).

REPRODUCIBILITY STATEMENT

We document the model architecture, training settings, all loss weights, and the two auxiliary loss terms of Eq. (6) in Appendix A.2, the dataset preprocessing and evaluation protocols in Appendix A.3, the user-study protocol, instructions and statistical analysis in Appendix A.4, and the custom CUDA implementation of the differentiable PBR rasterizer in Appendix A.6. Additional quantitative results, ablations, and visualizations are provided in the appendix. The source code, trained models, and evaluation scripts will be made publicly available. Infer FPS and memory in Tab. 1 are measured by us for every method on the same GPU and at the same input resolution. An anonymized project page (`https://anonymous.4open.science/w/lumen-gs-projectpage-6EFD/`) hosts the rendered video and further qualitative results; this link will be replaced by the public one on acceptance.

REFERENCES

Tim Brooks, Aleksander Holynski, and Alexei A Efros. Instructpix2pix: Learning to follow image editing instructions. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 18392–18402, 2023.

Holger Caesar, Varun Bankiti, Alex H Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. nuscenes: A multimodal dataset for autonomous driving. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 11621–11631, 2020.

Hezhi Cao, Panhao Cheng, Huangsheng Du, Qibiao Li, Youcheng Cai, and Ligang Liu. Lightbridge: Feed-forward generative relighting for 3d gaussian splatting. *arXiv preprint arXiv:2609.02543*, 2026.

David Charatan, Sizhe Lester 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.

Hongze Chen, Zehong Lin, and Jun Zhang. Gi-gs: Global illumination decomposition on gaussian splatting for inverse rendering. In *Int. Conf. Learn. Represent.*, 2025a.

Xiaoxue Chen, Bhargav Chandaka, Chih-Hao Lin, Ya-Qin Zhang, David Forsyth, Hao Zhao, and Shenlong Wang. Invrgb+ l: Inverse rendering of complex scenes with unified color and lidar reflectance modeling. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 27176–27186, 2025b.

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*, pp. 370–386. Springer, 2024.

Yutong Chen, Marko Mihajlovic, Xiyi Chen, Yiming Wang, Sergey Prokudin, and Siyu Tang. Splatformer: Point transformer for robust 3d gaussian splatting. In *The Thirteenth International Conference on Learning Representations*, 2025c. URL https://openreview.net/forum?id=9NfHbWKqMF.

Worameth Chinchuthakun, Pakkapon Phongthawee, Amit Raj, Varun Jampani, Pramook Khungurn, and Supasorn Suwajanakorn. Diffusionlight-turbo: Accelerated light probes for free via single-pass chrome ball inpainting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2026.

Xiaobiao Du, Yida Wang, Haiyang Sun, Zhuojie Wu, Hongwei Sheng, Shuyun Wang, Jiaying Ying, Ming Lu, Tianqing Zhu, Kun Zhan, et al. 3drealcar: An in-the-wild rgb-d car dataset with 360-degree views. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 26488–26498, 2025.

Xin Fei, Wenzhao Zheng, Yueqi Duan, Wei Zhan, Masayoshi Tomizuka, Kurt Keutzer, and Jiwen Lu. Pixelgaussian: Generalizable 3d gaussian reconstruction from arbitrary views. *arXiv preprint arXiv:2410.18979*, 2024.

Haoran Feng, Dizhe Zhang, Xiangtai Li, Bo Du, and Lu Qi. Dit360: High-fidelity panoramic image generation via hybrid training. *arXiv preprint arXiv:2510.11712*, 2025a.

Wei Feng, Kangrui Ye, Qi Zhang, Qian Zhang, and Nan Li. 2d gaussian splatting for outdoor scene decomposition and relighting. In *Proceedings of the Thirty-Fourth International Joint Conference on Artificial Intelligence, IJCAI-25*, pp. 990–998, 2025b.

Jian Gao, Chun Gu, Youtian Lin, Zhihao Li, Hao Zhu, Xun Cao, Li Zhang, and Yao Yao. Relightable 3d gaussians: Realistic point cloud relighting with brdf decomposition and ray tracing. In *European Conference on Computer Vision*, pp. 73–89. Springer, 2024.

Théo Gieruc, Marius Kästingschäfer, Sebastian Bernhard, and Mathieu Salzmann. 6img-to-3d: Few-image large-scale outdoor driving scene reconstruction. *arXiv preprint arXiv:2404.12378*, 2024.

Google. Nano Banana Pro: Gemini 3 Pro Image model from Google DeepMind. https://blog.google/innovation-and-ai/products/nano-banana-pro/, 2025. Accessed: 2026-09-25.

Julian Jorge Andrade Guerreiro, Mitsuru Nakazawa, and Björn Stenger. Pct-net: Full resolution image harmonization using pixel-wise color transformations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5917–5926, 2023.

Quentin Herau, Tianshuo Xu, Depu Meng, Jiezhi Yang, Chensheng Peng, Spencer Sherk, Yihan Hu, and Wei Zhan. Spectralsplat: Appearance-disentangled feed-forward gaussian splatting for driving scenes. *arXiv preprint arXiv:2604.03462*, 2026.

Binbin Huang, Zehao Yu, Anpei Chen, Andreas Geiger, and Shenghua Gao. 2d gaussian splatting for geometrically accurate radiance fields. In *ACM SIGGRAPH 2024 conference papers*, pp. 1–11, 2024.

Yuanhui Huang, Wenzhao Zheng, Yunpeng Zhang, Jie Zhou, and Jiwen Lu. Tri-perspective view for vision-based 3d semantic occupancy prediction. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 9223–9232, 2023.

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.

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.

Glenn Jocher. Ultralytics yolov5: A state-of-the-art, real-time object detection system, 2020. URL https://github.com/ultralytics/yolov5.

Glenn Jocher, Ayush Chaurasia, and Jing Qiu. Ultralytics yolov8, 2023. URL https://github.com/ultralytics/ultralytics.

Joanna Kaleta, Kacper Kania, Tomasz Trzcinski, and Marek Kowalski. Lumigauss: relightable gaussian splatting in the wild. In *Proceedings of the Winter Conference on Applications of Computer Vision*, pp. 1–10, 2025.

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

Zhanghan Ke, Chunyi Sun, Lei Zhu, Ke Xu, and Rynson WH Lau. Harmonizer: Learning to perform white-box image and video harmonization. In *European conference on computer vision*, pp. 690–706. Springer, 2022.

Bernhard Kerbl, Georgios Kopanas, Thomas Leimkuehler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Trans. Graph.*, 42(4), July 2023. ISSN 0730-0301. doi: 10.1145/3592433. URL https://doi.org/10.1145/3592433.

Xin Lai, Zhuotao Tian, Yukang Chen, Yanwei Li, Yuhui Yuan, Shu Liu, and Jiaya Jia. Lisa: Reasoning segmentation via large language model. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9579–9589, 2024.

Jingzhi Li, Zongwei Wu, Eduard Zamfir, and Radu Timofte. Recap: Better gaussian relighting with cross-environment captures. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 21307–21316, 2025.

Zhengqi Li and Noah Snavely. Megadepth: Learning single-view depth prediction from internet photos. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2041–2050, 2018.

Ruofan Liang, Zan Gojcic, Huan Ling, Jacob Munkberg, Jon Hasselgren, Chih-Hao Lin, Jun Gao, Alexander Keller, Nandita Vijaykumar, Sanja Fidler, et al. Diffusion renderer: Neural inverse and forward rendering with video diffusion models. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 26069–26080, 2025.

Chih-Hao Lin, Bohan Liu, Yi-Ting Chen, Kuan-Sheng Chen, David Forsyth, Jia-Bin Huang, Anand Bhattad, and Shenlong Wang. Urbanir: Large-scale urban scene inverse rendering from a single video. In *2025 International Conference on 3D Vision (3DV)*, pp. 512–523. IEEE, 2025a.

Haotong Lin, Sili Chen, Junhao Liew, Donny Y Chen, Zhenyu Li, Guang Shi, Jiashi Feng, and Bingyi Kang. Depth anything 3: Recovering the visual space from any views. *arXiv preprint arXiv:2511.10647*, 2025b.

Xu Lin, Jinlong Peng, Zhenye Gan, Jiawen Zhu, and Jun Liu. Yolo-master: Moe-accelerated with specialized transformers for enhanced real-time detection. *arXiv preprint arXiv:2512.23273*, 2025c.

Yifan Liu, Keyu Fan, Weihao Yu, Chenxin Li, Hao Lu, and Yixuan Yuan. Monosplat: Generalizable 3d gaussian splatting from monocular depth foundation models. In *Proceedings of the Computer Vision and Pattern Recognition Conference (CVPR)*, pp. 21570–21579, June 2025.

Hao Lu, Tianshuo Xu, Wenzhao Zheng, Yunpeng Zhang, Wei Zhan, Dalong Du, Masayoshi Tomizuka, Kurt Keutzer, and Yingcong Chen. Drivingrecon: Large 4d gaussian reconstruction model for autonomous driving. *arXiv preprint arXiv:2412.09043*, 2024.

Sheng Miao, Jiaxin Huang, Dongfeng Bai, Xu Yan, Hongyu Zhou, Yue Wang, Bingbing Liu, Andreas Geiger, and Yiyi Liao. Evolsplat: Efficient volume-based gaussian splatting for urban view synthesis. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 11286–11296, 2025.

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

Zhiyuan Min, Yawei Luo, Jianwen Sun, and Yi Yang. Epipolar-free 3d gaussian splatting for generalizable novel view synthesis. *arXiv preprint arXiv:2410.22817*, 2024.

Patrick Noras, Jun Myeong Choi, Didier Stricker, Pieter Peers, and Roni Sengupta. GAINS: Gaussian-based inverse rendering from sparse multi-view captures. In *European Conference on Computer Vision (ECCV)*, 2026.

Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy V. Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel HAZIZA, Francisco Massa, Alaaeldin El-Nouby, Mido Assran, Nicolas Ballas, Wojciech Galuba, Russell Howes, Po-Yao Huang, Shang-Wen Li, Ishan Misra, Michael Rabbat, Vasu Sharma, Gabriel Synnaeve, Hu Xu, Herve Jegou, Julien Mairal, Patrick Labatut, Armand Joulin, and Piotr Bojanowski. DINOv2: Learning robust visual features without supervision. *Transactions on Machine Learning Research*, 2024. ISSN 2835-8856. URL https://openreview.net/forum?id=a68SUt6zFt. Featured Certification.

Chenghao Qian, Wenjing Li, Yuhu Guo, and Gustav Markkula. Weatheredit: Controllable weather editing with 4d gaussian field. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 8511–8519, 2026.

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.

Axel Sauer, Dominik Lorenz, Andreas Blattmann, and Robin Rombach. Adversarial diffusion distillation. In *European Conference on Computer Vision*, pp. 87–103. Springer, 2024.

Chen Shi, Shaoshuai Shi, Xiaoyang Lyu, Chunyang Liu, Kehua Sheng, Bo Zhang, and Li Jiang. Unisplat: Unified spatio-temporal fusion via 3d latent scaffolds for dynamic driving scene reconstruction. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=Ng2VDbKD4r.

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*, 2025.

Shanlin Sun, Yifan Wang, Hanwen Zhang, Yifeng Xiong, Qin Ren, Ruogu Fang, Xiaohui Xie, and Chenyu You. Ouroboros: Single-step diffusion models for cycle-consistent forward and inverse rendering. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 10386–10397, 2025.

Stanislaw Szymanowicz, Chrisitian Rupprecht, and Andrea Vedaldi. Splatter image: Ultra-fast single-view 3d reconstruction. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 10208–10217, 2024.

Qijian Tian, Xin Tan, Yuan Xie, and Lizhuang Ma. Drivingforward: Feed-forward 3d gaussian splatting for driving scene reconstruction from flexible surround-view input. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 7374–7382, 2025.

Yunjie Tian, Qixiang Ye, and David Doermann. YOLOv12: Attention-centric real-time object detectors. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2026. URL https://openreview.net/forum?id=gCvByDI4FN.

Bruce Walter, Steve Marschner, Hongsong Li, and Kenneth E. Torrance. Microfacet models for refraction through rough surfaces. In *Rendering Techniques*, 2007. URL https://api.semanticscholar.org/CorpusID:8061072.

Ruicheng Wang, Sicheng Xu, Yue Dong, Yu Deng, Jianfeng Xiang, Zelong Lv, Guangzhong Sun, Xin Tong, and Jiaolong Yang. Moge-2: Accurate monocular geometry with metric scale and sharp details. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2026. URL https://openreview.net/forum?id=16mDq7m2OK.

Yunsong Wang, Tianxin Huang, Hanlin Chen, and Gim Hee Lee. Freesplat: Generalizable 3d gaussian splatting towards free view synthesis of indoor scenes. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (eds.), *Advances in Neural Information Processing Systems*, volume 37, pp. 107326–107349. Curran Associates, Inc., 2024. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/c2166d01fe4bcd694aba89f608737678-Paper-Conference.pdf.

Yunsong Wang, Tianxin Huang, Hanlin Chen, and Gim Hee Lee. Freesplat++: Generalizable 3d gaussian splatting for efficient indoor scene reconstruction. *arXiv preprint arXiv:2503.22986*, 2025.

Dongxu Wei, Zhiqi Li, and Peidong Liu. Omni-scene: Omni-gaussian representation for ego-centric sparse-view scene reconstruction. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 22317–22327, 2025.

Haoqian Wu, Zhipeng Hu, Lincheng Li, Yongqiang Zhang, Changjie Fan, and Xin Yu. Nefii: Inverse rendering for reflectance decomposition with near-field indirect illumination. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4295–4304, 2023.

