LoFT-GS: Factorized Local Transport for Relightable Gaussian Splatting

Anonymous authors
Paper under double-blind review

Abstract

Relightable Gaussian splatting combines explicit reconstruction with fast rendering under novel views and illumination. However, a single visibility value per Gaussian cannot preserve the spatial distribution of occlusion within its footprint. When a cast-shadow boundary crosses a Gaussian, illuminated and occluded regions are collapsed into a shared shadow response. We introduce LoFT-GS, a locally factorized relighting representation that addresses this limitation by recording visibility on a local texture grid within each 2D Gaussian during light-space splatting. A range-conditioned shadow formulation combines this local visibility with a Gaussian-level range predictor. More generally, LoFT-GS separates spatially varying relighting factors from shared Gaussian-level computation. Local texels represent visibility and diffuse appearance and modulate specular amplitude and sharpness, while geometry, base directional responses, and neural predictions remain shared per Gaussian. Experiments across diverse scenes demonstrate improved relighting fidelity over the evaluated baselines.

1 Introduction

Reconstructing relightable 3D objects from multi-view photographs enables digital assets to be rendered from novel viewpoints and under novel illumination, supporting tasks in embodied AI, autonomous driving simulation, virtual reality (VR), and other interactive applications.

Gaussian splatting (Kerbl et al., 2023) provides an explicit and efficiently rasterized representation for such assets. However, unlike fixed-light view synthesis, relightable reconstruction must capture how appearance varies across an object as the lighting changes. Cast-shadow boundaries move across surfaces, while highlights change in position, intensity, and extent, which vanilla Gaussian representations struggle to model.

Recent point-light methods enrich the response of each Gaussian (e.g., GS^3, RNG, SSD-GS) (Bi et al., 2024; Fan et al., 2025; Zheng et al., 2026). Despite their different formulations, they retain a common spatial granularity: a Gaussian remains the indivisible unit of relighting. However, aggregating visibility into a single value per Gaussian discards the spatial distribution of occlusion within its footprint. Consider a cast shadow boundary cutting through a Gaussian footprint: if the left half is occluded and the right half is illuminated, a single shared shadow value can only capture the average visibility. It inherently loses the spatial structure, failing to distinguish this state from a uniformly dimmed footprint. As illumination changes, the boundary can move across different locations within the same footprint, making this lost intra-Gaussian structure particularly relevant to relighting.

LoFT-GS addresses this limitation by separating spatially varying relighting factors from computation that can remain shared over a Gaussian footprint. We attach a compact local grid to each 2D Gaussian while retaining its geometry, base directional responses, and neural predictions at the Gaussian level. During light-space splatting, we record visibility at local grid locations rather than averaging it over the entire Gaussian, preserving illuminated and occluded regions within the same footprint. A range-conditioned shadow formulation then uses a Gaussian-level predictor to calibrate this local visibility within a bounded response range.

Reflectance follows the same local/shared organization. Local texels store diffuse albedo and modulate the amplitude and sharpness of a shared specular response, while base directional responses

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.

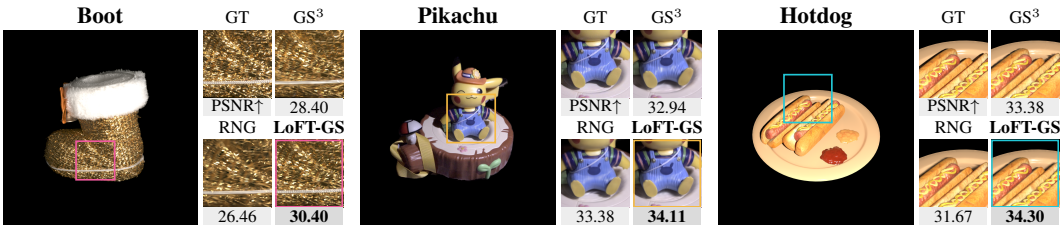

Figure 1: **Locally Factorized Relightable Gaussian Splatting (LoFT-GS).** From left to right, we illustrate three representative challenges that vary at a sub-Gaussian level: fine-scale diffuse textures, specular highlights, and cast shadows. Gaussian-level relighting can blur these high-frequency variations, whereas our locally factorized representation explicitly resolves them within each footprint.

and neural predictions are evaluated once per Gaussian and reused across its texels. Together with local visibility, this yields a two-level representation combining spatially varying relighting factors with shared Gaussian-level computation.

Our main contributions are:

- We introduce a two-level relighting representation that places spatially varying visibility and reflectance factors on local texel grids while sharing directional and neural computation at the Gaussian level.
- We design a range-conditioned shadow formulation that combines texel-level light-space visibility with a Gaussian-level range predictor, preserving spatially varying shadows without per-texel neural evaluation.
- Extensive evaluations across 21 diverse scenes demonstrate that LoFT-GS achieves state-of-the-art relighting fidelity, significantly outperforming existing baselines.

2 Related Work

2.1 Relightable Object Reconstruction

Traditional appearance capture combines explicit surface geometry with spatially varying reflectance, or densely samples reflectance fields under controlled viewing and lighting conditions (Debevec et al., 2000; Dorsey et al., 2007). Neural inverse-rendering methods such as NeRFactor, Neural-PIL, and TensoIR jointly estimate geometry, materials, and illumination from images (Zhang et al., 2021; Boss et al., 2021; Jin et al., 2023). For controlled object relighting, Deferred Neural Lighting, NRHints, and Relightable Neural Assets learn view- and light-dependent appearance from multi-view observations (Gao et al., 2020; Zeng et al., 2023; Mullia et al., 2025). Recent methods use generative priors: IllumiNeRF reconstructs a radiance field from diffusion-relit images, while LightSwitch performs material-guided multi-view relighting (Zhao et al., 2024; Litman et al., 2025).

2.2 Gaussian Relighting and Inverse Rendering

GaussianShader introduces Gaussian-based shading functions for reflective appearance, Relightable 3D Gaussians combines BRDF decomposition with point-based ray tracing, and GS-IR jointly estimates geometry, materials, and environment lighting (Jiang et al., 2024; Gao et al., 2024; Liang et al., 2024). GI-GS models indirect illumination with deferred shading and lightweight path tracing, while LumiGauss and ReCap address relighting from unconstrained or cross-environment captures (Chen et al., 2025; Kaleta et al., 2025; Li et al., 2025). IRGS further introduces 2D Gaussian ray tracing for inter-reflection, and GeoSplatting uses mesh-guided geometry to improve normal estimation and light transport (Gu et al., 2025; Ye et al., 2025). A second line learns relightable objects from controlled point-light or OLAT observations. PRTGaussian represents radiance transfer with spherical harmonics, OLAT Gaussians learns incident-illumination and scattering functions, and BiGS uses bidirectional Gaussian primitives for surface and volumetric materials (Zhang et al., 2024; Kuang et al., 2024; Liu et al., 2025). GS^3 combines spatial and angular Gaussians with light-space shadow

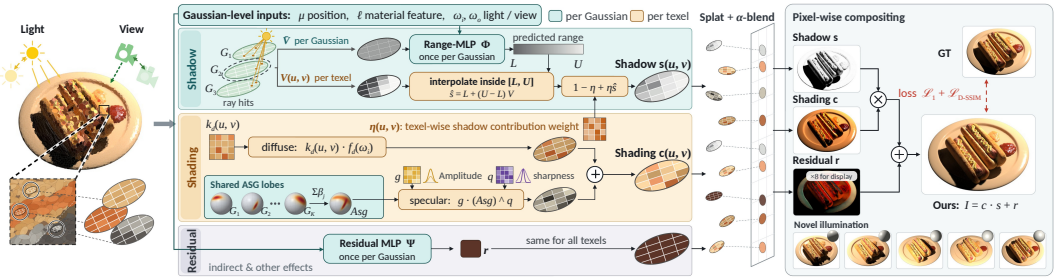

Figure 2: **Overview of LoFT-GS.** Shared computation (Range-MLP Φ, ASG response, residual MLP Ψ) is evaluated once per Gaussian, while texels store local visibility, shadow weight η, diffuse albedo, and specular amplitude/sharpness. The resulting shadow, shading, and residual texels are splatted and composited as $I = c \cdot s + r$. Residual shown $\times 8$; spheres indicate light direction.

splatting, whereas RNG decodes radiance from view-, light-, and shadow-conditioned Gaussian features (Bi et al., 2024; Fan et al., 2025). SSD-GS further decomposes each Gaussian's shading into diffuse, specular, shadow, and subsurface-scattering components (Zheng et al., 2026).

2.3 Textured and Spatially Varying Gaussians

Texture-GS represents view-independent appearance with a learned surface texture, while GStex and Textured Gaussians attach local texture maps to Gaussian primitives to decouple appearance detail from geometric complexity (Xu et al., 2024; Rong et al., 2025; Chao et al., 2025). SVG-IR represents spatially varying material and normal attributes within Gaussian surfels for inverse rendering, while evaluating visibility at the Gaussian level (Sun et al., 2025). Recent work also adapts texture resolution or sampling density: Content-Aware Texturing changes per-primitive texture resolution during optimization, while FACT-GS allocates texture sampling density according to local visual frequency (Papantonakis et al., 2025; Xie et al., 2026). LoFT-GS introduces a two-level representation that combines local visibility, diffuse, and specular factors with shared Gaussian-level directional and neural computation. Unlike learned appearance textures, the local visibility field is recomputed for the current illumination.

3 Method

3.1 Preliminaries: 2D Gaussian Splatting

We represent the object using 2D Gaussian primitives (Huang et al., 2024). Each Gaussian is defined by a center $\boldsymbol{\mu}$, two orthonormal tangent vectors $\mathbf{t}_1, \mathbf{t}_2$, in-plane scales s_1, s_2, and opacity o. These quantities define a local coordinate system on the Gaussian plane, whose geometric normal is $\mathbf{n} = \mathbf{t}_1 \times \mathbf{t}_2$. During rasterization, the intersection of a camera ray with this plane determines the local sampling position and the corresponding Gaussian weight.