Xiangzuo Wu, Chengwei Ren, Jun Zhou, Xiu Li, and Yuan Liu. Mvinverse: Feed-forward multi-view inverse rendering in seconds. *arXiv preprint arXiv:2512.21003*, 2025.

Enze Xie, Wenhai Wang, Zhiding Yu, Anima Anandkumar, Jose M Alvarez, and Ping Luo. Segformer: Simple and efficient design for semantic segmentation with transformers. *Advances in neural information processing systems*, 34:12077–12090, 2021.

Haolin Xiong, Sairisheek Muttukuru, Rishi Upadhyay, Pradyumna Chari, and Achuta Kadambi. Sparsegs: Real-time 360 {\deg} sparse view synthesis using gaussian splatting. *arXiv preprint arXiv:2312.00206*, 2023.

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 Computer Vision and Pattern Recognition Conference*, pp. 16453–16463, 2025.

Jiawei Yang, Jiahui Huang, Boris Ivanovic, Yuxiao Chen, Yan Wang, Boyi Li, Yurong You, Apoorva Sharma, Maximilian Igl, Peter Karkus, Danfei Xu, Yue Wang, and Marco Pavone. STORM: Spatio-temporal reconstruction model for large-scale outdoor scenes. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=M2NFWRPMUd.

Yao Yao, Jingyang Zhang, Jingbo Liu, Yihang Qu, Tian Fang, David McKinnon, Yanghai Tsin, and Long Quan. Neilf: Neural incident light field for physically-based material estimation. In *European conference on computer vision*, pp. 700–716. Springer, 2022.

Haochen Yu, Qiankun Liu, Hongyuan Liu, Jianfei Jiang, Juntao Lyu, Jiansheng Chen, and Huimin Ma. Xyzcylinder: Towards compatible feed-forward 3d gaussian splatting for driving scenes via unified cylinder lifting method. *arXiv preprint arXiv:2510.07856*, 2025.

Wangbo Yu, Jinbo Xing, Li Yuan, Wenbo Hu, Xiaoyu Li, Zhipeng Huang, Xiangjun Gao, Tien-Tsin Wong, Ying Shan, and Yonghong Tian. Viewcrafter: Taming video diffusion models for high-fidelity novel view synthesis. *arXiv preprint arXiv:2409.02048*, 2024.

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

Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. Scaling in-the-wild training for diffusion-based illumination harmonization and editing by imposing consistent light transport. In *The Thirteenth International Conference on Learning Representations*, 2025a. URL https://openreview.net/forum?id=u1cQYxRI1H.

Youjia Zhang, Teng Xu, Junqing Yu, Yuteng Ye, Yanqing Jing, Junle Wang, Jingyi Yu, and Wei Yang. Nemf: Inverse volume rendering with neural microflake field. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 22919–22929, 2023b.

Zhiqiu Zhang, Dongqi Fan, Mingjie Wang, Qiang Tang, Jian Yang, and Zili Yi. Region-to-region: Enhancing generative image harmonization with adaptive regional injection. *arXiv preprint arXiv:2508.09746*, 2025b.

Pengfei Zhou, Fangxiang Feng, Guang Liu, Ruifan Li, and Xiaojie Wang. Diffharmony++: Enhancing image harmonization with harmony-vae and inverse harmonization model. In *Proceedings of the 32nd ACM International Conference on Multimedia*, pp. 10592–10601, 2024.

A TECHNICAL APPENDICES AND SUPPLEMENTARY MATERIAL

A.1 EXTENDED RELATED WORK

Image harmonization baselines. The five harmonization methods we compare against in Tab. 3 all take a composited cut-and-paste image as input and return a recolored image; none of them reconstructs the scene or its illumination, which is the structural difference from our pipeline. They differ mainly in how the correction is parameterized and how much prior the generator is allowed to carry. *Harmonizer* (Ke et al., 2022) casts harmonization as a white-box regression problem: it is trained on paired composites, takes the information available at composition time as an extra input, and learns to paint the foreground so that it matches the background, for both images and video. *PCTNet* (Guerreiro et al., 2023) instead predicts a per-pixel color transform and applies it to the foreground, which keeps the correction low-dimensional and lets the network run at full resolution. *Region-to-Region* (Zhang et al., 2025b) moves to a generative formulation and conditions the generator on regional cues injected adaptively, so that different parts of the composite receive different corrections. *InstructPix2Pix* (Brooks et al., 2023) is not a harmonization model but a general text-instructed diffusion editor; we include it to test how far an editor trained for open-ended instructions goes when the instruction is to make a composite consistent. *DiffHarmony++* (Zhou et al., 2024) is the diffusion-based end of the spectrum, pairing a harmony-VAE with an inverse-harmonization model so the generator is supervised in a latent space tied to the harmonization task. Across all five, the correction is a function of pixels: the network may darken a region to match its surroundings, but no shadow direction, reflection geometry or light transport is ever represented, which is why they adjust color tone more reliably than they recover contact shadows and directional highlights (Fig. 6).

Feed-forward relighting and appearance disentanglement. Two recent works drop the per-scene stage for reasons close to ours; what separates them is what they leave behind. SpectralSplat (Herau et al., 2026) disentangles appearance from geometry inside a feed-forward Gaussian model for driving scenes: a shared MLP factors color into an appearance-agnostic base stream and an appearance-conditioned adapted stream, both driven by a global appearance embedding extracted from DINOv2 features, and the factorization is trained on paired observations produced by a hybrid relighting pipeline that combines physics-based intrinsic decomposition with diffusion refinement — the same concern that leads us to let teachers enter Eq. (6) as priors rather than as targets. An appearance-adaptable temporal history stores appearance-agnostic features, so Gaussians accumulated across frames can be re-rendered under a chosen appearance. What is disentangled, however, is a scene-level appearance code rather than a material: no BRDF or roughness output is reported and no solar direction is estimated, so an appearance can be swapped but the sun cannot be moved. LightBridge (Cao et al., 2026) starts from an already reconstructed 3DGS asset and relights it in a single pass: relighting is cast as source-to-target transport in latent space, which replaces iterative diffusion sampling with a one-step prediction of 2D cues, and those cues are spread over the asset by sparse image-to-point self-attention followed by point-to-image cross-attention rather than full attention over all tokens. It needs no per-scene optimization, but it neither decomposes the scene nor predicts illumination parameters: the output is a plausible appearance transfer conditioned on a reference illumination, not a render whose sunlight, sky term and material can be set independently. Both are also trained from paired relighting supervision generated by a synthesis pipeline, while our teachers only bias the objective and the references we score against are defined in appendix A.3. Their evaluation target differs from ours as well — relighting and appearance transfer judged on image similarity, against the decomposition accuracy of Tab. 2 — which is why we place them here rather than in that table.

A.2 IMPLEMENTATION DETAILS

Our model takes 6 multi-view images as input, each uniformly resized to a resolution of 224×400. For feature aggregation, we employ DINOv2 (with registers small) (Oquab et al., 2024) as the image backbone to extract base features with a dimension of $D = 384$, and utilize MoGe-2 (normal-vitl) (Wang et al., 2026) as the metric depth estimator. The multi-view features are projected and aggregated onto a Hybrid Ring plane with a spatial size of 56×512. The volume-ring feature dimension D_v and the pixel-ring feature dimension D_p are set to 576 and 128, respectively.

Table 6: Implementation hyperparameters.

(a) Network Architecture		
Input	# input views n	6
	image resolution $H \times W$	224×400
Feature Aggregation	Image backbone	DINOv2 w/ registers small (Oquab et al., 2024)
	Depth estimator	MoGe-2 normal-vitl (Wang et al., 2026)
	Feature dimension D	384
	Image feature resolution $H_f \times W_f$	56×100
	Ring plane size $H_u \times W_u$	56×512
	Volume feature dim D_v	576
	Pixel feature dim D_p	128
PBR	Render representation	3DGS
	NDF / D term	Trowbridge-Reitz / GGX (Walter et al., 2007)
	Geometry / G term	Schlick-GGX (Walter et al., 2007)
	Fresnel / F term	Spherical Gaussian approximation (Walter et al., 2007)
Gaussian Decoder	Gaussian number of pixel branch N_p	458,752
	Gaussian number of volume branch N_v	Dynamic (depends on Occ branch in (Yu et al., 2025))
	Intermediate feature dimension d	64 for pixel; 288 for voxel
	Expert heads depth	4 for intermediate feature; 6 overall
(b) Hyperparameters		
Loss Weights	λ_{lpips}	0.2
	λ_{ssim}	0.2
	λ_{pbr}	0.1
	λ_{depth}	0.01
	λ_{mask}	0.1
	λ_{tea}	0.1
	λ_{msg}	0.01
	λ_{vis}	0.1
	λ_{reg}	1.0
	$\lambda_{\text{diffuse-smooth}}$	0.1
	λ_{local}	0.2
Training Details	Batch Size	8
	Optimizer	AdamW
	Initial / Min learning rate	2×10^{-3} / 2×10^{-4}
	Betas	$(0.0, 0.99)$
	Weight Decay	0.01
	LR Scheduler	Cosine Annealing
	Mixed Precision	bf16
	Warm-up Steps	1,000
	Training Steps	100,000

We build our physically-based rendering (PBR) pipeline upon the 3DGS representation. For the microfacet BRDF model, we use the Trowbridge-Reitz (GGX) distribution (Walter et al., 2007) for the normal distribution function (NDF), the Schlick-GGX formulation for the geometry term, and a Spherical Gaussian approximation for the Fresnel term. In the decoding stage, the pixel branch generates a fixed number of Gaussian primitives ($N_p = 458,752$), while the number of Gaussians generated by the volume branch (N_v) is dynamic, depending on the occupancy branch outputs following (Yu et al., 2025). The expert heads are constructed using MLPs with a depth of 4 for intermediate feature extraction and an overall depth of 6. The intermediate feature dimension d is set to 64 for the pixel branch and 288 for the volume branch.

We train our model on 8 A800 80GB GPUs with a batch size of 8 for approximately 6 days. The model is optimized using the AdamW optimizer with $\beta = (0.0, 0.99)$ and a weight decay of 0.01. We adopt a cosine annealing learning rate scheduler with a linear warm-up period of 1,000 steps. The initial learning rate is set to 2×10^{-3} and gradually decays to a minimum of 2×10^{-4}. The entire training process is conducted using bf16 mixed precision, spanning a total of 100,000 training steps. Tab. 6 lists all architecture and loss-weight settings.

For the final joint objective function, the base RGB rendering L1 loss weight is set to 1.0, with both λ_{lpips} and λ_{ssim} set to 0.2. The photometric loss weight for the PBR branch (λ_{pbr}) is set to 0.1. For additional physical constraints and geometric supervision, the depth loss weight λ_{depth} and the multi-scale gradient loss weight λ_{msg} are both set to 0.01. The mask loss λ_{mask}, teacher-guided material loss λ_{tea}, and visibility loss λ_{vis} are all equally weighted at 0.1.

The two paragraphs below complete the objective of Eq. (6) with the terms left out of the main text; their weights are listed in Tab. 6.

Extra Regularization Term. As formulated in Eq. (7), we render the color-independent diffuse map of the scene and enforce a spatial smoothness penalty using TV loss.

$$\begin{aligned} \mathbf{r}_{\text{diffuse}} &= \int_{\Omega} \mathbf{r}_{\text{env}}(\boldsymbol{\omega}_i, \mathbf{x})(\boldsymbol{\omega}_i \cdot \mathbf{n})_+ d\boldsymbol{\omega}_i, \\ \hat{\mathbf{I}}_{\text{diffuse}} &= \Pi^{\text{pbr}}(\mathbf{r}_{\text{diffuse}}), \\ \mathcal{L}_{\text{diffuse-smooth}} &= \text{TV}(\hat{\mathbf{I}}_{\text{diffuse}}). \end{aligned} \tag{7}$$

Regarding the localized emissive component $\mathbf{r}_{\textbf{local}}$, we explicitly impose an $\mathcal{L}_1$ regularization penalty to attenuate its overall contribution to the scene luminance, as formulated in Eq. (8). This regularization serves a dual purpose: first, secondary bounces and intrinsic self-emission are generally negligible in these specific environments; second, it rigorously prevents the dominant solar and ambient illumination from being erroneously absorbed into the emissive term. Consequently, this constraint significantly facilitates robust illumination disentanglement.

$$\mathcal{L}_{\text{local}} = \mathbb{E}_{G \sim \mathcal{G}} \left[\left\| \mathbf{r}^G_{\text{local}} \right\|_1 \right]. \tag{8}$$

The total regularization term is as Eq. (9) shows.

$$\mathcal{L}_{\text{reg}} = \lambda_{\text{local}} \mathcal{L}_{\text{local}} + \lambda_{\text{diffuse-smooth}} \mathcal{L}_{\text{diffuse-smooth}} \tag{9}$$

Visibility Loss. Due to the geometric sparsity inherent in feed-forward scene reconstructions at extended distances, the ray-traced visibility estimates for remote Gaussian primitives exhibit critically low confidence. Consequently, we introduce an ego-centric spatial threshold to isolate reliable visibility signals. Specifically, we strictly retain visibility computations within a 40-meter radius of the ego-vehicle, while explicitly truncating the gradient flow for all primitives beyond this boundary. This deliberate masking compels the network to optimize distant regions by exclusively leveraging the generalization priors of the feed-forward architecture, rather than relying on noisy geometric cues.