For each pixel, the contributing Gaussians are composited in front-to-back order:

$$C_{\text{pixel}} = \sum_{i=1}^{N} T_i \alpha_i \mathbf{c}_i, \qquad T_i = \prod_{j=1}^{i-1} (1 - \alpha_j), \tag{1}$$

where α_i is the Gaussian weight scaled by its opacity, $\mathbf{c}_i$ is the color of Gaussian i, and T_i is the accumulated transmittance. The same compositing rule is applied to the rendering components defined below.

3.2 Shading Formulation

Following GS^3 (Bi et al., 2024), we decompose our relighting model into four components: diffuse reflection, specular reflection, shadows, and other effects. This Gaussian-level shading model pro-

vides a stable foundation for coarse appearance, while the texel-level representation introduced in Sec. 3.3 effectively captures local variations.

Diffuse and specular terms. For incident direction $\boldsymbol{\omega}_i$ and viewing direction $\boldsymbol{\omega}_o$, the direct-shading response is

$$\mathbf{f}(\boldsymbol{\omega}_i, \boldsymbol{\omega}_o) = \mathbf{k}_\mathrm{d} f_\mathrm{d}(\boldsymbol{\omega}_i) + \mathbf{k}_\mathrm{s} f_\mathrm{s}(\boldsymbol{\omega}_i, \boldsymbol{\omega}_o), \tag{2}$$

where $\mathbf{k}_\mathrm{d}$ and $\mathbf{k}_\mathrm{s}$ are the diffuse and specular albedos, and f_d and f_s are their corresponding appearance functions, including incident-angle weighting. The diffuse term describes view-independent reflection, whereas the specular term models directional highlights.

Shadow term. To account for occlusion, we estimate geometric visibility by splatting the scene from the point-light position. This visibility measures how much incident light reaches a Gaussian. Our shadow formulation uses this geometric estimate to predict a shadow factor that modulates direct shading, as detailed in Sec. 3.3. A factor of one leaves the shading unattenuated; smaller values produce stronger shadows.

Other effects. To account for appearance effects not captured by direct shading, a shared MLP Ψ predicts an additive RGB response for each Gaussian, conditioned on its position, a learned material feature $\boldsymbol{\ell}$, and light and view directions. This residual can accommodate effects such as indirect illumination without introducing an explicit multi-bounce transport model. The network is specified in Appendix A.3.

3.3 Local Texture Factorization

LoFT-GS separates quantities that require spatial variation within a Gaussian from directional and neural computation that can be shared across its footprint. The local grid represents the former, while the latter remains Gaussian-level. A 2D Gaussian represents a small planar surface patch. We place a local texture grid directly on this patch, using the Gaussian's own two-dimensional coordinate system. Thus, the grid is attached to the represented surface patch without requiring a separate mesh or texture atlas. The intersection of a light or camera ray with the plane determines its local texture coordinates (u, v), used for visibility accumulation and appearance lookup, respectively. Their construction is detailed in Appendix A.1.

3.3.1 Shadow term.

We construct the shadow term in two stages. First, we measure visibility at both Gaussian and texel levels. Second, we refine these geometric estimates into the shadow factor that modulates the rendered appearance.

Stage 1: multi-level geometric visibility. We splat the scene from the point-light position and accumulate transmittance along the light rays in front-to-back order. For a particular Gaussian with opacity o, let T_m be the transmittance reaching it at light-view sample m, and w_m its projected Gaussian weight. A Gaussian-level estimate combines all of its samples into one visibility value:

$$\bar{V} = \frac{\sum_m o w_m T_m}{\sum_m o w_m} = \frac{\sum_m w_m T_m}{\sum_m w_m}. \tag{3}$$

However, when a shadow boundary crosses the Gaussian, this average mixes illuminated and occluded locations, so the result no longer indicates where the occlusion occurs.

Instead, we use the light-ray intersection to assign each sample to a local texel and accumulate visibility separately in that texel (Figure 3). If $\mathcal{B}(u, v)$ is the set of samples assigned to the texel at (u, v), its visibility is

$$V(u, v) = \frac{\sum_{m \in \mathcal{B}(u,v)} w_m T_m}{\sum_{m \in \mathcal{B}(u,v)} w_m}. \tag{4}$$

The resulting map $V(u, v) \in [0, 1]$ therefore retains the spatial distribution of visibility within the Gaussian. Note that we retain $\bar{V}$ as a Gaussian-level summary; Stage 2 combines it with the detailed texel map $V(u, v)$.

Stage 2: Range-MLP shadow prediction. To improve shadow quality beyond the raw geometric estimate, we learn a shared Gaussian-level calibration with a shared MLP Φ. For each Gaussian, the MLP uses its average visibility, position, material feature, and light/view directions to determine the appropriate shadow attenuation. This learned correction can compensate for inaccuracies in the Gaussian reconstruction.

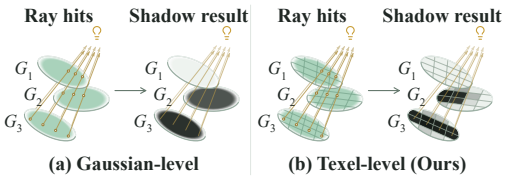

Figure 3: **Gaussian-level vs. local visibility.**

However, assigning one corrected value to the entire Gaussian would discard the visibility differences between its texels. Evaluating the MLP separately at every texel could provide local corrections, but would repeat the neural computation.

For shadow evaluation over N Gaussians with T texels each, let C_Φ denote the cost of one MLP evaluation. A texel-wise MLP requires $\mathcal{O}(NTC_\Phi)$ work, whereas one prediction per Gaussian followed by scalar operations at its texels requires $\mathcal{O}(NC_\Phi + NT)$.

Our Range-MLP instead predicts a shadow range once per Gaussian. Each texel uses its own visibility to select a value within this range, preserving local shadow structure while sharing the neural prediction.

Specifically, Φ predicts two scalars:

$$(L, W) = \Phi(\bar{V}, \boldsymbol{\omega}_i, \boldsymbol{\omega}_o; \boldsymbol{\mu}, \boldsymbol{\ell}), \qquad U = L + (1 - L)W, \tag{5}$$

where $\boldsymbol{\omega}_i$ and $\boldsymbol{\omega}_o$ are the incident-light and viewing directions at Gaussian center $\boldsymbol{\mu}$, and $\boldsymbol{\ell}$ is its learned material feature. The input $\bar{V}$ provides its coarse geometric visibility. The outputs satisfy $L, W \in [0, 1]$. The first output gives a lower shadow factor L. The second chooses how far the upper shadow factor U lies between L and one, ensuring $0 \le L \le U \le 1$. For each texel, we interpolate between these responses according to its measured visibility:

$$\widehat{s}(u, v) = L + (U - L)V(u, v) = L + (1 - L)WV. \tag{6}$$

A reconstructed surface can be represented by several overlapping Gaussian layers, which can introduce unnecessary darkening due to mutual occlusion between layers, even where the represented surface has no external occluder. We therefore equip each Gaussian with a texel-wise shadow contribution weight $\eta(u, v) \in [0, 1]$, allowing such contributions to be suppressed locally. The final shadow factor interpolates between the predicted factor and the unoccluded value one:

$$s(u, v) = 1 - \eta(u, v) + \eta(u, v)\widehat{s}(u, v). \tag{7}$$

Here, $\eta = 0$ contributes no shadow and $\eta = 1$ uses the predicted factor $\widehat{s}$. Thus, Gaussian-level context calibrates the shadow range, texel visibility preserves the spatial boundary, and the local weight controls the contribution of that prediction. The MLP architecture is given in Appendix A.3.

3.3.2 Diffuse and Specular terms.

Diffuse term. We adopt a common cosine-weighted Lambertian diffuse model with the smooth incident-angle weighting of GS^3. Specifically, its response depends solely on the angle between the shading normal and the incident-light direction, being strongest when aligned with the normal and decreasing toward grazing incidence. The RGB albedo $\mathbf{k}_\mathrm{d}$ controls the diffuse color. We store it as a local texture $\mathbf{k}_\mathrm{d}(u, v)$, so different locations within a Gaussian can have different diffuse colors. The smooth diffuse function is given in Eq. 15.

Specular term. Specular reflection is concentrated around particular directions: as the light or camera moves, a surface region enters or leaves a highlight. We represent this angular response using anisotropic spherical Gaussians (ASGs) (Xu et al., 2013). Each basis function describes a smooth peak on the sphere of directions, called a lobe. Its two angular widths can differ, allowing an elongated highlight instead of only a circular one.

For the current light and view, the basis functions are evaluated at the half vector $\mathbf{h} = (\boldsymbol{\omega}_i + \boldsymbol{\omega}_o)/\|\boldsymbol{\omega}_i + \boldsymbol{\omega}_o\|$. Their weighted combination gives the directional response

$$A(\boldsymbol{\omega}_i, \boldsymbol{\omega}_o) = F(\boldsymbol{\omega}_i, \mathbf{h}) \sum_{j=1}^{K} \beta_j G_j(\mathbf{h}), \tag{8}$$

where G_j is an ASG basis response and β_j is its nonnegative learned mixture weight for the Gaussian. The Fresnel factor F accounts for the increase in reflectivity at grazing incidence. Both F and A are scalars. The individual basis functions and Fresnel term are detailed in Appendix A.2.

To introduce local specular variation, one could store an RGB specular albedo at every texel, as for the diffuse term, or allow local shading normals and additional ASG shape parameters. These choices introduce different costs: local RGB albedos add color degrees of freedom, while varying normals or angular shapes requires evaluating different directional responses within a Gaussian. We instead share $\mathbf{k}_\mathrm{s}$ and the directional model across the Gaussian, and concentrate the local parameters on highlight amplitude and sharpness.

Each texel has a positive scalar amplitude parameter $g(u, v)$ and a positive sharpness parameter $q(u, v)$:

$$f_\mathrm{s}(\boldsymbol{\omega}_i, \boldsymbol{\omega}_o; u, v) = \pi f_\mathrm{d}(\boldsymbol{\omega}_i)\, g(u, v)\, A(\boldsymbol{\omega}_i, \boldsymbol{\omega}_o)^{q(u,v)}. \tag{9}$$

The amplitude scales the highlight, while the exponent concentrates or broadens the existing angular response. The factor πf_d supplies the same smooth incident-angle weighting as in the diffuse term. The two-level structure thus assigns directional highlight shape to the shared Gaussian model and spatial highlight variation to its texels, allowing highlight amplitude and sharpness to vary locally while reusing the same evaluated $A(\boldsymbol{\omega}_i, \boldsymbol{\omega}_o)$ for all texels of the Gaussian.

3.4 Rendering and Optimization

Rendering. For each light-view pair, we evaluate the ASG response and both MLPs once per Gaussian, then combine them with the local factors to obtain direct-shading colors, shadow factors, and additive RGB responses. We bilinearly sample the texel-varying quantities and composite each component using Eq. 1. Following the deferred composition of GS^3, the resulting images are combined pixelwise:

$$\mathbf{I} = \mathbf{I}_\mathrm{shading} S + \mathbf{I}_\mathrm{residual}. \tag{10}$$

The shading and sampling details are given in Appendix A.1. We use the standard 3DGS photometric loss (Kerbl et al., 2023), with optimization details in Appendix A.4.

4 Experiments

Datasets. The main comparison contains 16 scenes. The real subset includes four LightStage captures (Boot, Container, Fox, and Nefertiti) (Bi et al., 2024) and four NRHints objects (Cat, CupFabric, Fish, and Pikachu) (Zeng et al., 2023). The synthetic subset includes eight objects (AnisoMetal, Drums, FurBall, Hotdog, Lego, Translucent, MaterialBalls and Cup). A separate experiment on scattering scenes uses Bunny, Candle, Dragon, Soap, and Statue from SSS-GS Synthetic (Dihlmann et al., 2024).

Baselines and metrics. We compare against SVG-IR (Sun et al., 2025), GS^3 (Bi et al., 2024), RNG (Fan et al., 2025), and SSD-GS (Zheng et al., 2026) on the main benchmark. For the scattering scenes, we additionally compare with KiloOSF (Yu et al., 2023) and SSS-GS (Dihlmann et al., 2024), and integrate our method into SSD-GS. We report PSNR, SSIM (Wang et al., 2004), and LPIPS (Zhang et al., 2018), with equal weight given to each scene. Implementation details and baseline adaptations are provided in Appendix A.5.

4.1 Main Relighting Results

Table 1(a,b) compares LoFT-GS with point-light relighting methods and the spatially varying inverse-rendering baseline SVG-IR. LoFT-GS achieves the best average PSNR, SSIM, and LPIPS on both the real and synthetic subsets. Across the 16 main scenes, it obtains the highest PSNR on 14 scenes and improves the average PSNR by 1.05 dB over GS^3, 2.61 dB over RNG, and 1.15 dB over SSD-GS. RNG remains strongest on Cat and CupFabric. To evaluate whether the proposed representation also complements more complex light-transport models, we integrate it into SSD-GS and evaluate on five scattering scenes. As shown in Table 1(c), the resulting model achieves the best average PSNR, SSIM, and LPIPS, improving the mean PSNR from 38.13 dB to 38.63 dB over SSD-GS. This result indicates that the proposed local representation remains effective when

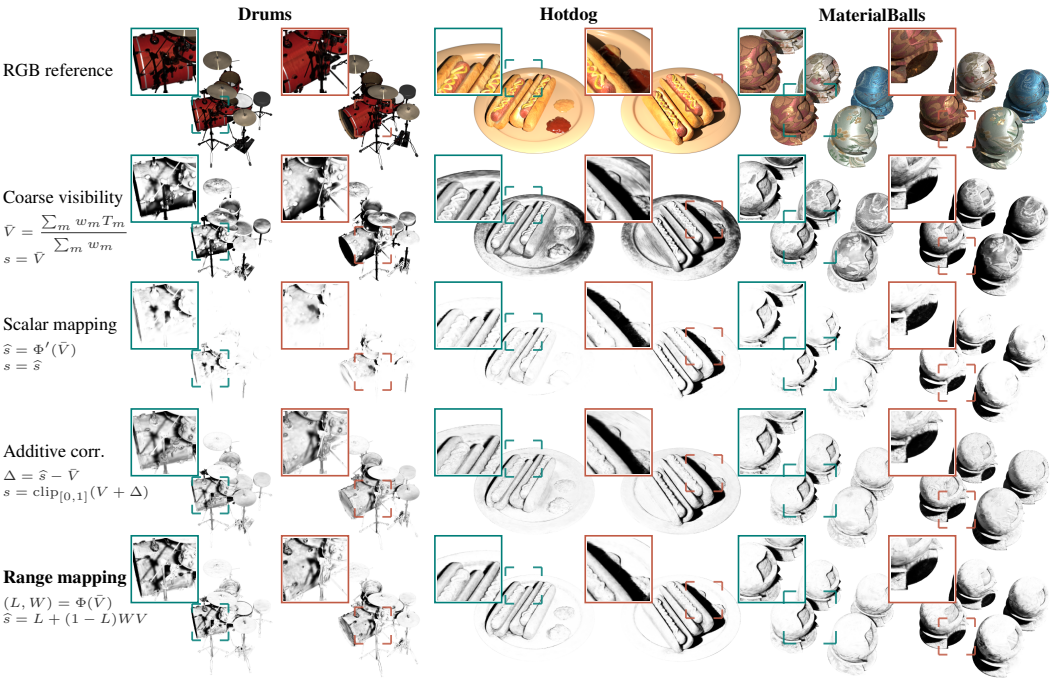

Figure 4: **Shadow formulation analysis.** We compare raw Gaussian-level visibility, GS^3-style scalar mapping, texel-level additive correction, and our range mapping. Insets highlight local shadow structure. Texel-level visibility restores fine occlusion details, while range mapping further suppresses excessive darkening and spiky shadow artifacts.

combined with an explicit scattering component. Figure 5 provides qualitative comparisons on representative scenes. LoFT-GS better preserves fine surface texture and compact specular highlights, while producing sharper and more spatially coherent shadow boundaries.

4.2 Ablation Studies

Shadow formulation. We evaluate the roles of texel-level visibility and learned range mapping in preserving shadow detail and correcting shadow intensity. Figure 4 compares four formulations: uncorrected Gaussian-level visibility, a Gaussian-level scalar prediction as in GS^3, and texel-level visibility with either additive correction or our Range-MLP. Scalar prediction reduces spurious darkening but also weakens fine shadows, as seen in Drums. Using texel-level visibility recovers additional shadow details, especially in Drums, and reduces the spiky boundary artifacts, especially in Hotdog and MaterialBalls. However, additive correction produces broad regions of excessive darkening. Our range mapping alleviates this darkening while retaining fine shadow structures. These results show that preserving local visibility and calibrating its response range address different aspects of shadow quality. Range-only fixes $\eta = 1$, while Gaussian η uses a single shared value per Gaussian. Moreover, as shown in Table 2(b), evaluating Φ independently at every texel provides no meaningful accuracy gain over our Gaussian-level Range-MLP, while reducing rendering speed from 81.8 to 28.3 FPS on Hotdog and from 81.9 to 38.1 FPS on Pikachu. This supports our local/shared factorization: visibility is spatially resolved, whereas neural prediction remains Gaussian-level.

Local amplitude and sharpness. Table 2(c) compares alternative local specular factors on AnisoMetal and Nefertiti. Removing local amplitude or sharpness reduces AnisoMetal PSNR from 28.84 dB to 28.58 and 28.70 dB, respectively. Predicting a local shading normal is substantially slower without improving accuracy, reaching 28.11 dB at 17.2 FPS on AnisoMetal and 38.59 dB at 64.1 FPS on Nefertiti, compared with 28.84/67.2 and 38.61/162.9 for Full. Local ASG shapes remain 0.29 and 0.06 dB below Full. To further explore its underlying mechanism, Figure A3 shows that amplitude alone adjusts peak magnitude but leaves the response broad, whereas sharpness alone narrows the response.

Table 1: **Main quantitative comparisons.** Results on real, synthetic, and scattering scenes. Best value and second value are highlighted.

(a) Main comparison on real scenes with SVG-IR(Sun et al., 2025), GS^3 (Bi et al., 2024), RNG (Fan et al., 2025) and SSD-GS (Zheng et al., 2026). Mean gives equal weight to each scene.

Scene	SVG-IR			GS^3			RNG			SSD-GS			**Ours**		
	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
LightStage															
Boot	20.46	0.8394	0.1803	28.40	0.8953	0.1265	26.46	0.8521	0.1403	28.93	0.8907	0.1322	**30.40**	**0.9157**	**0.1049**
Container	25.10	0.9403	0.0471	36.84	0.9766	0.0197	33.21	0.9589	0.0285	38.41	0.9803	0.0202	**39.08**	**0.9829**	**0.0166**
Fox	25.15	0.8737	0.1312	34.38	0.9239	0.0866	32.60	0.8915	0.0941	34.79	0.9205	0.0924	**35.40**	**0.9275**	**0.0849**
Nefertiti	26.49	0.9297	0.0723	36.69	0.9560	0.0547	22.40	0.8844	0.1091	35.38	0.9502	0.0615	**38.61**	**0.9662**	**0.0474**
NRHints															
Cat	15.23	0.7180	0.2934	27.69	0.9051	0.1613	**29.55**	**0.9113**	**0.1532**	27.29	0.8984	0.1671	27.74	0.9038	0.1558
CupFabric	24.44	0.9396	0.1121	36.86	0.9832	0.0685	**38.65**	**0.9857**	**0.0647**	36.31	0.9813	0.0733	38.00	0.9847	0.0655
Fish	23.38	0.8393	0.1997	31.11	0.9248	0.0983	31.01	0.9174	0.1138	31.16	**0.9256**	0.0977	**31.29**	0.9255	**0.0945**
Pikachu	19.49	0.9025	0.1258	32.94	0.9687	0.0782	33.38	0.9702	0.0767	32.54	0.9679	0.0788	**34.11**	**0.9709**	**0.0743**
Mean	22.47	0.8728	0.1452	33.11	0.9417	0.0867	30.91	0.9214	0.0976	33.10	0.9394	0.0904	**34.33**	**0.9472**	**0.0805**

(b) Main comparison on synthetic scenes with the same methods as the real-scene comparison.