The tracer V_{ray} is used solely as an out-of-loop, non-differentiable occlusion solver. The tracer is implemented in CUDA–a BVH is built in parallel over the axis-aligned bounding boxes of the Gaussians by Morton-code sorting and kept in GPU memory, and each ray is traced by one thread that walks the tree with a thread-private stack, testing node AABBs with the slab method and pushing children in near-to-far order, then analytically intersecting rays with the Gaussian ellipsoids at the leaves and accumulating transmittance with early termination once ,thereby returning per-ray visibility for the whole batch in a single kernel launch. Rays are traced from 20K randomly sampled anchors per iteration against all scene Gaussians, and the result serves as a detached target for an L1 fit of the differentiable spherical-harmonic visibility field.

A.3 Experimental Details

Dataset Partitioning. For our experimental setup, we strictly adhere to the partitioning protocol introduced by Omni-Scene for the nuScenes dataset, employing their exact training and testing splits to ensure rigorous comparability. Furthermore, during data preparation, we run the pre-trained teachers described below to infer a corresponding intrinsic map for every input image. These predicted maps are strictly maintained at the identical spatial resolution of the original images (224×400).

Teacher Models. Real street-scene datasets provide no ground truth for intrinsic properties, so every intrinsic target in Eq. (6) is produced by an off-the-shelf estimator, which we collectively call a *teacher*. Tab. 7 states which teacher supplies which tensor, and which of its failure modes dictated the way we use it. All teachers use their public weights, are not fine-tuned on nuScenes, and are never updated by our losses: their outputs are generated once during data preparation, stored at the resolution of the input images (224×400), and kept frozen for the whole training run. MVInverseaccepts an arbitrary number of images, so we run it jointly on groups of 24 images (4 timestamps $\times$ 6 views) drawn from the same scene bin, the extra views stabilising its albedo/shading split through cross-view and cross-time consistency. This is a teacher-side setting only and does not change what our model sees: Lumen-GS still receives one timestamp $\times$ 6 views. Each intrinsic quantity is referenced by exactly one teacher.

Why the teachers are trusted only partially. Each teacher fails in its own way, and each is confined accordingly: MoGe-2 drifts in metric scale and over-smooths normals on long, texture-poor asphalt, so depth is supervised and scored as a correlation (PCC) and normals as an angular error, and a scale

Table 7: Teacher models that supply intrinsic supervision.

Teacher	Supplies	Model description
MoGe-2 (normal-vitl) (Wang et al., 2026)	depth $\mathbf{D}$, normal $\mathbf{I}_n$	A feed-forward monocular *geometry* estimator rather than a depth estimator. It regresses a 3D point map, one metrically scaled 3D point per pixel, by pairing MoGe's affine-invariant point-map prediction with a decoupled global scale factor, and the depth and surface-normal maps we supervise with are derived from that point map. We use the public *normal-vitl* checkpoint, whose normal head is the strongest of the released variants. Code: github.com/microsoft/MoGe.
LISA (Lai et al., 2024)	opacity $\mathbf{O}$	A reasoning-segmentation model built on a multimodal large language model, which grounds a free-form text query to a binary pixel mask of the image; we use it solely to delineate sky and other unbounded regions, never to segment individual objects. Code: github.com/dvlab-research/LISA.
MVInverse (Wu et al., 2025)	albedo $\mathbf{I}_{\mathrm{B}}$	A feed-forward multi-view network that predicts per-view intrinsic maps from a set of images in seconds. It accepts an arbitrary number of views, which is what makes the joint 4-timestamp $\times$ 6-view pseudo-labelling run described above possible. Code: github.com/Maddog241/mvinverse.
Ouroboros (Sun et al., 2025)	roughness $\mathbf{I}_{\mathrm{R}}$	A one-step diffusion model for joint forward and inverse rendering, trained with a cycle-consistency objective that ties the predicted materials back to the observed appearance; we take only its roughness output and deliberately leave its albedo and normal outputs unused.

error of the teacher is never counted as a geometric error of the model; the boundary of the LISA mask follows the text query rather than the object edge, which is tolerable for a soft opacity target on sky but not for a material; the albedo/shading split of MVInverse is unstable on weakly textured roads and under strong highlights, and its overall albedo level shifts across domains; and the diffusion prior inside Ouroboros over-smooths strongly specular or wet surfaces and is not deterministic. The teachers enter the objective in two restricted ways, and we deliberately do not treat their outputs as targets to be matched exactly. First, $\mathcal{L}_{\text{tea}}$ acts in the image domain on the corresponding expert head only, at weight $\lambda_{\text{tea}}{=}0.1$ against weight 1.0 on the RGB photometric term, so a teacher prediction can never outrank the observed pixel. Second, every intrinsic quantity must additionally reproduce the real RGB image through the differentiable PBR rasterizer (the λ_{pbr} term of Eq. (6)); absorbing a teacher error therefore costs photometric fidelity, and the two sources of supervision cross-check each other rather than one of them being authoritative. The ablation "Complete trust teachers" in Tab. 9 severs precisely this second path, which keeps teacher supervision but removes the gradient from the PBR branch to the intrinsic primitives, and RGB quality stays acceptable while PBR-PSNR falls to 18.92 and Albedo RMSE rises to 0.262. That is the failure mode the restricted trust scheme is designed to prevent: once the render-consistency path is removed, a network can satisfy every teacher and still leave the intrinsic layer inconsistent with the observation.

CB/R-RMSE quantify agreement with one shared, frozen teacher reference: all methods receive the same teacher supervision and are scored against the same reference. They do not measure absolute material accuracy; yet, as no existing street-view dataset offers ground-truth materials, adopting the strongest material estimation model as the reference remains the most pragmatic trade-off.

Temporal Sampling Protocol. We sample each scene bin at different temporal granularities: setting $k{=}0$ retains every timestamp of the bin, settings $k \in \{1, \ldots, 4\}$ retain every k-th timestamp, and setting *sparse* retains the target timestamp alone. Each retained timestamp contributes all six surround-view images, so the number of input frames equals six times the number of retained timestamps. The

optimizer settings and the number of optimization steps are held fixed across settings for each baseline, and identical dynamic-object masks are applied to every method during optimization. Lumen-GS is defined for a single-timestamp input and is therefore evaluated under setting *sparse* only. This is a memory constraint rather than a design choice: at our resolution one timestamp already yields 0.46M pixel-branch Gaussians together with the dynamic volume branch, and feeding several timestamps into training exceeds the memory available on our hardware (see also appendix A.5). Extending the feed-forward input to multiple timestamps is left to future work. Settings $k{=}0$ and *sparse* correspond to the *dense* and *sparse* rows of Tab. 2, respectively.

Combined feed-forward baseline (MVInv.+DA3). As far as we are aware, no existing feed-forward method relights an outdoor street scene from a single timestamp, so Tab. 2 compares against per-scene optimization methods, supplemented by a feed-forward reference composition that we assemble for this purpose. MVInv.+DA3 cascades two frozen estimators onto a shared pixel grid. DA3 is a pixel-aligned feed-forward geometry model: it regresses the depth of every input pixel, and the geometric scaffold is obtained by back-projecting those pixels through the camera intrinsics, one Gaussian per pixel, with no iterative fitting. MVInverse is likewise a per-pixel predictor: its inverse-rendering network regresses the albedo and the roughness map of each input view at the input resolution. Because both estimators are indexed by the same input pixels of the same six surround-view images, the material value of a pixel attaches to the Gaussian generated by that pixel through a single index lookup, with no registration, reprojection, or joint fitting between the two networks; it is this shared pixel parameterisation that makes two otherwise unrelated frozen models composable in one forward pass. The resulting representation is rasterised by our differentiable PBR rasteriser, so that its outputs are generated under exactly the same rendering and evaluation protocol as ours. The illumination term is likewise held constant across the two pipelines: the composition does not estimate illumination itself, but is rendered with the illumination predicted by our model for the same scene, namely the sunlight intensity and the environment light, together with the same GPS/IMU-derived sun direction supplied to every other baseline. Lighting is therefore a controlled variable, and the observed deficit is attributable to the geometric and the material layer rather than to the illumination estimate. The quantities the composition does not inherit are the per-primitive sunlight and environment visibility predicted by our representation, which are what allow our renderer to place shadows and occlusion consistently. We therefore interpret these scores as characterising the performance attainable by a purely feed-forward composition under our protocol, not as a measure of the capability of any published relighting method.

Detection Augmentation Protocol. The detection experiments measure how much our rendered weather frames help a 2D vehicle detector once they are mixed into its training data. Every accuracy figure is computed on real photographs of the same weather condition, and rendered frames appear in the training split only, with no rendered image ever evaluated. A gain can therefore be read as improved generalisation to real imagery, not as the detector being tested on data produced by our own model.

(a) Real night evaluation set. We evaluate on nuImages (Caesar et al., 2020), the street-image companion of nuScenes, restricted to its official validation split and to annotated key frames ($16{,}445$ images). Each image is assigned to day or night from three signals: the capture time parsed from the file name, the recording location taken from the log metadata (Singapore One-North, Singapore Queenstown, Boston Seaport) together with the corresponding NOAA sunrise and sunset times, and the image luma $0.2126R + 0.7152G + 0.0722B$. The astronomical criterion decides on its own when an image was captured more than 45 min outside sunrise and sunset, in which case it is labelled night. Near the two transitions that criterion is too coarse, so the luma takes over as the deciding signal. An image captured within 90 min of sunrise or sunset is labelled night only if its mean luma is below 48, and the threshold is relaxed to 72 inside the inner ± 45 min window, which keeps images that are still dark immediately after dusk or shortly before dawn. The resulting test list contains 192 night images, corresponding to 32 key frames each observed by all six surround cameras, of which 96 carry at least one vehicle and 291 car boxes in total; detection is scored on the single class *car*. Every number in Tab. 4 is computed on this list, which contains no rendered image.

(b) Rendered night augmentation set. The augmentation set is derived from the daytime images of the training pool itself. Lumen-GS is run on each of the $1{,}500$ daytime training images in a single forward pass, the reconstructed scene is then switched from day to night by replacing the sunlight and environment illumination while the geometry, the materials and the camera are left untouched, and the result is re-rendered. This yields exactly $1{,}500$ additional images, one per daytime image,

and the vehicle labels are inherited from the daytime image without any re-annotation, since neither the camera nor any object moves between the two renderings.

(c) Training conditions and confound control. Both conditions start from the same real training pool of $1{,}600$ nuImages training-split images, sampled as $1{,}500$ day and 100 night images so that night remains the rare class. The baseline trains on these $1{,}600$ images; the augmented condition adds the $1{,}500$ night renderings of the daytime subset, giving $3{,}100$. All remaining factors are held identical across the two conditions and across detectors: the single *car* class, an input resolution of 640, the same augmentation pipeline and optimization schedule, and the same evaluation list. The pair is repeated for four nano-scale detectors (YOLOv5n, YOLOv8n, YOLOv12n and YOLO-Master-N), so that the conclusion rests on four independent optimization runs rather than on a single one.

(d) Snow augmentation. The snow experiment of Tab. 12 reuses this design with the weather changed. Its real training pool of $1{,}541$ frames is compared against the same pool augmented with 840 snowy images rendered by Lumen-GS from those clear-weather frames with their annotations transferred, and the evaluation set is again real snow imagery from an external benchmark rather than any rendered frame.

Generation Rules in Harmonization Experiments. To compute the generative metrics for visual harmonization, we first process each nuScenes sequence by executing our feed-forward reconstruction and caching the physically-based rendering (PBR) outputs. Subsequently, we employ SegFormer to extract drivable-area masks across designated candidate views. By integrating these semantic masks with corresponding depth maps, camera intrinsics, and sensor-to-ego extrinsic parameters, we back-project valid road pixels into 3D spatial coordinates within the ego-vehicle reference frame. These candidate 3D points are then subjected to a rigorous filtering regimen, conditioned on depth confidence, radial distance thresholds, lateral offsets, valid 3D elevation ranges, and scene obstacle occupancy. For a given scene, the algorithm stochastically permutes the target camera sequence and iteratively samples valid road points within each view. Candidate locations must satisfy strict geometric criteria: they must reside strictly in front of the camera at a moderate viewing distance, possess robust support from neighboring road points, and maintain an adequate spatial margin from previously inserted vehicles. The vertical elevation of each vehicle is anchored to the median height of its localized road-point neighborhood. Vehicle assets are randomly sampled from our pre-baked 3DRealCar library, with yaw angles stochastically assigned from a set of feasible orientations. Should this random sampling fail to yield a valid configuration, the algorithm activates a fallback mechanism, reverting to a predefined template location aligned along the camera's forward axis. Ultimately, this procedure generates multiple non-overlapping insertion variants per scene. Prior to the final PBR rendering, we systematically re-bake the solar and ambient illumination visibility specifically for these newly inserted assets.

A.4 User Study Protocol and Statistics

Study question and design. The study is designed around a single measurement: the degree to which observers judge a vehicle asset to be integrated with the geometry and illumination of the scene into which it is inserted. It is a within-trial full-ranking experiment rather than a pairwise comparison or a rating scale. In each trial the six candidate renderings of one scene (Lumen-GS, ReCap, LumiGauss, GIR, GaussianShader, and a *direct-paste* composite in which the asset is dropped into the image without any model, no relighting and no compositing step, labelled "insert" in Fig. 7) are displayed together, and the participant assigns them six distinct ranks from 1 (best) to 6 (worst). We chose a full ranking over pairwise judgements because the six pipelines differ along several axes simultaneously (shadow direction, ground contact, highlights, reflections, overall tone), and a binary task would have required many separate comparisons that still do not define one overall ordering.