Scene	SVG-IR			GS^3			RNG			SSD-GS			**Ours**		
	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
AnisoMetal	16.87	0.8709	0.0869	27.53	0.9516	0.0546	27.25	0.9537	0.0507	27.94	0.9519	0.0561	**28.84**	**0.9593**	**0.0451**
MaterialBalls	16.48	0.8158	0.1415	30.44	0.9475	**0.0783**	30.45	**0.9527**	0.0797	30.39	0.9431	0.0822	**30.49**	0.9490	0.0804
Drums	23.79	0.9398	0.0615	30.92	0.9744	0.0306	30.24	0.9751	0.0300	31.47	0.9757	0.0299	**32.18**	**0.9788**	**0.0269**
FurBall	19.65	0.8796	0.1506	35.77	0.9686	0.0741	36.45	0.9724	**0.0547**	35.54	0.9671	0.0868	**37.73**	**0.9741**	0.0561
Hotdog	17.11	0.8576	0.1218	33.38	0.9742	0.0351	31.67	0.9743	0.0332	33.24	0.9742	0.0353	**34.30**	**0.9776**	**0.0310**
Lego	16.19	0.7919	0.1454	31.02	0.9568	0.0496	28.25	0.9463	0.0567	30.90	0.9531	0.0535	**31.40**	**0.9608**	**0.0471**
Translucent	16.46	0.8884	0.0896	32.92	0.9759	0.0402	30.35	0.9742	0.0319	33.00	0.9768	0.0368	**33.94**	**0.9790**	**0.0318**
Cup	20.02	0.9311	0.0535	35.72	0.9896	**0.0241**	35.81	**0.9902**	0.0262	33.75	0.9853	0.0260	**35.97**	0.9896	0.0250
Mean	18.32	0.8719	0.1064	32.21	0.9673	0.0483	31.31	0.9674	0.0454	32.03	0.9659	0.0508	**33.11**	**0.9710**	**0.0429**

(c) Main comparison on scattering scenes with KiloOSF (Yu et al., 2023), SSS-GS (Dihlmann et al., 2024), and SSD-GS (Zheng et al., 2026). Ours augments SSD-GS with the proposed local representation.

Scene	KiloOSF			SSS-GS			SSD-GS			**Ours(SSD-GS)**		
	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
Bunny	25.35	0.9412	0.0662	36.11	0.9801	0.0333	37.06	0.9862	0.0187	**38.72**	**0.9891**	**0.0141**
Candle	18.84	0.9125	0.2602	38.71	0.9865	0.0244	40.13	0.9920	0.0128	**40.58**	**0.9925**	**0.0114**
Dragon	27.23	0.9121	0.1117	35.19	0.9634	0.0532	36.36	0.9783	0.0264	**36.86**	**0.9791**	**0.0239**
Soap	26.40	0.9129	0.1123	36.66	0.9761	0.0290	**41.19**	**0.9924**	**0.0086**	41.06	0.9913	0.0104
Statue	22.93	0.9036	0.1177	35.59	0.9773	0.0372	**35.92**	0.9836	0.0198	35.91	**0.9843**	**0.0197**
Mean	24.15	0.9165	0.1336	36.45	0.9767	0.0354	38.13	0.9865	0.0173	**38.63**	**0.9873**	**0.0159**

Table 2: **Ablation studies.** All variants are trained independently. Mean gives equal scene weight. Best and second-best results are highlighted using unrounded values.

(a) Mean performance after removing individual local factors. *All 16 scenes*

	w/o local diffuse			w/o local shadow			w/o local specular			**Full**		
	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
Real	34.27	0.9465	0.0811	34.21	0.9463	0.0810	34.20	0.9464	0.0812	**34.33**	**0.9472**	**0.0805**
Synthetic	32.89	0.9697	0.0448	32.53	0.9688	0.0451	32.51	0.9681	0.0475	**33.11**	**0.9710**	**0.0429**
Mean	33.58	0.9581	0.0629	33.37	0.9575	0.0631	33.36	0.9573	0.0643	**33.72**	**0.9591**	**0.0617**

(b) Shadow formulation

	Hotdog				*Pikachu*			
	PSNR↑	SSIM↑	LPIPS↓	FPS↑	PSNR↑	SSIM↑	LPIPS↓	FPS↑
w/o local shadow	33.89	0.9767	0.0323	**85.8**	33.92	0.9705	0.0750	76.5
Additive corr.	33.94	0.9752	0.0389	84.9	34.05	0.9707	0.0749	71.8
Range-only	34.02	0.9770	0.0316	84.4	33.97	0.9704	0.0751	68.1
Gaussian η	34.05	0.9769	0.0315	84.5	34.02	0.9705	0.0750	80.9
w/ per-texel Φ	34.27	0.9769	0.0319	28.3	**34.11**	0.9706	0.0747	38.1
Full	**34.30**	**0.9776**	**0.0310**	81.8	**34.11**	**0.9709**	**0.0743**	**81.9**

(c) Local specular

	AnisoMetal				*Nefertiti*			
	PSNR↑	SSIM↑	LPIPS↓	FPS↑	PSNR↑	SSIM↑	LPIPS↓	FPS↑
w/o local specular	28.46	0.9569	0.0475	67.2	38.41	0.9648	0.0476	**166.4**
w/o local amplitude	28.58	0.9572	0.0475	61.9	38.49	0.9651	0.0477	165.4
w/o local sharpness	28.70	0.9577	0.0466	62.2	38.48	0.9649	0.0480	165.9
w/ local normal	28.11	0.9547	0.0490	17.2	38.59	0.9657	**0.0471**	64.1
w/ local ASG shape	28.55	0.9580	0.0455	64.1	38.55	0.9653	0.0477	151.5
Full	**28.84**	**0.9593**	**0.0451**	**67.2**	**38.61**	**0.9662**	0.0474	162.9

4.3 RENDERING EFFICIENCY

Table 3 reports memory, frame rate, and Gaussian count. LoFT-GS renders at 77–136 FPS, compared with 15–55 FPS for RNG and 27–115 FPS for SSD-GS, and uses 27–50% fewer Gaussians than GS^3. Its specular and neural stages are faster than those of GS^3 on every scene, while shadow, compositing, and rasterization are slower; total frame time differs from GS^3 by -13% to $+13\%$. Overall, LoFT-GS increases intra-Gaussian relighting resolution while maintaining rendering efficiency comparable to GS^3.

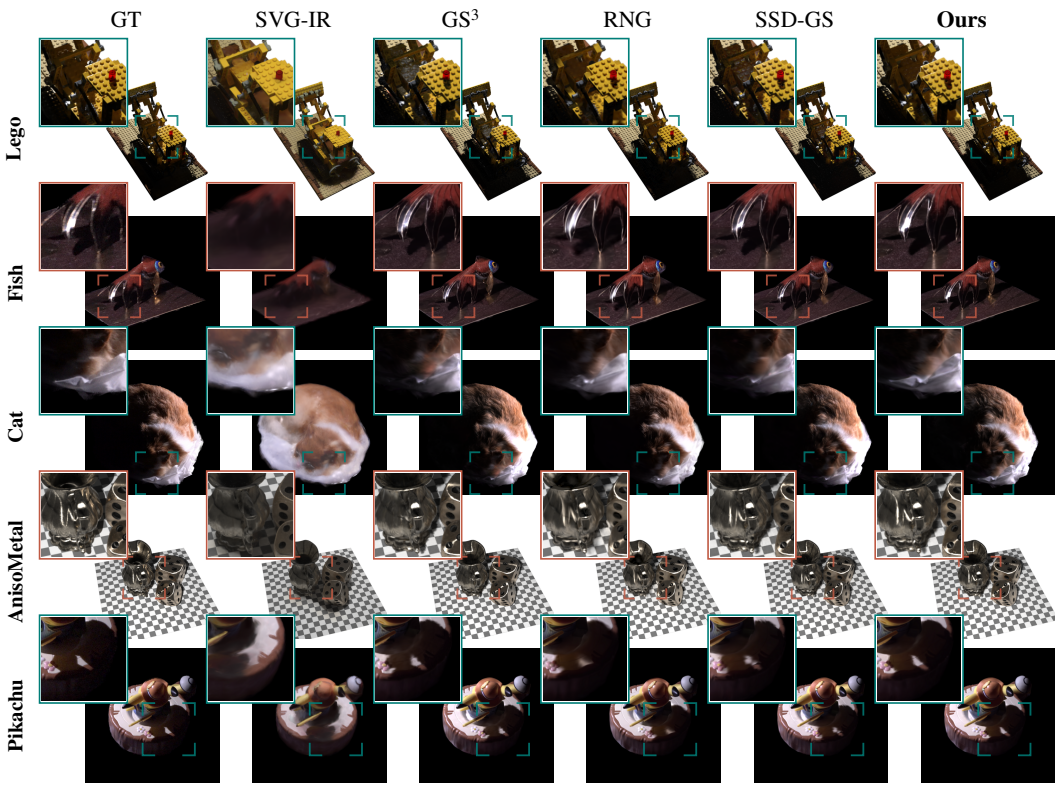

Figure 5: **Qualitative comparisons on representative real and synthetic scenes.** Insets highlight fine diffuse textures, specular highlights, and cast-shadow boundaries.

Table 3: **Rendering efficiency and module-time decomposition.** Mem.: peak PyTorch process allocated memory (MiB); #GS: number of Gaussians ($\times 10^3$). Spec.: specular; Comp.: compositing; Rast.: rasterization. Total time is independently measured synchronized wall time, not the sum.

(a) Rendering efficiency.

	GS³			RNG			SSD-GS			**Ours**		
Scene	Mem.↓	FPS↑	#GS↓	Mem.↓	FPS↑	#GS↓	Mem.↓	FPS↑	#GS↓	Mem.↓	FPS↑	#GS↓
Lego	741.5	77.29	529.2	2,437.1	19.95	949.7	1,909.4	27.33	548.7	775.5	77.04	320.2
Translucent	581.6	100.10	412.9	3,126.4	15.17	1,222.6	1,773.5	29.66	504.6	528.3	108.47	214.8
Fish	406.6	81.53	279.6	708.4	54.73	266.4	512.7	52.50	140.5	385.5	93.42	139.0
Container	285.3	153.75	185.8	1,078.8	45.26	395.7	296.9	115.12	62.3	405.9	135.67	135.3

(b) Module-time decomposition (ms/frame, ↓).

	GS³						**Ours**					
Scene	Shadow	Spec.	Neural	Comp.	Rast.	Total	Shadow	Spec.	Neural	Comp.	Rast.	Total
Lego	5.282	4.105	2.816	0.067	1.971	12.94	5.650	1.983	2.370	1.986	3.038	12.98
Translucent	3.351	2.650	2.232	0.055	1.514	9.99	4.028	1.098	1.424	1.037	2.130	9.22
Fish	5.681	2.409	1.399	0.044	2.502	12.27	6.382	0.835	0.676	0.530	2.932	10.70
Container	2.529	1.228	0.684	0.036	1.067	6.50	3.644	0.791	0.644	0.507	1.760	7.37

5 Conclusion

We presented LoFT-GS, a two-level representation for relightable Gaussian splatting that preserves spatially varying appearance within each Gaussian footprint while retaining shared Gaussian-level directional and neural computation. Experiments across 21 real, synthetic, and scattering scenes demonstrate consistent improvements over existing relighting methods and show that the proposed representation can also complement models with additional scattering components. Together, these results show that increasing relighting resolution within a Gaussian provides a compact and effective way to improve high-fidelity novel-light rendering while retaining the efficiency of Gaussian splatting.

AI USE STATEMENT

Generative AI tools were used to assist with manuscript polishing and editing, discussion of experimental analysis, design, and presentation, implementation of parts of the proposed algorithm, data processing, and modification of experimental scripts. All AI-assisted text, code, analyses, and experimental workflows were reviewed and verified by the authors. The authors take full responsibility for the final manuscript, implementation, experimental results, and conclusions.

REFERENCES

Zoubin Bi, Yixin Zeng, Chong Zeng, Fan Pei, Xiang Feng, Kun Zhou, and Hongzhi Wu. GS^3: Efficient relighting with triple Gaussian splatting. In *SIGGRAPH Asia 2024 Conference Papers*, pp. 1–12, December 2024. doi: 10.1145/3680528.3687576.

Mark Boss, Varun Jampani, Raphael Braun, Ce Liu, Jonathan T. Barron, and Hendrik P. A. Lensch. Neural-PIL: Neural pre-integrated lighting for reflectance decomposition. In *Advances in Neural Information Processing Systems*, volume 34, pp. 10691–10704, 2021.

Brian Chao, Hung-Yu Tseng, Lorenzo Porzi, Chen Gao, Tuotuo Li, Qinbo Li, Ayush Saraf, Jia-Bin Huang, Johannes Kopf, Gordon Wetzstein, and Changil Kim. Textured Gaussians for enhanced 3D scene appearance modeling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 8964–8974, June 2025. doi: 10.1109/CVPR52734.2025.00838.

Hongze Chen, Zehong Lin, and Jun Zhang. GI-GS: Global illumination decomposition on Gaussian splatting for inverse rendering. In *International Conference on Learning Representations*, pp. 41825–41854, 2025. URL https://openreview.net/forum?id=hJIEtJlvhL.

Paul Debevec, Tim Hawkins, Chris Tchou, Haarm-Pieter Duiker, Westley Sarokin, and Mark Sagar. Acquiring the reflectance field of a human face. In *Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques*, pp. 145–156, 2000. doi: 10.1145/344779.344855.

Jan-Niklas Dihlmann, Arjun Majumdar, Andreas Engelhardt, Raphael Braun, and Hendrik P. A. Lensch. Subsurface scattering for Gaussian splatting. In *Advances in Neural Information Processing Systems*, volume 37, pp. 121765–121789. Curran Associates, Inc., 2024. doi: 10.52202/079017-3870. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/dc72529d604962a86b7730806b6113fa-Paper-Conference.pdf.

Julie Dorsey, Holly E. Rushmeier, and François X. Sillion. *Digital Modeling of Material Appearance*. Morgan Kaufmann, 2007. ISBN 978-0-12-221181-2.

Jiahui Fan, Fujun Luan, Jian Yang, Miloš Hašan, and Beibei Wang. RNG: Relightable neural Gaussians. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 26525–26534, June 2025. doi: 10.1109/CVPR52734.2025.02470.

Duan Gao, Guojun Chen, Yue Dong, Pieter Peers, Kun Xu, and Xin Tong. Deferred neural lighting: Free-viewpoint relighting from unstructured photographs. *ACM Transactions on Graphics*, 39(6), December 2020. doi: 10.1145/3414685.3417767.

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 *Computer Vision – ECCV 2024*, volume 15103 of *Lecture Notes in Computer Science*, pp. 73–89. Springer, 2024. doi: 10.1007/978-3-031-72995-9_5.

Chun Gu, Xiaofei Wei, Zixuan Zeng, Yuxuan Yao, and Li Zhang. IRGS: Inter-reflective Gaussian splatting with 2D Gaussian ray tracing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 10943–10952, June 2025. doi: 10.1109/CVPR52734.2025.01022.

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, July 2024. doi: 10.1145/3641519.3657428.

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 (CVPR)*, pp. 5322–5332, June 2024. doi: 10.1109/CVPR52733.2024.00509.

Haian Jin, Isabella Liu, Peijia Xu, Xiaoshuai Zhang, Songfang Han, Sai Bi, Xiaowei Zhou, Zexiang Xu, and Hao Su. TensoIR: Tensorial inverse rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 165–174, June 2023. doi: 10.1109/CVPR52729.2023.00024.

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

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

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, 2015. doi: 10.48550/arXiv.1412.6980. URL `https://arxiv.org/abs/1412.6980`.

Zhiyi Kuang, Yanchao Yang, Siyan Dong, Jiayue Ma, Hongbo Fu, and Youyi Zheng. OLAT Gaussians for generic relightable appearance acquisition. In *SIGGRAPH Asia 2024 Conference Papers*, pp. 1–11, December 2024. doi: 10.1145/3680528.3687671.

Jingzhi Li, Zongwei Wu, Eduard Zamfir, and Radu Timofte. ReCap: Better Gaussian relighting with cross-environment captures. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 21307–21316, June 2025. doi: 10.1109/CVPR52734.2025.01985.

Zhihao Liang, Qi Zhang, Ying Feng, Ying Shan, and Kui Jia. GS-IR: 3D Gaussian splatting for inverse rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 21644–21653, June 2024. doi: 10.1109/CVPR52733.2024.02045.

Yehonathan Litman, Fernando De la Torre, and Shubham Tulsiani. LightSwitch: Multi-view relighting with material-guided diffusion. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 27750–27759, October 2025. doi: 10.1109/ICCV51701.2025.02576.

Zhenyuan Liu, Yu Guo, Xinyuan Li, Bernd Bickel, and Ran Zhang. BiGS: Bidirectional primitives for relightable 3D Gaussian splatting. In *2025 International Conference on 3D Vision (3DV)*, pp. 1022–1031. IEEE, March 2025. doi: 10.1109/3DV66043.2025.00099.

Krishna Mullia, Fujun Luan, Xin Sun, and Miloš Hašan. RNA: Relightable neural assets. *ACM Transactions on Graphics*, 44(1), February 2025. doi: 10.1145/3695866.

Panagiotis Papantonakis, Georgios Kopanas, Frédo Durand, and George Drettakis. Content-aware texturing for Gaussian splatting. In Beibei Wang and Alexander Wilkie (eds.), *Eurographics Symposium on Rendering*. The Eurographics Association, 2025. ISBN 978-3-03868-292-9. doi: 10.2312/sr.20251190.

Victor Rong, Jingxiang Chen, Sherwin Bahmani, Kiriakos N. Kutulakos, and David B. Lindell. GStex: Per-primitive texturing of 2D Gaussian splatting for decoupled appearance and geometry modeling. In *Proceedings of the Winter Conference on Applications of Computer Vision (WACV)*, pp. 3508–3518, February 2025. doi: 10.1109/WACV61041.2025.00346.

Christophe Schlick. An inexpensive BRDF model for physically-based rendering. *Computer Graphics Forum*, 13(3):233–246, 1994. doi: 10.1111/1467-8659.1330233.

Hanxiao Sun, Yupeng Gao, Jin Xie, Jian Yang, and Beibei Wang. SVG-IR: Spatially-varying Gaussian splatting for inverse rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 16143–16152, June 2025. doi: 10.1109/CVPR52734.2025.01505.

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

Tianhao Xie, Linlian Jiang, Xinxin Zuo, Yang Wang, and Tiberiu Popa. FACT-GS: Frequency-aligned complexity-aware texture reparameterization for 2D Gaussian splatting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Findings*, 2026. URL https://arxiv.org/abs/2511.23292.

Kun Xu, Wei-Lun Sun, Zhao Dong, Dan-Yong Zhao, Run-Dong Wu, and Shi-Min Hu. Anisotropic spherical Gaussians. *ACM Transactions on Graphics*, 32(6), 2013. doi: 10.1145/2508363.2508386.

Tian-Xing Xu, Wenbo Hu, Yu-Kun Lai, Ying Shan, and Song-Hai Zhang. Texture-GS: Disentangling the geometry and texture for 3D Gaussian splatting editing. In *Computer Vision – ECCV 2024*, volume 15083 of *Lecture Notes in Computer Science*, pp. 37–53. Springer, 2024. doi: 10.1007/978-3-031-72698-9_3.