Stimuli and randomisation. Each trial shows one scene at a single insertion pose; the six images differ only in the pipeline that produced them, so asset, pose, camera and background are matched by construction within a trial. The six candidates are presented in randomised order within every trial, so no method is bound to a fixed screen position and no method name is ever shown. The exported response table records, for each participant and scene, the six methods ordered from best to worst; it records no identity, demographic or device information, and no other interface parameters were fixed by the protocol.

Participants and procedure. The study involved $P = 96$ participants and 40 scenes: 61 participants completed all 40 rankings and the remaining 35 completed between 37 and 39, yielding $N = 3793$ usable rankings. Every response is a complete 6-way ordering, so no observation was discarded and no participant–scene pair occurs twice. Participants were graduate students with research training in computer vision or graphics, drawn from the authors' own and collaborating research groups; all took part voluntarily and without compensation, and were instructed to complete every trial. The questionnaire was administered online.

Instructions shown to each participant. The questionnaire was administered online and its complete instruction text is a single sentence:

> "This study contains 40 questions. Each question shows six images. Please rank the images by how harmonious the car in them looks; rank 1 is the most harmonious."

The instruction is deliberately minimal: participants were asked for a single holistic "harmony" judgement of the inserted car and were *not* given a checklist of criteria such as shadow direction, ground contact, highlights or reflections. We keep the wording here exactly as it was used, because it bounds what the study can support: the ranking reflects an undecomposed overall impression. The instrument required a full ordering, and every one of the 3793 responses is indeed a complete permutation of the six candidates, so no tie handling was needed.

Statistics. Fig. 7 reports the empirical rank distribution of each method, and Tab. 8 summarizes the $N = 3793$ rankings by three statistics per method: the *first-choice share*, the proportion of rankings that assign rank 1 to the method; the *mean rank* over all rankings (1 best and 6 worst), the six values of which average 3.5 by construction; and the *head-to-head win rate* of Lumen-GS against each other method, the proportion of rankings in which Lumen-GS is ordered above it. The last column counts, for each of those five methods, the participants whose own win rate exceeds one half.

All reported intervals are 95% percentile bootstraps obtained by resampling the 96 participants with replacement (2×10^4 replications); rankings collected within a participant are not independent, so the participant rather than the individual ranking is the resampling unit, and each interval quantifies how far the corresponding statistic would vary across different samples of observers drawn from the population described above.

Lumen-GS is assigned rank 1 in 55.8% of the rankings (2118 of 3793) against 13.4% for the strongest baseline (510 of 3793, one ranking ahead of the direct-paste composite), a difference of 42.4 percentage points, and its head-to-head win rates lie between 68.7% and 87.1%. At the participant level, the last column of Tab. 8 makes the same point visible: all 96 observers prefer Lumen-GS to each of the four published baselines in more than half of their own trials, and 95 do so even against the direct-paste composite, so the observed direction is not attributable to a subset of the observers.

Table 8: Per-method preference statistics of the object-insertion user study ($P = 96$ participants, $N = 3793$ complete rankings over 40 scenes). The **best**, second-best and third-best entries are highlighted in the first two columns, whose entries are comparable across rows; the last two columns are quantities of ours against the method of that row and are therefore not ranked.

Method	First choice (%) ↑	Mean rank ↓	Win rate of ours (%) ↑	Observers >50%
Lumen-GS (ours)	**55.8 (54.5–57.2)**	**2.17 (2.13–2.21)**	—	—
Direct paste	13.4 (12.4–14.4)	3.02 (2.98–3.07)	68.8 (67.4–70.2)	95/96
ReCap	13.4 (12.3–14.6)	2.93 (2.89–2.98)	68.7 (67.5–69.9)	96/96
LumiGauss	8.5 (7.5–9.4)	3.48 (3.44–3.52)	74.6 (73.3–75.8)	96/96
GIR	5.8 (5.1–6.4)	4.43 (4.39–4.47)	83.8 (82.8–84.8)	96/96
GaussianShader	3.1 (2.6–3.5)	4.96 (4.92–5.00)	87.1 (86.2–88.1)	96/96

A.5 Limitations

Training dataset. Overcast weather and low illumination pose significant challenges for sun-direction prediction due to the lack of shadow and texture cues. Our model still requires prior sun directions for auxiliary training, which depend on GPS and IMU localization data. However, the Waymo dataset removes precise geolocation information for privacy protection, limiting the datasets compatible

with our training pipeline. Fortunately, the sun direction is architecturally decoupled from intrinsic prediction: no encoder feature and no intrinsic head takes $\boldsymbol{\omega}_{\text{sun}}$ as input—it enters the model only through the transport terms ($\mathbf{v}_{\text{global}}$, v_{sun}) and the rendering path. For a fixed model, the predicted intrinsics are therefore functionally independent of the sun direction at inference time, and relighting remains available when ground-truth sun directions are absent (e.g., Waymo): sun-direction accuracy there affects relit-RGB fidelity, not the intrinsics themselves. Losing the sun prior during training does propagate into the intrinsics through the PBR rendering gradients—retraining with noisy or estimated directions raises $\mathbf{C}_{\text{B}}$-RMSE from 0.088 to 0.112 ($\sigma = 25°$) or 0.117 (*est. pred*) and lowers PBR-PSNR by 1.84 dB and 1.22 dB, respectively (Tab. 11). In the future, we plan to construct geolocation annotations for Waymo using manual labeling and AI-based geographic search, although this may introduce about 3% error in sun-position estimation.

Static Reconstruction. Our model is still limited to static reconstruction. Even with an efficient CUDA implementation, the static reconstruction pipeline already nearly exhausts GPU memory, and extending it to a dynamic pipeline would still require further optimization. Since Lumen-GS focuses on single-timestamp feed-forward reconstruction, dynamic objects (e.g., moving vehicles and pedestrians) within the timestamp are reconstructed as static entities in the current frame.

Ambiguity of complex illumination. Due to the complexity of illumination in open scenes, environments often contain difficult-to-model emissive light sources and their secondary reflections. Although the self-emissive primitives in our representation can approximate these lighting effects to some extent (as shown in Fig. 8), ambiguities still remain (as shown in Fig. 9), with some secondary-reflection textures being inadvertently learned into the albedo. Addressing this phenomenon would require incorporating a ray-tracing-capable Gaussian renderer, which would, however, introduce substantial computational overhead. In terms of coverage, the illumination that Lumen-GS controls *by itself* is restricted to sunlight and sky: emissive sources already present in a night scene—vehicle headlights, street lights, traffic signals, LED billboards—are at best partially absorbed by $\mathbf{r}_{\text{local}}$, and are neither localised, relit nor toggled; the point lights in Fig. 16 are hand-specified at inference time and are not predicted, so that result does not contradict this limitation. Likewise, the GGX microfacet model only approximately fits strongly specular or layered materials such as glass, wet asphalt, car paint and chrome. Therefore, developing an elegant solution to this issue is one of the core directions for our future work.



Figure 8: Self-emissive light and secondary reflections.

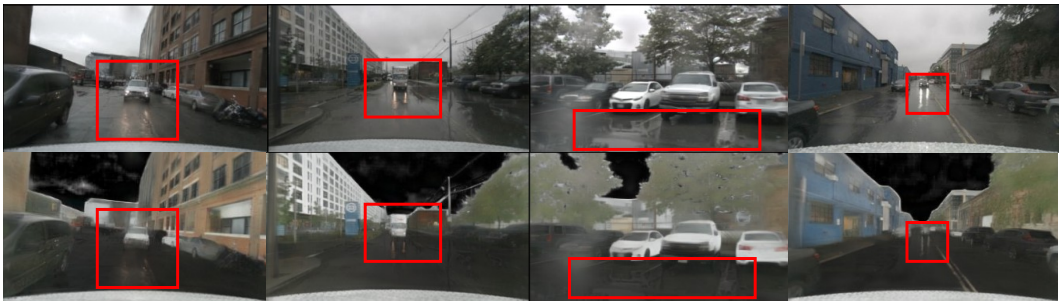

Figure 9: Bad case under secondary reflection problems.

A.6 CUSTOM CUDA

Notation Note: For simplicity of derivation in this section, we use $\mathbf{b}$, r, $\mathbf{L}_{global}$, $\mathbf{L}_{local}$, *and* q *to denote the albedo* $\mathbf{C}_B$, *roughness* R, *global environment radiance* $\mathbf{r}_{global}$, *local illumination residual* $\mathbf{r}_{local}$, *and global environment visibility* $\mathbf{v}_{global}$, *respectively.* Although we simplify the global light into

3-channel RGB values, the following derivation still remains in the more general spherical harmonics form (RGB represents the 0-th order form of spherical harmonics).

Analytical backward of the GGX specular term. For simplicity of derivation, we omit several implementation details such as clamping and normal flipping, and focus on the unclamped core of the GGX specular term with respect to the normalized variables. Let $\mathbf{l}, \mathbf{v}, \mathbf{n}$ be normalized light, view, and normal directions. We define

$$\mathbf{t} = \mathbf{l} + \mathbf{v}, \qquad \mathbf{h} = \frac{\mathbf{t}}{\|\mathbf{t}\|},$$

$$x = \mathbf{n}^\top \mathbf{l}, \quad y = \mathbf{n}^\top \mathbf{v}, \quad z = \mathbf{n}^\top \mathbf{h}, \quad w = \mathbf{v}^\top \mathbf{h}.$$

The GGX specular term s is

$$s = \frac{F\beta}{4\pi D_0^2 D_1 D_2},$$

where

$$\alpha = r^2, \qquad \beta = \alpha^2 = r^4, \qquad k = \frac{\alpha + 2r + 1}{8},$$

$$F = F_0 + (1 - F_0)2^{(-5.55473w - 6.98316)w},$$

$$D_0 = z^2(\beta - 1) + 1, \quad D_1 = y(1 - k) + k, \quad D_2 = x(1 - k) + k.$$

Ignoring clamping and differentiating only the normalization of $\mathbf{h}$, the normal gradient is

$$\frac{\partial s}{\partial \mathbf{n}} = \left(\frac{\partial s}{\partial x}\frac{\partial x}{\partial \mathbf{n}} + \frac{\partial s}{\partial y}\frac{\partial y}{\partial \mathbf{n}} + \frac{\partial s}{\partial z}\frac{\partial z}{\partial \mathbf{n}}\right) = \left(\frac{\partial s}{\partial x}\mathbf{l} + \frac{\partial s}{\partial y}\mathbf{v} + \frac{\partial s}{\partial z}\mathbf{h}\right)$$

$$= s\left(-\frac{1-k}{D_2}\mathbf{l} - \frac{1-k}{D_1}\mathbf{v} - \frac{4z(\beta-1)}{D_0}\mathbf{h}\right) = \frac{F\beta}{4\pi D_0^2 D_1 D_2}\left(-\frac{1-k}{D_2}\mathbf{l} - \frac{1-k}{D_1}\mathbf{v} - \frac{4z(\beta-1)}{D_0}\mathbf{h}\right).$$

For the light direction, the half-vector also depends on $\mathbf{l}$. With

$$J_h = \frac{I}{\|\mathbf{t}\|} - \frac{\mathbf{t}\mathbf{t}^\top}{\|\mathbf{t}\|^3},$$

we have

$$\frac{\partial s}{\partial \mathbf{l}} = \left(\frac{\partial s}{\partial x}\mathbf{n} + J_h^\top\left(\frac{\partial s}{\partial z}\mathbf{n} + \frac{\partial s}{\partial w}\mathbf{v}\right)\right) = s\left(-\frac{1-k}{D_2}\mathbf{n} + J_h^\top\left(-\frac{4z(\beta-1)}{D_0}\mathbf{n} + \frac{F_w}{F}\mathbf{v}\right)\right)$$

$$= \frac{F\beta}{4\pi D_0^2 D_1 D_2}\left(-\frac{1-k}{D_2}\mathbf{n} + J_h^\top\left(-\frac{4z(\beta-1)}{D_0}\mathbf{n} + \frac{F_w}{F}\mathbf{v}\right)\right).$$

For the view direction,

$$\frac{\partial s}{\partial \mathbf{v}} = \left(\frac{\partial s}{\partial y}\mathbf{n} + \frac{\partial s}{\partial w}\mathbf{h} + J_h^\top\left(\frac{\partial s}{\partial z}\mathbf{n} + \frac{\partial s}{\partial w}\mathbf{v}\right)\right)$$

$$= s\left(-\frac{1-k}{D_1}\mathbf{n} + \frac{F_w}{F}\mathbf{h} + J_h^\top\left(-\frac{4z(\beta-1)}{D_0}\mathbf{n} + \frac{F_w}{F}\mathbf{v}\right)\right)$$

$$= \frac{F\beta}{4\pi D_0^2 D_1 D_2}\left(-\frac{1-k}{D_1}\mathbf{n} + \frac{F_w}{F}\mathbf{h} + J_h^\top\left(-\frac{4z(\beta-1)}{D_0}\mathbf{n} + \frac{F_w}{F}\mathbf{v}\right)\right),$$

where

$$F_w = (1 - F_0)2^{(-5.55473w - 6.98316)w}\ln 2(-11.10946w - 6.98316).$$

Finally, the roughness gradient is

$$\frac{\partial s}{\partial r} = s\left(\frac{\beta_r}{\beta} - 2\frac{D_{0,r}}{D_0} - \frac{D_{1,r}}{D_1} - \frac{D_{2,r}}{D_2}\right)$$

$$= \frac{F\beta}{4\pi D_0^2 D_1 D_2}\left(\frac{4r^3}{\beta} - 2\frac{4r^3 z^2}{D_0} - \frac{(1-y)(r+1)/4}{D_1} - \frac{(1-x)(r+1)/4}{D_2}\right).$$

Analytical backward of the outer rendering equation. For one incident sample with direction $\mathbf{l}$, let $\{Y_i(\mathbf{l})\}$ denote the spherical harmonics (SH) basis evaluated at $\mathbf{l}$. The local incident lighting, global incident lighting, and visibility are represented by SH expansions:

$$\mathbf{L}_{\text{local}} = \sum_{i=0}^{S_{\text{incident}}-1} \mathbf{I}_i Y_i(\mathbf{l}), \quad \mathbf{L}_{\text{global}} = \sum_{i=0}^{S_{\text{direct}}-1} \mathbf{G}_i Y_i(\mathbf{l}), \quad q = \sum_{i=0}^{S_{\text{vis}}-1} V_i Y_i(\mathbf{l}).$$

Here $\mathbf{I}_i \in \mathbb{R}^3$ and $\mathbf{G}_i \in \mathbb{R}^3$ are SH coefficients for local and global incident lighting, respectively, and $V_i \in \mathbb{R}$ are SH coefficients for visibility.