Kai Ye, Chong Gao, Guanbin Li, Wenzheng Chen, and Baoquan Chen. GeoSplatting: Towards geometry guided Gaussian splatting for physically-based inverse rendering. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 28991–29000, October 2025. doi: 10.1109/ICCV51701.2025.02692.

Hong-Xing Yu, Michelle Guo, Alireza Fathi, Yen-Yu Chang, Eric Ryan Chan, Ruohan Gao, Thomas Funkhouser, and Jiajun Wu. Learning object-centric neural scattering functions for free-viewpoint relighting and scene composition. *Transactions on Machine Learning Research*, 2023. doi: 10.48550/arXiv.2303.06138. URL https://openreview.net/forum?id=NrfSRtTpN5.

Chong Zeng, Guojun Chen, Yue Dong, Pieter Peers, Hongzhi Wu, and Xin Tong. Relighting neural radiance fields with shadow and highlight hints. In *ACM SIGGRAPH 2023 Conference Proceedings*, pp. 1–11, August 2023. doi: 10.1145/3588432.3591482.

Libo Zhang, Yuxuan Han, Wenbin Lin, Jingwang Ling, and Feng Xu. PRTGaussian: Efficient relighting using 3D Gaussians with precomputed radiance transfer. In *2024 Asia Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*, pp. 1–6. IEEE, December 2024. doi: 10.1109/APSIPAASC63619.2025.10849016.

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

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

Xiaoming Zhao, Pratul P. Srinivasan, Dor Verbin, Keunhong Park, Ricardo Martin-Brualla, and Philipp Henzler. IllumiNeRF: 3D relighting without inverse rendering. In *Advances in Neural Information Processing Systems*, volume 37, pp. 42593–42617, 2024. doi: 10.52202/079017-1349.

Iris Zheng, Guojun Tang, Alexander Doronin, Paul D. Teal, and Fang-Lue Zhang. SSD-GS: Scattering and shadow decomposition for relightable 3D Gaussian splatting. In *International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=7m2Dqz9g05.

A Additional Method Details

This supplement expands the components introduced in the Method. We first describe how the Gaussian plane carries the local grid, then specify the directional reflectance model, shared networks, and optimization schedule.

A.1 Local Coordinates and Pixel Sampling

The local grid is attached to the Gaussian, not to a camera image. Using the geometric quantities in Sec. 3.1, a point on its plane can be written as

$$\mathbf{x}(\xi, \zeta) = \boldsymbol{\mu} + s_1 \xi \mathbf{t}_1 + s_2 \zeta \mathbf{t}_2. \tag{11}$$

Here, ξ and ζ measure displacement along the two Gaussian axes in units of their corresponding scales. Consequently, different camera or light rays reaching the same plane location refer to the same local texture location. We map the region spanning three in-plane standard deviations along each axis to normalized coordinates $(u, v) \in [0, 1]^2$. The fixed-resolution representation uses a compact 4×4 grid.

Texel shading. For each light-view pair, we evaluate the ASG response and both MLPs once per Gaussian. The light and viewing directions $\boldsymbol{\omega}_i, \boldsymbol{\omega}_o$ and point-light distance d are measured at the Gaussian center and shared by all its texels. Combining the shared responses with the local factors gives the direct-shading color

$$\mathbf{c}(u, v) = \frac{1}{d^2} \left[\mathbf{k}_{\mathrm{d}}(u, v) f_{\mathrm{d}}(\boldsymbol{\omega}_i) + \mathbf{k}_{\mathrm{s}} f_{\mathrm{s}}(\boldsymbol{\omega}_i, \boldsymbol{\omega}_o; u, v) \right], \tag{12}$$

the shadow factor from Eq. 7,

$$s(u, v) = 1 - \eta(u, v) + \eta(u, v) \widehat{s}(u, v), \tag{7}$$

and the additive RGB response

$$\mathbf{r} = \frac{1}{d^2} \Psi(\boldsymbol{\omega}_i, \boldsymbol{\omega}_o; \boldsymbol{\mu}, \boldsymbol{\ell}). \tag{13}$$

Here, $\widehat{s}$ is the range-mapped visibility in Eq. 6. Both the direct-shading and additive responses include inverse-square distance attenuation. The shading color and shadow factor vary over texels, whereas the additive response is shared across the Gaussian.

Camera-pixel sampling. To obtain a Gaussian's contribution to a camera pixel, we first intersect the pixel's ray with its plane. This gives a location (u, v) on the texture grid, generally between texels. We therefore interpolate the values at the four surrounding texels rather than choosing a single texel.

For an $r \times r$ grid, let $x = (r - 1)u$ and $y = (r - 1)v$, and let $a = x - \lfloor x \rfloor$ and $b = y - \lfloor y \rfloor$ be the fractional distances within a grid cell. Denote its four neighboring values by $X_{00}, X_{10}, X_{01}, X_{11}$, where the subscripts indicate the horizontal and vertical offsets. Bilinear interpolation gives

$$\widetilde{X}(u, v) = (1 - a)(1 - b) X_{00} + a(1 - b) X_{10} + (1 - a) b X_{01} + ab X_{11}. \tag{14}$$

At grid boundaries, neighbor indices are clamped to the valid texel range; a 1×1 grid returns its sole value. We apply Eq. 14 to the already-evaluated shading color $\mathbf{c}$ and shadow factor s in Eqs. 12 and 7, not to the appearance parameters before shading. The additive response $\mathbf{r}$ is Gaussian-level and needs no interpolation. These three quantities are then composited over the Gaussians contributing to the pixel, using Eq. 1.

A.2 Diffuse and Specular Functions

The smooth incident-angle weighting and modified ASG basis below follow the appearance model of GS^3 (Bi et al., 2024). The local factors in the Method add spatial variation without changing these shared functions.

Diffuse reflection. Let $\mathbf{n}_\mathrm{s}$ be the normal of a learned Gaussian-level shading frame, distinct from the geometric normal $\mathbf{n}$ in Sec. 3.1. Expressing the normal and incident-light direction in the same coordinate frame, the diffuse appearance function is

$$f_\mathrm{d}(\boldsymbol{\omega}_i) = \frac{\mathrm{ELU}_\varepsilon(\mathbf{n}_\mathrm{s}^\top \boldsymbol{\omega}_i) + \varepsilon(1 - e^{-1})}{\big(1 + \varepsilon(1 - e^{-1})\big)\pi}, \qquad \varepsilon = 0.01. \tag{15}$$

Here, $\mathrm{ELU}_\varepsilon(x)$ equals x for $x \geq 0$ and $\varepsilon(e^x - 1)$ otherwise. Compared with a clamped cosine, this smooth extension maintains a nonzero derivative for back-facing lighting directions, allowing the shading frame to continue adjusting. The cosine is included in f_d itself; it is not an additional factor in the image-formation equation. The local RGB albedo $\mathbf{k}_\mathrm{d}(u, v)$ sets the diffuse color, while the shared shading normal determines its incident-angle dependence. Neither quantity introduces a dependence on viewing direction.

ASG basis functions. ASGs were introduced by Xu et al. (2013) to represent anisotropic functions on the sphere. We use the angular-distance variant of GS^3, rather than the original hemisphere-weighted ASG formula. Each basis decreases with the angular distance from a preferred direction, at different rates along its two tangent axes. The mixture in Eq. 8 combines these bases into the Gaussian's directional response.

For basis j, let $\mathbf{z}_j$ be the peak direction and $\mathbf{x}_j, \mathbf{y}_j$ its two orthogonal tangent axes. The effective axes below include the learned Gaussian-level shading rotation. This frame orients the directional response as a whole and is shared by all texels; no separate shading normal is introduced at individual texels. Let $\theta_j = \arccos(\mathbf{h}^\top \mathbf{z}_j)$ be the angle from $\mathbf{h}$ to $\mathbf{z}_j$, and $\mathbf{s}_j$ the normalized projection of $\mathbf{h}$ onto the plane spanned by $\mathbf{x}_j, \mathbf{y}_j$. The basis is

$$G_j(\mathbf{h}) = \frac{1}{\sigma_{z,j}} \exp\left[-\frac{\theta_j^2}{2\sigma_{z,j}^2}\left(\frac{(\mathbf{s}_j^\top \mathbf{x}_j)^2}{\sigma_{x,j}^2} + \frac{(\mathbf{s}_j^\top \mathbf{y}_j)^2}{\sigma_{y,j}^2}\right)\right]. \tag{16}$$

The two tangent widths control how quickly the response falls off in different angular directions, while $\sigma_{z,j}$ controls its overall angular spread. A common constant scale factor is absorbed into the mixture coefficients β_j. For numerical evaluation, the inverse-cosine input is clamped away from ± 1, and projection normalization uses a small positive denominator floor.

The model uses eight basis functions. Their canonical shapes and frames are shared across the scene, while each Gaussian learns its mixture weights and the shading rotation that orients them. The local amplitude and sharpness in Eq. 9 modulate the resulting response; they do not introduce additional basis functions at individual texels.

Fresnel weighting. The Fresnel term describes the change in reflectivity with incidence angle. We use the Schlick approximation (Schlick, 1994):

$$F(\boldsymbol{\omega}_i, \mathbf{h}) = F_0 + (1 - F_0)\big(1 - \boldsymbol{\omega}_i^\top \mathbf{h}\big)^5, \tag{17}$$

where F_0 is the normal-incidence reflectance and is fixed to 0.04 in the inherited model. As incidence becomes more grazing relative to the half vector, the factor increases. It weights the ASG mixture in Eq. 8; the RGB specular color is provided separately by $\mathbf{k}_\mathrm{s}$.