For simplicity of derivation, we ignore clamping operations, including the non-negativity clamp of incident lighting, the visibility clipping, and the cosine clamp. The total incident lighting is

$$\mathbf{L} = q\mathbf{L}_{\text{global}} + \mathbf{L}_{\text{local}}.$$

The diffuse BRDF, cosine term, and Monte Carlo weight are

$$\mathbf{f}_d = \frac{\mathbf{b}}{\pi}, \qquad m = \mathbf{n}^\top \mathbf{l}, \qquad A = \frac{2\pi m}{N}.$$

The per-sample color and diffuse-light outputs are

$$\mathbf{C} = A(\mathbf{f}_d + s\mathbf{1}) \odot \mathbf{L} = \frac{2\pi m}{N}\left(\frac{\mathbf{b}}{\pi} + s\mathbf{1}\right) \odot \mathbf{L}, \quad \mathbf{C}_d = A\mathbf{L} = \frac{2\pi m}{N}\mathbf{L}.$$

Let

$$\bar{\mathbf{C}} = \frac{\partial \mathcal{L}}{\partial \mathbf{C}}, \qquad \bar{\mathbf{C}}_d = \frac{\partial \mathcal{L}}{\partial \mathbf{C}_d}.$$

The outer gradient with respect to the GGX scalar is

$$\bar{s} = \frac{\partial \mathcal{L}}{\partial s} = \left\langle \bar{\mathbf{C}}, \frac{\partial \mathbf{C}}{\partial s} \right\rangle = \langle \bar{\mathbf{C}}, A\mathbf{L} \rangle = A\langle \bar{\mathbf{C}}, \mathbf{L} \rangle = \frac{2\pi m}{N}\langle \bar{\mathbf{C}}, \mathbf{L} \rangle.$$

The incident-light gradient is

$$\bar{\mathbf{L}} = \frac{\partial \mathcal{L}}{\partial \mathbf{L}} = \frac{\partial \mathcal{L}}{\partial \mathbf{C}}\frac{\partial \mathbf{C}}{\partial \mathbf{L}} + \frac{\partial \mathcal{L}}{\partial \mathbf{C}_d}\frac{\partial \mathbf{C}_d}{\partial \mathbf{L}} = \frac{2\pi m}{N}\left[\bar{\mathbf{C}} \odot \left(\frac{\mathbf{b}}{\pi} + s\mathbf{1}\right) + \bar{\mathbf{C}}_d\right].$$

The cosine-term gradient is

$$\bar{m} = \frac{\partial \mathcal{L}}{\partial m} = \frac{\partial \mathcal{L}}{\partial \mathbf{C}}\frac{\partial \mathbf{C}}{\partial m} + \frac{\partial \mathcal{L}}{\partial \mathbf{C}_d}\frac{\partial \mathbf{C}_d}{\partial m} = \frac{2\pi}{N}\left[\left\langle \bar{\mathbf{C}}, \left(\frac{\mathbf{b}}{\pi} + s\mathbf{1}\right) \odot \mathbf{L} \right\rangle + \langle \bar{\mathbf{C}}_d, \mathbf{L} \rangle\right].$$

Therefore, the one-sample gradients are

$$\bar{\mathbf{n}} = \frac{\partial \mathcal{L}}{\partial \mathbf{n}} = \frac{\partial \mathcal{L}}{\partial s}\frac{\partial s}{\partial \mathbf{n}} + \frac{\partial \mathcal{L}}{\partial m}\frac{\partial m}{\partial \mathbf{n}} = \bar{s}\frac{\partial s}{\partial \mathbf{n}} + \bar{m}\,\mathbf{l},$$

$$\bar{\mathbf{v}} = \frac{\partial \mathcal{L}}{\partial \mathbf{v}} = \frac{\partial \mathcal{L}}{\partial s}\frac{\partial s}{\partial \mathbf{v}} = \bar{s}\frac{\partial s}{\partial \mathbf{v}},$$

$$\frac{\partial \mathcal{L}}{\partial \mathbf{l}} = \frac{\partial \mathcal{L}}{\partial s}\frac{\partial s}{\partial \mathbf{l}} + \frac{\partial \mathcal{L}}{\partial m}\frac{\partial m}{\partial \mathbf{l}} + \frac{\partial \mathcal{L}}{\partial \mathbf{L}}\frac{\partial \mathbf{L}}{\partial \mathbf{l}} = \bar{s}\frac{\partial s}{\partial \mathbf{l}} + \bar{m}\,\mathbf{n} + \sum_i \left[\langle \bar{\mathbf{L}}, \mathbf{I}_i + q\mathbf{G}_i + V_i\mathbf{L}_{\text{global}} \rangle\right] \nabla_1 Y_i(\mathbf{l}).$$

The roughness and base-color gradients are

$$\frac{\partial \mathcal{L}}{\partial r} = \frac{\partial \mathcal{L}}{\partial s}\frac{\partial s}{\partial r} = \bar{s}\frac{\partial s}{\partial r},$$

$$\frac{\partial \mathcal{L}}{\partial \mathbf{b}} = \frac{\partial \mathcal{L}}{\partial \mathbf{C}}\frac{\partial \mathbf{C}}{\partial \mathbf{b}} = \frac{\partial \mathcal{L}}{\partial \mathbf{C}}\frac{\partial \mathbf{C}}{\partial \mathbf{f}_d}\frac{\partial \mathbf{f}_d}{\partial \mathbf{b}} = \bar{\mathbf{C}}A \odot \mathbf{L}\frac{1}{\pi} = \frac{2m}{N}(\bar{\mathbf{C}} \odot \mathbf{L}).$$

For the SH lighting and visibility coefficients, since clamping is ignored, we have

$$\frac{\partial \mathcal{L}}{\partial \mathbf{I}_i} = \frac{\partial \mathcal{L}}{\partial \mathbf{L}}\frac{\partial \mathbf{L}}{\partial \mathbf{I}_i} = \frac{\partial \mathcal{L}}{\partial \mathbf{L}}\frac{\partial \mathbf{L}_{\text{local}}}{\partial \mathbf{I}_i} = \bar{\mathbf{L}}Y_i(\mathbf{l}) = \frac{2\pi m}{N}\left[\bar{\mathbf{C}} \odot \left(\frac{\mathbf{b}}{\pi} + s\mathbf{1}\right) + \bar{\mathbf{C}}_d\right] Y_i(\mathbf{l}),$$

$$\frac{\partial \mathcal{L}}{\partial \mathbf{G}_i} = \frac{\partial \mathcal{L}}{\partial \mathbf{L}} \frac{\partial \mathbf{L}}{\partial \mathbf{G}_i} = \frac{\partial \mathcal{L}}{\partial \mathbf{L}} \frac{\partial \mathbf{L}}{\partial \mathbf{L}_{\text{global}}} \frac{\partial \mathbf{L}_{\text{global}}}{\partial \mathbf{G}_i} = \bar{\mathbf{L}} q Y_i(\mathbf{l}) = q \frac{2\pi m}{N} \left[\bar{\mathbf{C}} \odot \left(\frac{\mathbf{b}}{\pi} + s\mathbf{1} \right) + \bar{\mathbf{C}}_d \right] Y_i(\mathbf{l}),$$

$$\frac{\partial \mathcal{L}}{\partial V_i} = \frac{\partial \mathcal{L}}{\partial \mathbf{L}} \frac{\partial \mathbf{L}}{\partial q} \frac{\partial q}{\partial V_i} = \left\langle \bar{\mathbf{L}}, \mathbf{L}_{\text{global}} \right\rangle Y_i(\mathbf{l}) = \left\langle \frac{2\pi m}{N} \left[\bar{\mathbf{C}} \odot \left(\frac{\mathbf{b}}{\pi} + s\mathbf{1} \right) + \bar{\mathbf{C}}_d \right], \mathbf{L}_{\text{glob}} \right\rangle Y_i(\mathbf{l}).$$

For multiple incident samples, each quantity takes a subscript j, and the final gradients are accumulated as

$$\frac{\partial \mathcal{L}}{\partial \theta} = \sum_{j=0}^{N-1} \frac{\partial \mathcal{L}_j}{\partial \theta}, \qquad \theta \in \{\mathbf{n}, \mathbf{v}, \mathbf{l}, r, \mathbf{b}, \mathbf{I}_i, \mathbf{G}_i, V_i\}.$$

Our evaluations demonstrate a significant performance disparity between the native Python and custom CUDA implementations. As Fig. 10 shows, when the point count increases, the Python version exhibits an exponential rise in execution time; for example, processing 475k points takes 613ms in Python versus only 22ms in CUDA, yielding a nearly 30× speedup. Furthermore, the Python implementation suffers from severe memory bottlenecks due to the massive allocation of intermediate tensors during automatic differentiation. Its memory footprint surges to 51 GB at 475k points and inevitably triggers an Out-Of-Memory (OOM) error at 761k points (Fig. 11). In contrast, the CUDA implementation maintains a highly efficient, linear memory footprint, seamlessly scaling up to 5 million points without exhausting GPU resources.

These results underscore the absolute necessity of custom CUDA kernels for compute-intensive tasks like physically based rendering. Relying exclusively on high-level PyTorch APIs causes severe memory bloat and latency, as each basic arithmetic or activation operation triggers independent kernel launches and redundant global memory read/writes. Our custom CUDA operator mitigates this overhead through kernel fusion, computing complex mathematical sequences within a single GPU thread and keeping intermediate results in fast local registers. By eliminating frequent global memory access, this approach fundamentally resolves the OOM bottleneck and unlocks the capability for large-scale, real-time training and rendering.

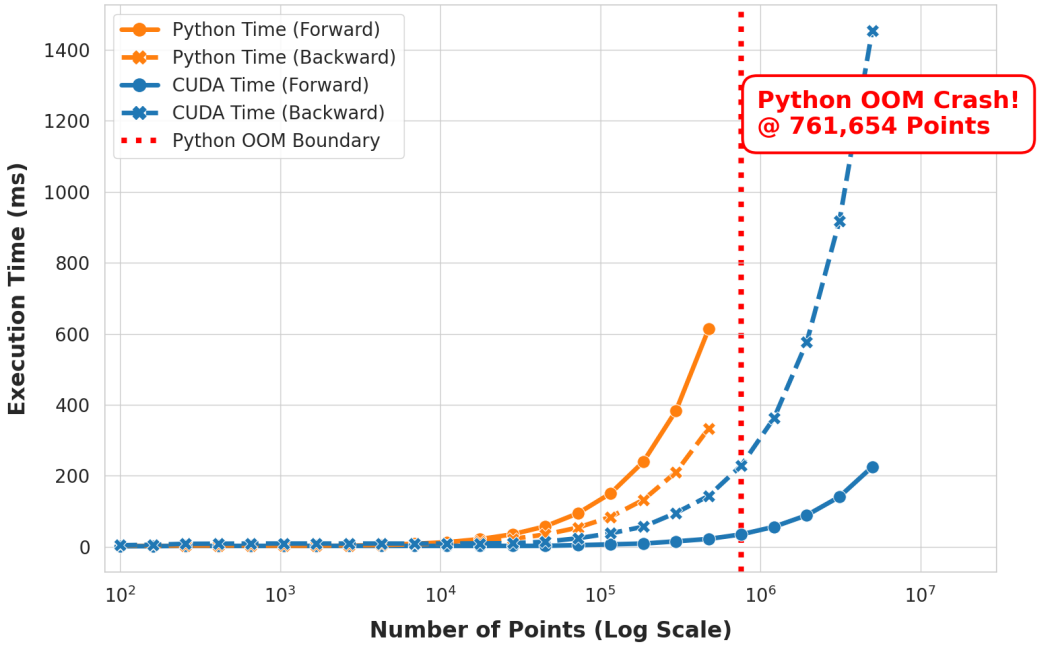

Figure 10: **Execution time comparison.** Execution time (in milliseconds) of the forward and backward passes across varying point counts. The native Python implementation exhibits an exponential increase in latency, whereas the custom CUDA kernel demonstrates efficient, near-linear scaling, successfully processing up to 5 million points.