A.3 Shared Neural Networks

To refine shadows without increasing neural evaluations with the number of texels, Φ predicts a shadow range once per Gaussian, as defined in Eq. 5. The second network, Ψ, accounts for appearance not captured by direct shading; its distance-scaled RGB response is defined in Eq. 13. The position and material feature allow each network to respond differently at different Gaussians while sharing its weights across the scene. The networks have separate weights, but use the same learned six-dimensional material feature for a given Gaussian. Their outputs are Gaussian-level quantities. In particular, Φ determines the shared shadow mapping, while the texture stores the local shadow contribution weight $\eta(u, v)$. Local shadow variation therefore does not require an MLP evaluation at every texel.

Network implementation. We encode position and light/view directions with four sine-cosine frequency bands before concatenating them with the material feature. Φ additionally receives the scalar visibility $\bar{V}$. The encoding uses the sinusoidal features without appending the raw coordinates. Both networks use three hidden layers with LeakyReLU activation; their widths and outputs are summarized in Table A1. Sigmoid output layers provide the bounded predictions of Φ and the nonnegative response of Ψ.

Table A1: The two shared MLPs. Each is evaluated once per Gaussian for the current light-view pair.

Network	Hidden widths	Output
Φ	$32, 32, 32$	Two bounded scalars L, W
Ψ	$128, 128, 128$	One nonnegative RGB response

A.4 Progressive Optimization

We use Adam (Kingma & Ba, 2015) to minimize the standard 3DGS photometric loss (Kerbl et al., 2023):

$$\mathcal{L} = (1 - \lambda)\mathcal{L}_1 + \lambda\mathcal{L}_{\text{D-SSIM}}, \qquad \lambda = 0.2, \tag{18}$$

Here, $\mathcal{L}_1$ is the pixelwise ℓ_1 loss and $\mathcal{L}_{\text{D-SSIM}} = 1 - \text{SSIM}$, with SSIM defined by Wang et al. (2004).

To establish a diffuse base before fitting flexible residuals and directional highlights, we introduce the appearance components progressively, following the diffuse-first strategy of GS^3 (Bi et al., 2024). Local grids are available from the beginning of training. Each local appearance factor participates as soon as its corresponding component is enabled, with no additional texel-only activation delay. The reference schedule has four phases:

1. **Diffuse warm-up (0–5K iterations).** Only the diffuse term contributes to the rendered image, establishing coarse geometry and a base surface color. The local diffuse albedo $\mathbf{k}_{\text{d}}(u, v)$ is optimized from the start and throughout later phases.
2. **Shadow and residual fitting (5K–9K).** We enable the shadow and other-effects networks while keeping specular reflection inactive. The model can then account for occlusion and residual illumination on top of the diffuse fit. The local shadow contribution weight $\eta(u, v)$ becomes active with the shadow term.
3. **Diffuse adaptation (9K–16K).** The shadow and residual networks remain in the forward model, but their weights and material features are held fixed. The local diffuse albedo and shadow contribution weights remain learnable. For HDR inputs, this phase also transitions the fitting target from gamma-compressed values to linear intensities.
4. **Full appearance fitting (16K onward).** We enable specular reflection, including the local amplitude $g(u, v)$ and sharpness $q(u, v)$, and resume the neural updates. The shared ASG tangent-width parameters and basis rotations remain fixed until 22K, after which they are also refined. This later release does not delay the local amplitude and sharpness, which are already active and learnable from 16K.

The visibility field $V(u, v)$ is recomputed from the light pass rather than optimized as an independent texture parameter. Thus, making all grids available from the start does not mean that all shading terms contribute during the diffuse warm-up.

This ordering encourages the components to retain their intended physical roles, without enforcing a unique physical decomposition. Subsequent training-schedule ablations in SSD-GS (Zheng et al., 2026) provide further evidence for progressive fitting: they report clearer component separation than introducing the terms jointly, even when scalar image-quality metrics are close. Their scattering-specific schedule differs from the one above.

Table A2: **Ablation of local appearance factors.** All variants are trained independently. Mean gives equal scene weight. Best and second-best results are highlighted using unrounded values.

Scene	w/o local diffuse			w/o local shadow			w/o local specular			**Full**		
	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
LightStage												
Boot	30.35	0.9149	0.1055	30.35	0.9137	0.1050	30.22	0.9143	0.1065	**30.40**	**0.9157**	**0.1049**
Container	38.90	0.9819	0.0174	**39.14**	**0.9831**	**0.0164**	38.94	0.9826	0.0166	39.08	0.9829	0.0166
Fox	35.34	0.9266	0.0852	35.35	0.9268	0.0857	35.30	0.9259	0.0870	**35.40**	**0.9275**	**0.0849**
Nefertiti	38.50	0.9653	0.0477	38.52	0.9657	**0.0473**	38.41	0.9648	0.0476	**38.61**	**0.9662**	0.0474
NRHints												
Cat	**27.93**	0.9034	0.1562	27.53	0.9016	0.1574	27.72	**0.9046**	**0.1553**	27.74	0.9038	0.1558
CupFabric	37.94	0.9846	0.0657	37.66	0.9841	0.0664	37.80	0.9844	0.0662	**38.00**	**0.9847**	**0.0655**
Fish	31.15	0.9248	0.0955	31.24	0.9251	**0.0945**	31.19	0.9242	0.0949	**31.29**	**0.9255**	**0.0945**
Pikachu	34.07	0.9705	0.0752	33.92	0.9705	0.0750	34.01	0.9706	0.0751	**34.11**	**0.9709**	**0.0743**
Synthetic												
AnisoMetal	28.82	0.9591	**0.0450**	28.64	0.9578	0.0472	28.46	0.9569	0.0475	**28.84**	**0.9593**	0.0451
MaterialBalls	30.44	0.9470	0.0816	30.24	0.9462	0.0836	29.94	0.9422	0.0890	**30.49**	**0.9490**	**0.0804**
Drums	31.78	0.9771	0.0288	29.98	0.9726	0.0310	31.52	0.9766	0.0298	**32.18**	**0.9788**	**0.0269**
FurBall	37.27	0.9721	0.0633	37.08	0.9709	0.0599	37.07	0.9717	0.0667	**37.73**	**0.9741**	**0.0561**
Hotdog	34.15	0.9767	0.0323	33.89	0.9767	0.0323	34.25	0.9771	0.0318	**34.30**	**0.9776**	**0.0310**
Lego	31.18	0.9582	0.0491	30.62	0.9568	0.0494	30.75	0.9560	0.0510	**31.40**	**0.9608**	**0.0471**
Translucent	33.54	0.9777	0.0337	33.89	**0.9796**	0.0322	33.00	0.9765	0.0346	**33.94**	0.9790	**0.0318**
Cup	35.95	0.9894	**0.0247**	35.92	0.9894	0.0255	35.10	0.9880	0.0295	**35.97**	**0.9896**	0.0250
Mean	33.58	0.9581	0.0629	33.37	0.9575	0.0631	33.36	0.9573	0.0643	**33.72**	**0.9591**	**0.0617**

A.5 Implementation Details and Baseline Adaptations

All methods use a 100K-iteration training budget. Densification ends at 80K iterations for real scenes and 50K iterations for synthetic scenes. These densification schedules are shared across methods, except for RNG staged training, which uses its own schedule.

RNG. For a fair comparison, we enable camera-pose optimization in the released RNG implementation, matching the pose-optimization setting used by GS^3 and SSD-GS. We train both forward and staged forward-to-deferred variants and report the stronger result for each scene.

Scattering scenes. To evaluate trained scattering models correctly, we fixed checkpoint restoration in the released SSD-GS code. After loading a checkpoint, the original code re-initialized the scattering state, overwriting the saved subsurface-scattering coefficient k_{sss} and scatter-MLP parameters, so renderings from a trained model did not use its learned scattering. With the fix, these saved parameters are restored before rendering.

Integration with SSD-GS. Ours (SSD-GS) combines the local diffuse and specular factors and the range-based texel shadow of LoFT-GS with the Gaussian-level scatter MLP and k_{sss} of SSD-GS, rendered in the 2DGS deferred renderer of LoFT-GS. Local color and shadow use texture channels, while scattering remains Gaussian-level in the deferred other-effects channel (I_{residual} in Eq. 10). Apart from the checkpoint fix above, the official SSD-GS renderer is unchanged.

B Additional Ablation Results

Figure A1 compares RNG, GS^3, and LoFT-GS under a moving point light with a fixed viewpoint. Table A2 lists the per-scene results behind the means in Table 2(a). Figure A2 compares the direct specular component of GS^3 and LoFT-GS, and Figure A3 probes local amplitude and sharpness on AnisoMetal.

C Local Grid Resolution

All main experiments use a fixed 4×4 grid per Gaussian. This appendix reports the runs behind that choice.