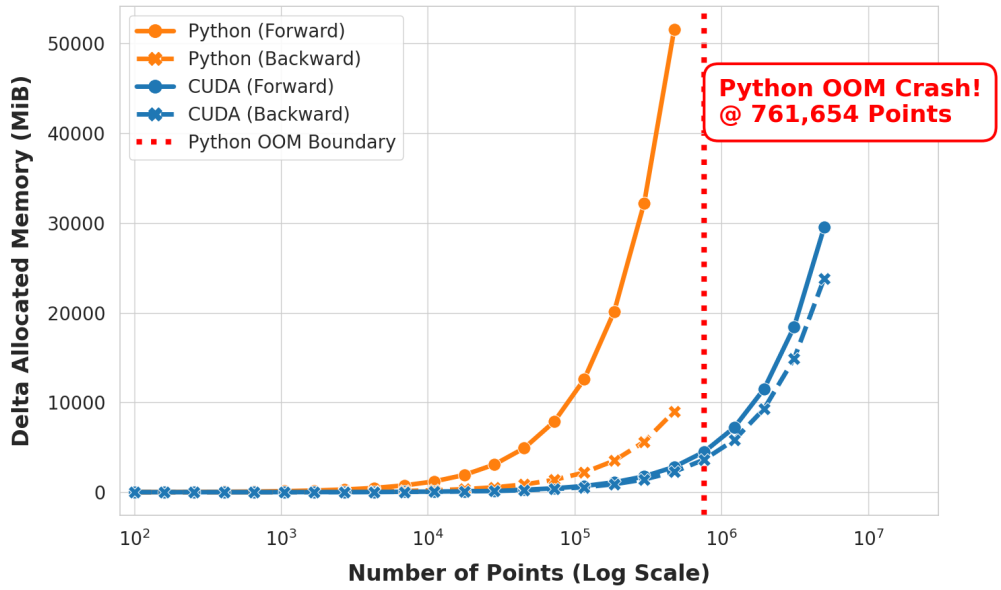

Figure 11: **Memory allocation and OOM bottleneck.** Delta allocated memory (in MiB) during the forward and backward passes. The Python implementation suffers from severe memory bloat, triggering an Out-Of-Memory (OOM) crash at 761k points (red dotted line). In contrast, the CUDA implementation maintains a minimal and stable memory footprint.

A.7 Detailed Ablation Results

To systematically evaluate the contribution of each core design in Lumen-GS, we conduct an extensive ablation study on the nuScenes validation set. By default the PBR branch back-propagates into albedo and into both visibilities, but not into roughness; we write $\mathrm{sg}_{\mathrm{pbr}}(\cdot)$ for blocking that path on a given primitive, and Tab. 9 reports what happens when each of these choices is reversed. Quantitative results are also reported in Tab. 9, where we isolate the effects of the Physically-Aligned Decoder (PAD), hybrid feature branches, PBR-aware gradient supervision, intrinsic teacher models, and explicit sun modeling.

Effect of PAD. Replacing the pixel-branch PAD with a plain MLP (w/o Pixel PAD) yields a noticeable decline in both RGB rendering (PSNR drops from 25.51 to 25.17, LPIPS rises to 0.225) and PBR quality (PBR-PSNR falls to 22.12). A similar degradation is observed when the volume-branch PAD is removed. When both PAD modules are replaced by standard MLPs (w/o All PAD), performance deteriorates further across all metrics, confirming that the expert-head architecture with physically constrained feature interaction is indispensable for high-fidelity disentanglement.

Contribution of Hybrid Feature Branches. Removing the pixel branch produces the most severe degradation (PSNR 22.84, LPIPS 0.358), underscoring its critical role in capturing fine-grained, view-dependent details. Eliminating the volume branch leads to a milder but still substantial drop (PSNR 24.56), indicating that the two branches are complementary: pixel features provide local precision while volume features ensure global geometric consistency.

Impact of PBR Gradient Supervision. We investigate the importance of the PBR gradient flow with respect to each intrinsic attribute. Disabling PBR gradients on albedo ($\mathbf{C}_{\mathrm{B}} \rightarrow \mathrm{sg}_{\mathrm{pbr}}(\mathbf{C}_{\mathrm{B}})$) slightly improves Albedo RMSE (0.084) at the cost of PBR rendering consistency, so we still keep that gradient. Removing gradients for sun visibility causes a clear drop in all PBR metrics. Removing PBR gradients for roughness does not take significant impact on metric in Tab. 9. Critically, complete removal of all PBR gradients ("Complete trust teachers") maintains acceptable RGB quality yet catastrophically harms the PBR branch (PBR-PSNR 18.92, Albedo RMSE 0.262), demonstrating that joint supervision from both the differentiable PBR rasterizer and the standard 3DGS rasterizer is essential for physically meaningful decomposition.

Table 9: Detailed ablation study of PAD decoders, feature branches, primitive gradients, intrinsic teachers and sun modeling in Lumen-GS.

Model	Remark	Rendering					PBR Branch			
		PSNR↑	LPIPS↓	SSIM↑	PCC↑	D-RMSE↓	PSNR↑	LPIPS↓	SSIM↑	C_B-RMSE↓
w/o Pixel PAD	Pixel PAD → MLP	25.17	0.225	0.748	0.882	5.53	22.12	0.278	0.706	0.090
w/o Volume PAD	Volume PAD → MLP	25.15	0.225	0.749	0.883	5.53	22.44	0.272	0.708	0.089
w/o All PAD	Only MLP decoder	24.96	0.232	0.741	0.882	5.54	22.07	0.281	0.703	0.091
w/o Pixel branch	Remove pixel Gaussian	22.84	0.358	0.652	0.741	6.75	19.85	0.402	0.621	0.134
w/o Volume branch	Remove volume Gaussian	24.56	0.264	0.728	0.865	6.01	21.68	0.294	0.692	0.102
w/o albedo PBR grad	$\mathbf{C}_B \to \mathrm{sg}_{\mathrm{pbr}}(\mathbf{C}_B)$	25.03	0.230	0.737	**0.889**	5.48	21.29	0.292	0.699	**0.084**
w/o sunvis PBR grad	$v_{\mathrm{sun}} \to \mathrm{sg}_{\mathrm{pbr}}(v_{\mathrm{sun}})$	24.92	0.236	0.734	0.879	5.50	21.03	0.304	0.686	0.116
w/ roughness PBR grad	$R \to \mathrm{sg}_{\mathrm{pbr}}(R)$	25.42	0.223	0.751	0.886	5.41	22.48	0.275	0.712	0.089
w/o all PBR grad	Complete trust teachers	25.02	0.238	0.740	0.875	5.58	18.92	0.428	0.585	0.262
w/o teachers	Only PBR constraints	24.51	0.278	0.715	0.842	5.85	21.14	0.308	0.675	0.112
w/o sun modeling	Remove sun modeling	25.16	0.228	0.744	0.884	5.41	20.73	0.367	0.674	0.167
w/o sun GT input	Sun direction Pred	25.26	0.226	0.747	0.884	5.42	21.33	0.291	0.689	0.117
Lumen-GS	Full model	**25.51**	**0.221**	**0.754**	0.888	**5.41**	**22.55**	**0.269**	**0.715**	0.088

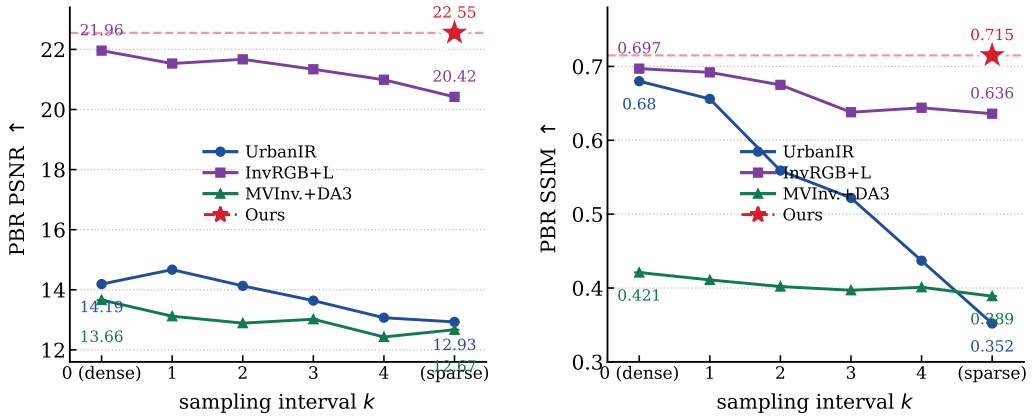

Figure 12: **Relighting quality versus temporal sampling density**, in terms of PBR PSNR (left) and PBR SSIM (right); the values are tabulated in Tab. 10, and the dashed segment of Lumen-GS is a visual guide only.

Role of Intrinsic Teachers and Sun Modeling. Training without teacher guidance (w/o teachers) reduces performance across the board, confirming that indirect supervision from strong intrinsic teachers provides valuable physical priors that stabilize learning. Finally, ablating sun modeling degrades performance on PBR branch (PSNR 25.16, PBR-PSNR 20.73, Albedo RMSE 0.167), validating that explicit modeling of directional sunlight is indispensable for outdoor street scenes where strong, view-dependent illumination dominates.

These controlled experiments indicate that the PAD architecture, the hybrid pixel–volume feature aggregation, the selective PBR gradient supervision, the teacher-guided intrinsic priors and the sunlight-integrated representation each contribute, and that removing any one of them costs quality. Under our protocol Lumen-GS remains competitive with the strongest feed-forward baselines in novel-view synthesis and with iterative inverse-rendering methods in PBR rendering quality; we do not read these ablations as evidence of absolute intrinsic accuracy, which the shared teacher references cannot measure.

Sensitivity to Temporal Sampling Density. Fig. 12 reports PBR rendering quality as the temporal density of the scene bin is reduced, in terms of PBR PSNR (left) and PBR SSIM (right), and Tab. 10 tabulates the values behind it: setting $k{=}0$ retains every timestamp of the bin, $k \in \{1, \ldots, 4\}$ retains every k-th timestamp, and *sparse* retains the target timestamp alone (appendix A.3), so the two endpoint columns are exactly the *dense* and *sparse* rows of Tab. 2. Lumen-GS is defined for a single-timestamp input and is therefore reported under *sparse* only; in Fig. 12 it appears as the red star, and the dashed segment through it joins no other point. Every baseline loses ground as observations are removed, by 0.99–1.54 dB in PBR-PSNR, whereas a single forward pass from the single-timestamp input already reaches 22.55 dB: that is +9.62 dB over UrbanIR and +9.88 dB over MVInv.+DA3 measured *at the same sparse setting* (+8.36 and +8.89 dB measured against their

Table 10: **PBR rendering quality versus temporal sampling density**; these are the values plotted in Fig. 12. Colour ranks the methods inside the *sparse* column, the only column in which every method is given the same input.

Sampling interval k	0 **(dense)**	1	2	3	4	**sparse**
PBR-PSNR (dB)↑						
UrbanIR	14.19	14.67	14.13	13.64	13.07	12.93
InvRGB+L	21.96	21.53	21.67	21.34	20.99	20.42
MVInv.+DA3	13.66	13.12	12.89	13.02	12.43	12.67
Lumen-GS	–	–	–	–	–	**22.55**
PBR-SSIM↑						
UrbanIR	0.680	0.656	0.559	0.522	0.437	0.352
InvRGB+L	0.697	0.692	0.675	0.638	0.644	0.636
MVInv.+DA3	0.421	0.411	0.402	0.397	0.401	0.389
Lumen-GS	–	–	–	–	–	**0.715**

Table 11: **Behaviour of Lumen-GS under a noisy solar direction.** Each row is a model trained from scratch with the GPS/IMU direction corrupted, at training and at inference, by an isotropic angular perturbation of standard deviation σ; *est. pred* uses a predicted direction (*w/o sun GT input* in Tab. 9) and *random* a direction drawn uniformly on the upper hemisphere. Colour marks the **best**, second and third entry per column;

	RGB branch			**PBR branch**			
Noise σ	**PSNR**↑	**SSIM**↑	**LPIPS**↓	**PSNR**↑	**SSIM**↑	**LPIPS**↓	$\mathbf{C_B}$**-RMSE**↓
0°	**25.51**	**0.754**	**0.221**	**22.55**	**0.715**	**0.269**	**0.088**
5°	25.41	0.750	0.225	22.42	0.710	0.274	0.091
10°	25.49	0.753	0.222	22.46	0.712	0.272	0.093
15°	25.43	0.749	0.224	22.11	0.698	0.281	0.098
20°	25.02	0.738	0.233	21.68	0.701	0.290	0.102
25°	24.68	0.722	0.248	20.71	0.658	0.323	0.112
est. pred	25.26	0.747	0.226	21.33	0.689	0.291	0.117
random	23.61	0.696	0.267	18.72	0.592	0.381	0.146

dense settings), and $+0.59$ dB over the strongest baseline, InvRGB+L, even when the latter is run in its *dense* configuration. Individual points are non-monotonic in both metrics and in both directions (UrbanIR rises from 14.19 to 14.67 dB between $k{=}0$ and 1, and the PBR SSIM of InvRGB+L rises from 0.638 to 0.644 between $k{=}3$ and 4); we attribute this to moving objects and exposure drift inside dense bins, which penalise per-scene optimization more strongly than feed-forward inference. The two metrics, however, disagree sharply in magnitude: over the same sweep the PBR SSIM of UrbanIR falls from 0.680 to 0.352 (-48% relative), against -9% for InvRGB+L and -8% for MVInv.+DA3, and the intrinsic view is sharper still, with the mean normal angle of UrbanIR rising from $12.49°$ to $34.91°$ (Tab. 2). Removing observations therefore damages the structural consistency of the relit render far more than its pixel-wise error, a distinction that PSNR alone would hide. We finally note that the two strongest baselines react to reduced input density at markedly different rates (-48% vs. -9% in PBR SSIM); a deeper account of why per-scene optimization degrades so unevenly is beyond the scope of this work.