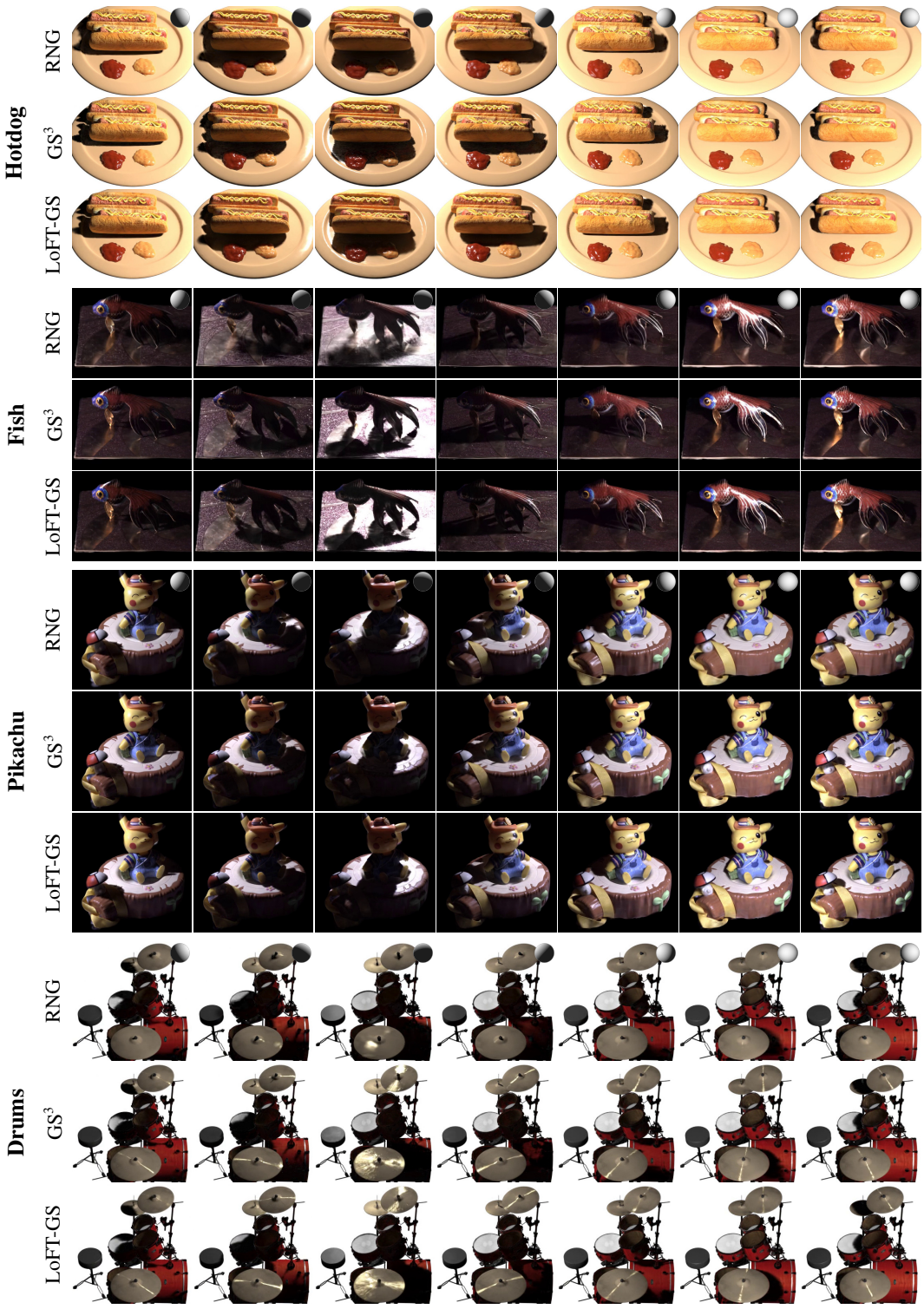

Figure A1: **Relighting under a moving point light.** For each scene, the viewpoint is fixed while a point light moves around the object; the seven columns are evenly spaced along its path. Rows compare RNG, GS^3, and LoFT-GS on the same enlarged crop. Insets indicate light direction. Full sequences are provided as supplementary videos.

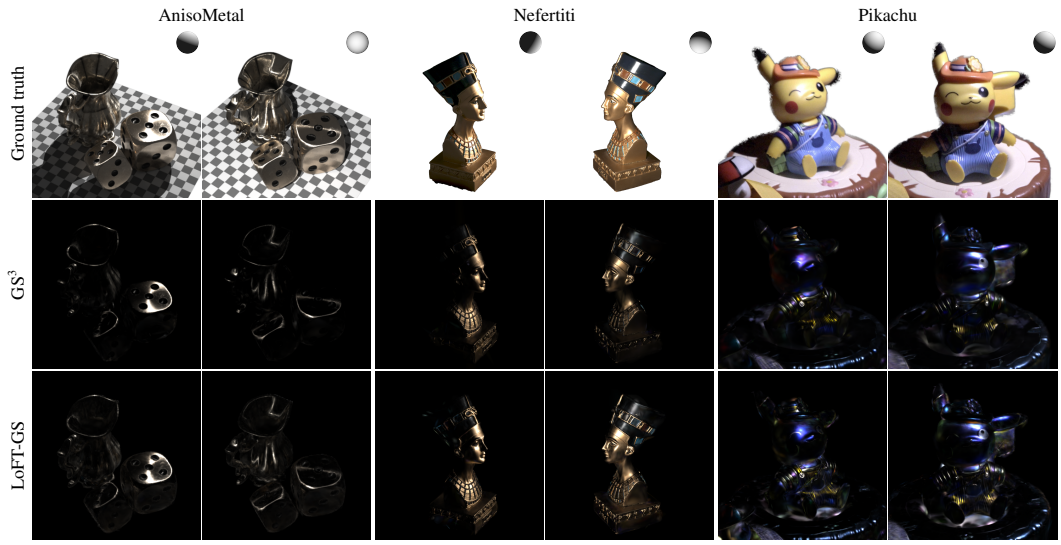

Figure A2: Specular-factor comparison for two captures per scene. The top row provides RGB references; the remaining rows show the isolated, pre-shadow direct specular component of each method. Insets indicate light direction.

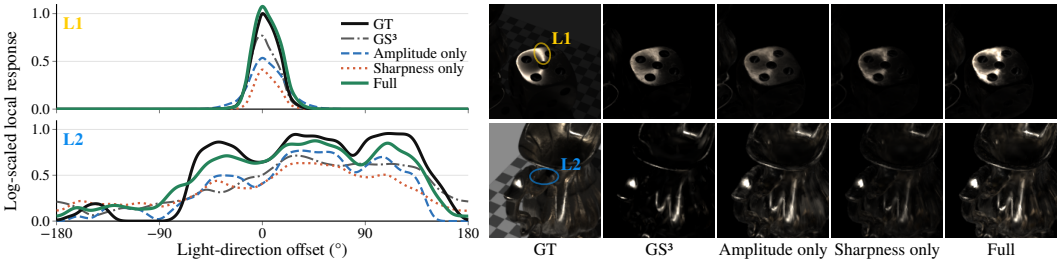

Figure A3: **Local amplitude and sharpness analysis.** We probe two representative regions L1 and L2 while varying the light direction. Amplitude-only and sharpness-only responses are obtained from the same trained Full model by disabling the complementary local modulation. The right panels show representative direct-specular renderings.

C.1 Choice of Grid Resolution

Table A3 varies the grid on Pikachu and Hotdog. Over Gaussian-level relighting, 2×2, 3×3, and 4×4 grids add 0.24, 0.33, and 0.52 dB on Pikachu, and 0.15, 0.37, and 0.54 dB on Hotdog. Beyond 4×4, the remaining gain is small. A 5×5 grid uses $1.4\times$ and $1.7\times$ the 4×4 texel count, yet leaves Pikachu unchanged (34.107 vs. 34.110 dB) and adds 0.06 dB on Hotdog. Growing grids during optimization (forward adaptive) adds 0.08 dB on Pikachu at $3.0\times$ the texels, and 0.01 dB on Hotdog at $1.5\times$. The fixed 4×4 grid thus reaches 86% and 90% of the largest gain observed in each scene, and we use it for all main experiments.

C.2 Reverse Coarsening

Since growing grids is expensive, we also tried the opposite direction on six scenes. Starting from the fixed 4×4 checkpoint at 80k iterations, we lower a grid by one level at 85k, 90k, and 95k whenever a coarser grid can reproduce its local factors, and continue training to 100k. For a candidate reduction, we downsample the diffuse albedo $X_{\mathrm{d}} = \mathbf{k}_{\mathrm{d}}$, the log amplitude and sharpness $X_{\mathrm{s}} = (\log g, \log q)$, and the centered visibility $X_{\mathrm{v}} = V - \bar{V}$ by area averaging, upsample them back to $\widehat{X}_f$, and measure

$$e_f = \frac{\mathrm{mean}\,|X_f - \widehat{X}_f|}{\max(\mathrm{mean}\,|X_f|, \delta_f)}, \qquad E = \max\{e_{\mathrm{d}}, e_{\mathrm{s}}, \bar{e}_{\mathrm{v}}\}, \tag{19}$$

Table A3: **Grid resolution.** Fixed and adaptive grids on Pikachu and Hotdog. Δ: PSNR gain over Gaussian-level only.

Local grid	Pikachu			Hotdog		
	Texels↓	PSNR↑	Δ	Texels↓	PSNR↑	Δ
Gaussian-level only	—	33.59	—	—	33.76	—
Fixed 2×2	0.43M	33.83	+0.24	0.73M	33.91	+0.15
Fixed 3×3	0.92M	33.92	+0.33	1.76M	34.13	+0.37
Fixed 4×4	1.79M	34.11	+0.52	3.13M	34.30	+0.54
Fixed 5×5	2.54M	34.11	+0.52	5.24M	34.36	+0.60
Forward adaptive	5.43M	34.19	+0.60	4.77M	34.31	+0.56

Table A4: **Reverse coarsening.** PSNR of the fixed 4×4 model and after coarsening. Texel↓: fraction of texels removed.

Scene	Fixed 4×4	Coarsened	Δ	Texel↓
Boot	30.40	30.31	-0.09	28.1%
Container	39.08	38.98	-0.10	28.5%
CupFabric	38.00	38.04	$+0.04$	29.4%
Fish	31.29	31.16	-0.13	29.2%
AnisoMetal	28.84	28.71	-0.13	26.4%
FurBall	37.73	37.68	-0.05	29.6%
Average	34.22	34.15	-0.08	28.5%

with $\delta_{\mathrm{d}} = 0.03$ and $\delta_{\mathrm{s}} = \delta_{\mathrm{v}} = 0.02$; $\bar{e}_{\mathrm{v}}$ averages the visibility error over light-view pairs in which the Gaussian is visible. A reduction is accepted when E is below the transition threshold (0.65, 0.45, and 0.30), with at most 25% of the Gaussians coarsened per event. The smaller grid receives area-averaged local factors; the shared directional model and networks are unchanged.

Table A4 lists the results. Coarsening removes 26.4–29.6% of the texels (28.5% on average). Compared with the fixed 4×4 models, PSNR rises on CupFabric (+0.04 dB) and drops by 0.05–0.13 dB on the other five scenes, -0.08 dB on average over all six. Coarsening thus trades a small PSNR loss for about 28% fewer texels; we keep the fixed 4×4 grid for all main results and report reverse coarsening only as an observation.