Replacing the GPS/IMU prior with an estimator. We additionally train a solar-direction estimator whose network structure mirrors that of the light-context branch and differs from it only in a widened decoding head; it is supervised with the ground-truth solar direction provided by nuScenes, with the loss defined as follows. The results of Tab. 9 (*w/o sun GT input*) show that replacing the ground-truth direction with the estimated one leaves the RGB branch almost unaffected, whereas the PBR branch loses 1.22 dB of quality. The directional error induced by the estimator is close to the $20°$ level of the synthetic sweep in Tab. 11, indicating that under sparse single-frame input the cues available for inferring the solar direction are limited and a high-accuracy estimate cannot be recovered. Fortunately, vehicle-mounted hardware offers the potential to resolve the solar direction directly.

Training and Inference with a Noisy Sun Prior. Tab. 11 and Fig. 13 report the performance of models retrained at each level of solar-direction noise, scored with the metrics of Tab. 2. As the noise standard deviation σ increases, the solar direction supplied to the model is corrupted in lockstep at

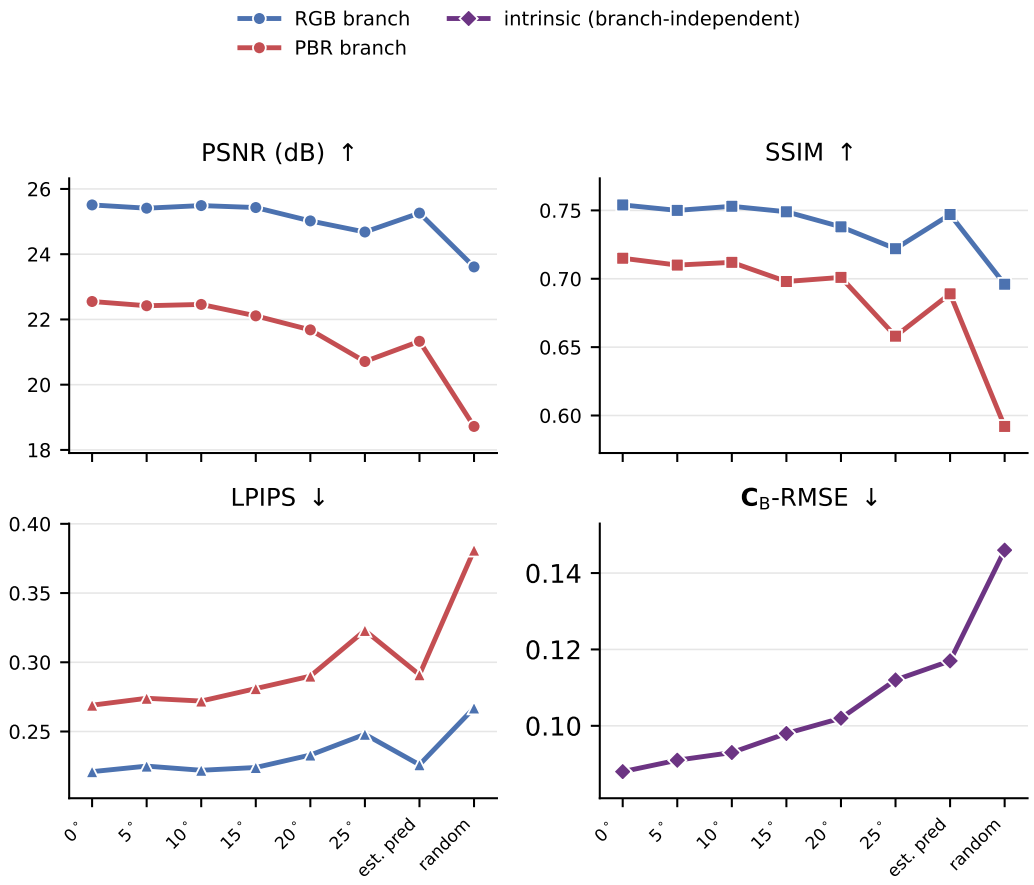

Figure 13: **Robustness to a noisy solar direction** for both branches; the values are those of Tab. 11, and the last two ticks are categories rather than noise levels.

both training and inference time, while the architecture, training data, and optimization settings are held fixed. Within $\sigma \leq 10^\circ$ the PBR branch degrades only mildly—by 0.13 dB of PSNR and 0.005 of SSIM, together with a gain of 0.005 in LPIPS—a reduction that remains within an acceptable range; over the full grid the same three quantities move by -1.84 dB, -0.057 and $+0.054$, while the albedo error $\mathbf{C}_\mathrm{B}$-RMSE grows monotonically from 0.088 to 0.112. As Fig. 13 shows, solar-direction noise affects the PBR-related metrics markedly more than the RGB branch. Taken together, these results indicate that our model is robust to solar-direction deviations of roughly 15° or less, beyond which the PBR rendering quality degrades observably. This finding carries direct implications for deployment: acquisition timestamps can be captured essentially exactly, so the solar position can be computed to high precision, while ordinary GPS positioning readily achieves metre-level accuracy and the heading accuracy of dual-antenna GNSS reaches 0.2°–1°. The pose error introduced by on-vehicle hardware therefore lies well within the 10° tolerance of our model, indicating that deployment on real vehicle-mounted platforms is feasible.

Snowy Weather Detection Augmentation. To further validate the utility of Lumen-GS under adverse weather conditions, we assess its capability to generate realistic snowy scenes for vehicle detection data augmentation. Unlike approaches that treat snow as appearance-entangled particles, Lumen-GS endows snow particles with intrinsic properties (opacity, albedo, normals, and roughness) that interact physically with the predicted light context, producing more plausible illumination, highlights, and shadows on accumulated snow. We train the same four YOLO detectors on a long-tail dataset consisting of 1,500 clear-weather frames and only 41 snowy frames, then augment the training set with snowy images synthesized by Lumen-GS from the clear-weather frames while reusing the original vehicle annotations; as in the night experiment, the evaluation set consists solely of real snow imagery and contains no rendered frame (protocol in appendix A.3). As reported in Tab. 12, the generated snowy data consistently improves detection performance across all models. For YOLOv5n,



Figure 14: Night-time and snow simulation results. Labels are transferred from daytime frames.

Table 12: Performance comparison of YOLO models with or without car labels in snowy weather generated by our model. Both settings are evaluated on real snow imagery; no rendered image appears in the test set.

YOLOv5n (Jocher, 2020)

Settings	P	R	mAP_{50}	$mAP_{50:95}$
w/o Gen	0.696	0.471	0.538	0.315
w/ Gen	**0.781**	**0.494**	**0.591**	**0.345**

YOLOv8n (Jocher et al., 2023)

Settings	P	R	mAP_{50}	$mAP_{50:95}$
w/o Gen	0.709	0.501	0.567	0.341
w/ Gen	**0.723**	**0.535**	**0.595**	**0.348**

YOLOv12n (Tian et al., 2026)

Settings	P	R	mAP_{50}	$mAP_{50:95}$
w/o Gen	0.709	0.425	0.495	0.292
w/ Gen	**0.720**	**0.469**	**0.536**	**0.318**

YOLO-Master (Lin et al., 2025c)

Settings	P	R	mAP_{50}	$mAP_{50:95}$
w/o Gen	0.683	0.463	0.536	0.309
w/ Gen	**0.716**	**0.484**	**0.556**	**0.321**

precision increases from 0.696 to 0.781 and mAP_{50} rises by 0.053. Comparable gains are observed for YOLOv8n (+0.028 in mAP_{50}), YOLOv12n (+0.041), and YOLO-Master (+0.020). The stricter $mAP_{50:95}$ metric also advances by 0.007–0.030, confirming improved localization under challenging snowy conditions. These results highlight that Lumen-GS serves as an effective, annotation-free data generation engine for adverse-weather scenarios. By enabling physically grounded snow simulation that respects scene illumination, the model helps mitigate the long-tail distribution and delivers measurable gains in real-time 2D perception, which is relevant to simulation at scale.

A.8 More Visualization

Novel View Synthesis. We qualitatively compare the novel-view synthesis of our model against the strongest feed-forward reconstruction baselines. The results in Fig. 15 show that DepthSplat suffers from severe distortions, while EVolSplat produces partial geometric distortions and detail degradation. STORM and Omni-Scene exhibit better visual quality than DepthSplat and EVolSplat, but still contain a small number of artifacts. In contrast, our model shows fewer distortions and better preserves texture detail (e.g., the pose of trees).

Night simulation. Our model can predict intrinsic primitives and light context in a feed-forward manner. By setting the environmental illumination and sunlight intensity to zero and introducing point light sources, our model can simulate low-light nighttime street environments, as Fig. 16 shows. The position, colour and intensity of every point light used here are *specified by hand at inference time*: no light is fitted during training and none is recovered from the input images, so this extension shows that the renderer can consume such lights, not that the model can discover them. The formulation of the point light source is shown as follows, where we extend the PBR equation.

$$\mathbf{r}_o(\boldsymbol{\omega}_o, \boldsymbol{\omega}_{\text{sun}}, \mathbf{x}) = \underbrace{\int_\Omega \mathbf{f}(\boldsymbol{\omega}_o, \boldsymbol{\omega}_i, \mathbf{x})\, \mathbf{r}_{\text{env}}(\boldsymbol{\omega}_i, \mathbf{x})\, (\boldsymbol{\omega}_i \cdot \mathbf{n})_+ \, d\boldsymbol{\omega}_i}_{\text{Environment illumination term}} + \underbrace{\mathbf{f}(\boldsymbol{\omega}_o, \boldsymbol{\omega}_{\text{sun}}, \mathbf{x})\, \mathbf{r}_{\text{sun}}(\mathbf{x})\, (\boldsymbol{\omega}_{\text{sun}} \cdot \mathbf{n})_+}_{\text{Direct sunlight term}}$$
$$+ \underbrace{\sum_{k=1}^{K} V_k(\mathbf{x})\, \mathbf{f}(\boldsymbol{\omega}_o, \boldsymbol{\omega}_k, \mathbf{x})\, \tilde{\mathbf{r}}_k(\mathbf{x})\, (\boldsymbol{\omega}_k \cdot \mathbf{n})_+}_{\text{Point-light illumination term}}. \tag{10}$$

$$\boldsymbol{\omega}_k = \frac{\mathbf{p}_k - \mathbf{x}}{\|\mathbf{p}_k - \mathbf{x}\|}, \qquad d_k = \|\mathbf{p}_k - \mathbf{x}\|. \tag{11}$$

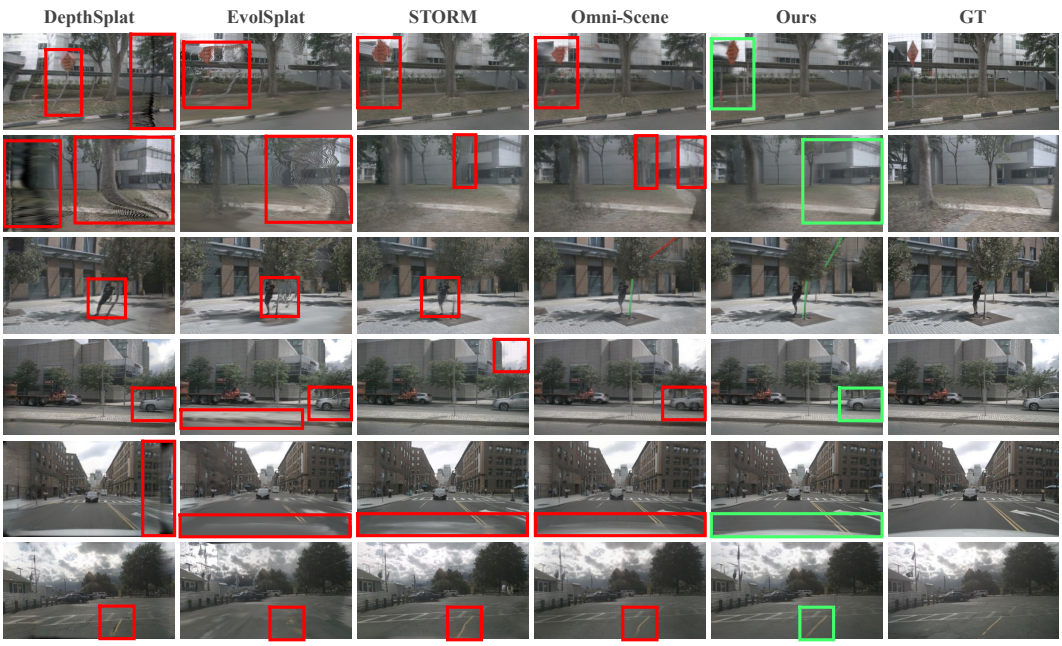

Figure 15: **Novel view synthesis comparison.** Qualitative comparison of our method with the strongest feed-forward reconstruction baselines.

$$\tilde{\mathbf{r}}_k(\mathbf{x}) = \frac{\bar{\mathbf{r}}_k(\mathbf{x})}{1 + \bar{\mathbf{r}}_k(\mathbf{x})/10}, \qquad \bar{\mathbf{r}}_k(\mathbf{x}) = \frac{I_k \mathbf{c}_k}{1 + 0.09 d_k + 0.032 d_k^2}. \tag{12}$$

Here, $\mathbf{x}$ denotes the surface point, $\mathbf{n}$ is the surface normal, and $\boldsymbol{\omega}_o$ is the outgoing viewing direction. $\mathbf{f}(\boldsymbol{\omega}_o, \boldsymbol{\omega}, \mathbf{x})$ denotes the BRDF, which consists of a Lambertian diffuse term and a GGX specular term in our implementation. The first term integrates the environment illumination $\mathbf{r}_{\text{env}}$ over the upper hemisphere Ω. The second term models direct sunlight from direction $\boldsymbol{\omega}_{\text{sun}}$ with radiance $\mathbf{r}_{\text{sun}}(\mathbf{x})$. The last term accumulates the contribution of K point lights. For the k-th point light, $\mathbf{p}_k$, I_k, and $\mathbf{c}_k$ denote its position, scalar intensity, and RGB color, respectively. $\boldsymbol{\omega}_k$ is the normalized direction from $\mathbf{x}$ to the point light, d_k is the corresponding distance, and $V_k(\mathbf{x}) \in [0, 1]$ is the learned point-light visibility. The point-light radiance is attenuated by a quadratic distance falloff and further compressed by a soft near-field stabilization term.

Illumination-aware snow rendering. Fig. 17 shows more visual results of snow weather simulation. Unlike screen-space snow overlays or unlit particles, our snow particles are inserted into the Gaussian rendering pipeline as physically meaningful anisotropic 3D Gaussians. Each snow particle carries opacity, color, normal, and material roughness, allowing both airborne snowflakes and accumulated snow to interact with scene illumination.

The particle normal is determined differently for falling and settled snow. For falling snowflakes, the local frame is aligned with the velocity direction, so their highlights and brightness vary dynamically with their motion. For accumulated snow, the Gaussian is flattened along the surface normal, and the normal is taken as the shortest axis of the anisotropic Gaussian. Consequently, airborne snow produces view- and light-dependent sparkling effects, while settled snow forms softly illuminated surface patches that respond consistently to sunlight, ambient illumination, and nearby point lights.

To maintain temporal coherence while enabling physically plausible snowfall, we simulate snow particles in 3D space using a unified Gaussian particle system. At each time step, falling particles are advected under gravity, with stochastic horizontal perturbations applied to mimic wind-induced variations. The basic motion update follows a semi-implicit Euler integration:

$$\mathbf{v}_i^{t+\Delta t} = \mathbf{v}_i^t + \mathbf{g}\Delta t, \qquad \mathbf{x}_i^{t+\Delta t} = \mathbf{x}_i^t + \mathbf{v}_i^{t+\Delta t}\Delta t. \tag{13}$$



Figure 16: **Nighttime street simulation.** Our model enables feed-forward prediction of intrinsic primitives and light context, allowing controllable night simulation by disabling environment and sunlight illumination and introducing point light sources. The extended PBR formulation models the accumulated contribution of multiple point lights with learned visibility, distance attenuation, and near-field radiance stabilization, producing realistic low-light street illumination.

Here, $\mathbf{x}_i^t$ and $\mathbf{v}_i^t$ denote the position and velocity of the i-th snow particle at time t, respectively. Δt is the simulation time step and $\mathbf{g}$ is the gravity vector. This update produces temporally coherent trajectories for airborne snow while keeping the simulation lightweight.

After updating particle positions, we test each falling particle against the scene geometry to determine whether deposition occurs. Specifically, collisions are detected using both the ground plane and a voxelized occupancy field constructed from the Gaussian scene. When a particle intersects the occupied voxel set or falls below the ground plane, it is converted into a settled anisotropic Gaussian attached to the contacted surface:

$$\mathbf{x}_i^{t+\Delta t} \rightarrow \mathbf{x}_i^{\text{set}}, \quad \text{if } \mathbf{x}_i^{t+\Delta t} \in \mathcal{O} \text{ or } z_i^{t+\Delta t} < z_{\text{ground}}. \tag{14}$$

In this expression, $\mathcal{O}$ denotes the occupied voxel set and z_{ground} is the ground-plane height. The resulting settled position $\mathbf{x}_i^{\text{set}}$ is then used to instantiate a persistent snow Gaussian on the surface. Particle recycling keeps the total number of Gaussians fixed, enabling continuous snowfall and gradual accumulation with bounded computational cost.

Controllable sky illumination. Because illumination is disentangled, we are able to explicitly separate and control the uniform global illumination in the environment and the intensity of sunlight. Consequently, we can simulate the warm orange-red global illumination at dusk, as well as the cool-toned global illumination under overcast conditions or in the early morning, as illustrated in Fig. 18. Because our model can independently disentangle the sky region, we can use texture maps to simulate realistic dusk and overcast effects. This constitutes a significant advantage over existing models, which often learn the sky as a Gaussian point cloud.



Figure 17: **More visual results of snow weather simulation.** Our method represents falling and accumulated snow as physically meaningful anisotropic 3D Gaussians within the rendering pipeline. 3D particle simulation with gravity, geometry-aware collision detection, and particle recycling enables temporally coherent snowfall and gradual snow accumulation under scene illumination.

More asset insertion results. We provide additional asset insertion results in Fig. 19 and Fig. 20. In Fig. 5 the *Probe ball* column is the single-frame environment probe predicted by DiffusionLight-Turbo (Chinchuthakun et al., 2026), drawn as a mirrored sphere; the asset is relit from that probe and then alpha-composited, as in the other baselines. In these examples the 3DRealCar assets baked through representation alignment are inserted under our predicted light context and look better integrated than the alternatives. The relighting-based and 2D harmonization pipelines receive the same assets but must composite them without a reconstructed scene, and, as appendix A.4 states, that input disadvantage also penalises them on ground contact. Combining feed-forward reconstruction with an intrinsic representation therefore yields better-integrated insertion in these examples, which is not a general claim of photorealism.

Zero-shot results. Fig. 21 presents zero-shot results on two kinds of out-of-distribution input: web-collected panoramic images and multi-view images synthesised by generative models. Specifically, the input images come from four sources: (1) anime-style and real-scene panoramas retrieved through image search engines such as Bing; (2) anime- and Minecraft-style results obtained by applying style transfer to nuScenes images with the Gemini 3 Pro Image model (Google, 2025); (3) realistic street panoramas generated by the panorama generator DiT360 (Feng et al., 2025a); and (4) multi-view generation results from our fine-tuned SD-Turbo model (Sauer et al., 2024). Although these inputs deviate from the camera configuration of nuScenes, our model remains partly compatible with cameras beyond the training distribution: for panoramas, we place six cameras at a shared position at the scene centre and feed them to the model, and the model still produces plausible results despite the shifted camera layout. Moreover, although the generated or collected images lack solar-direction annotations, no intrinsic head or encoder feature consumes the solar direction (Sec. 3.2), so the predicted materials are unaffected by the missing annotation; a specified or estimated sun direction is needed only at relighting time and acts purely through the rendering path.



Figure 18: By disentangling illumination, our model enables independent control over global environment illumination, allowing realistic simulation of warm dusk lighting and cool overcast or early-morning conditions. In addition, the independently disentangled sky region can be replaced with texture maps to produce coherent dusk and overcast appearances, avoiding the limitations of methods that represent the sky as a Gaussian point cloud.

A.9 Broader Impact

The proposed Lumen-GS is a single-timestamp, light-disentangled feed-forward approach for street-scene reconstruction. By exposing intrinsic scene properties and environmental illumination for explicit control, it makes a class of scene manipulations available without per-scene optimization, which we expect to be useful for 3D simulation, digital twins, and spatial intelligence. One of its most promising applications is in autonomous driving, where Lumen-GS can serve as a scalable, physically-grounded data generation engine. By synthesizing critical edge-case scenarios—such as adverse weather conditions or nighttime driving—it can enhance the robustness of perception systems without the need for costly and potentially hazardous real-world data collection in dangerous environments.

However, the ability to reconstruct and explicitly manipulate realistic street scenes warrants careful consideration. Like many advanced rendering and generative technologies, Lumen-GS could potentially be misused to create fabricated dashcam footage or seamlessly insert non-existent objects into real-world locations, which could be exploited to spread disinformation or manipulate evidence. Furthermore, when applied to safety-critical domains like autonomous driving, there is a risk associated

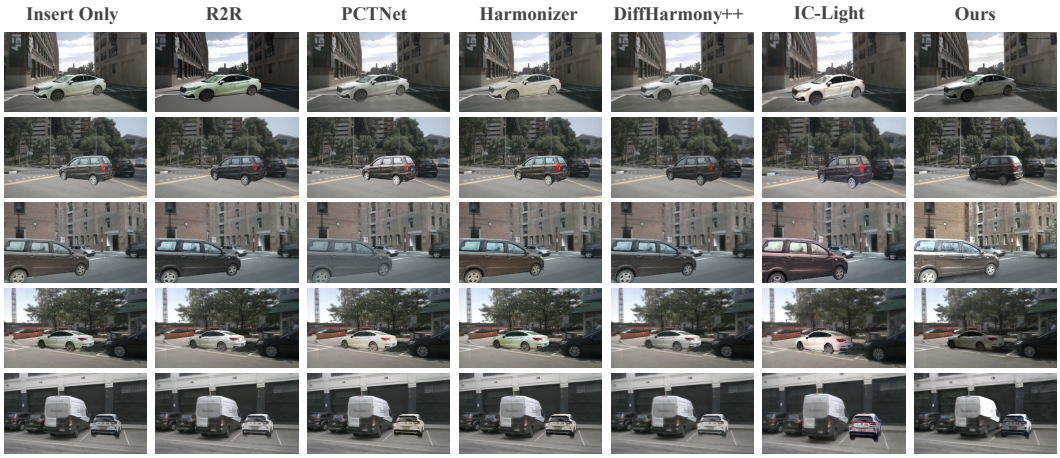

Figure 19: Additional asset insertion results comparing 2D harmonization methods and Lumen-GS. Lumen-GS leverages the predicted light context to produce visually harmonious and geometrically consistent results. In contrast, 2D harmonization methods struggle to capture scene geometry and illumination, leading to less realistic object insertion.



Figure 20: Additional asset insertion results comparing relighting-based insertion and Lumen-GS. By baking 3DRealCar assets through representation alignment and inserting them into feed-forward reconstructed scenes, Lumen-GS produces visually harmonious and geometrically consistent results.

with the "simulation-to-reality" gap; perception models trained heavily on synthetic data might inherit unseen biases, artifacts, or physical inaccuracies from the reconstruction process. To mitigate these risks, we advocate for the rigorous safeguards and closed-loop testing of synthesized data before its use in real-world autonomous systems.

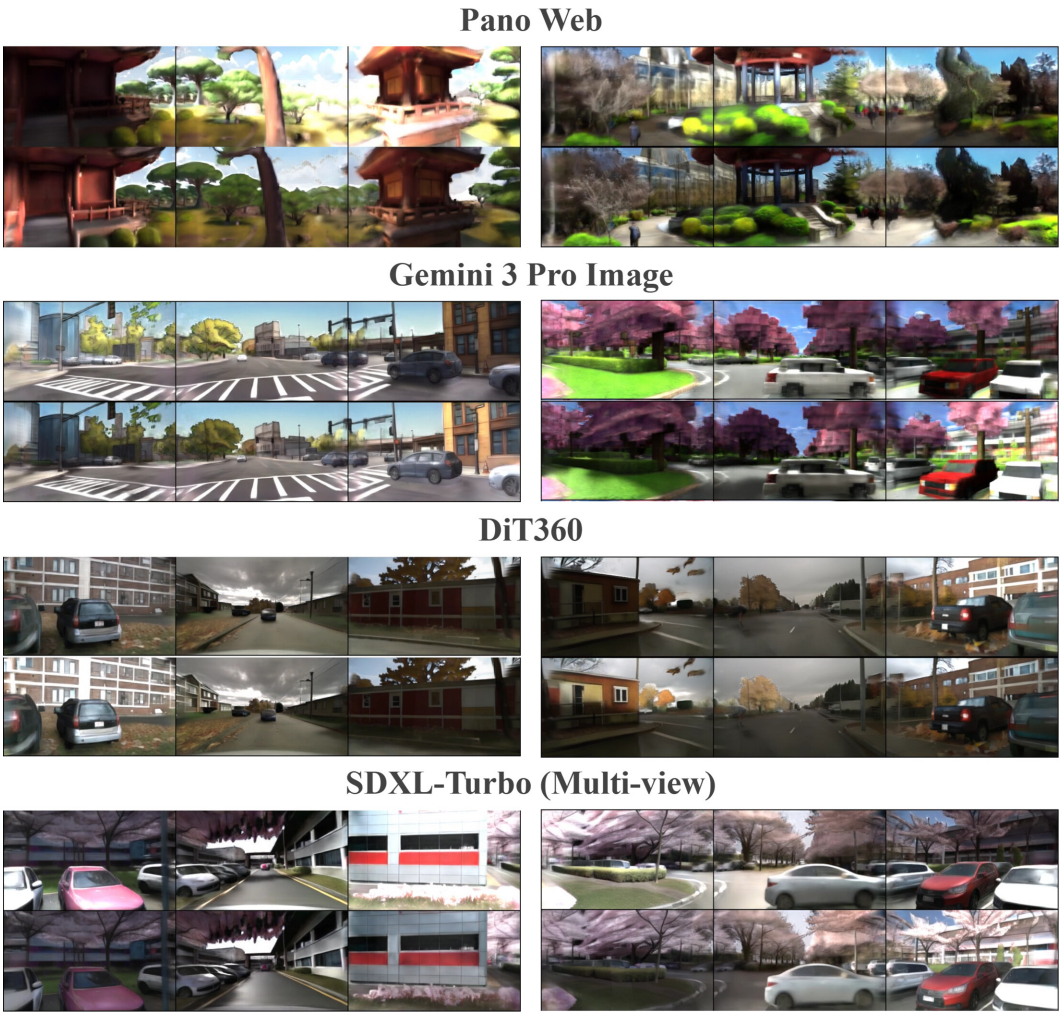

Figure 21: Zero-shot results. Inputs are panoramas from the web, panoramas generated by DiT360 and Gemini, and multi-view images from our fine-tuned multi-view SD-Turbo.