The Visual Computer (2026) 42:404
https://doi.org/10.1007/s00371-026-04610-y

RESEARCH

PGSR-DR: high-fidelity reflective surface reconstruction with planar-based Gaussians and deferred rendering

Jingfeng Li[1] · Xiaokun Wang[1] · Haokai Zeng[1] · Xingyu Ye[2] · Jiří Kosinka[3] · Alexandru C. Telea[4] · Yalan Zhang[1] · Yanrui Xu[5]

Received: 5 May 2026 / Accepted: 17 June 2026

Abstract

While 3D Gaussian Splatting (3DGS) has revolutionized novel-view synthesis, accurately recovering reflective surfaces remains a significant challenge due to inherent depth estimation errors and the limited capacity of spherical harmonics in representing high-frequency reflections. In this paper, we propose PGSR-DR, a reflection-aware framework that integrates planar-based Gaussian reconstruction with deferred rendering for high-fidelity geometry and appearance recovery. We first establish a reliable geometric foundation by introducing a depth-calculation method for planar-based Gaussians. Our method eliminates conventional estimation artifacts and incorporates joint depth-normal consistency and multi-view supervision to ensure global structural coherence. To capture intricate specularities, we incorporate a learnable environment map within a deferred rendering pipeline that uses Nvdiffrast for efficient sampling and explicit modeling of view-dependent appearances. Experimental results demonstrate that our method achieves competitive rendering quality and notably improved geometric accuracy for reflective surfaces, with planar-based Gaussian primitives closely adhering to the underlying surfaces while maintaining real-time performance.

Keywords Reflective surface reconstruction · Planar-based Gaussian splatting · Geometric optimization · Deferred rendering

1 Introduction

Photorealistic novel-view synthesis (NVS) and accurate geometric reconstruction are key objectives in computer graphics and vision with extensive applications in AR/VR, 3D content generation, and autonomous driving [1, 2]. Neural Radiance Fields (NeRF) [1] represent a milestone in this domain by leveraging neural implicit functions and volume-based ray marching to achieve high-fidelity rendering. However, the heavy computational cost of volume rendering often yields long training cycles and also hinders real-time inference. Recently, 3D Gaussian Splatting (3DGS) [2] has emerged as a transformative alternative. By optimizing explicit 3D Gaussian primitives and using a tile-based rasterization pipeline, 3DGS achieves rapid convergence and millisecond-level rendering speeds.

Despite these advantages, standard 3DGS relies primarily on *photometric loss* to supervise Gaussian optimization,

✉ Xiaokun Wang
wangxiaokun@ustb.edu.cn

Jingfeng Li
m202421736@xs.ustb.edu.cn

Haokai Zeng
haok_z@xs.ustb.edu.cn

Xingyu Ye
xye@bournemouth.ac.uk

Jiří Kosinka
j.kosinka@rug.nl

Alexandru C. Telea
a.c.telea@uu.nl

Yalan Zhang
zhangyl@ustb.edu.cn

Yanrui Xu
xuyr@mail.tsinghua.edu.cn

1 School of Artificial Intelligence, University of Science and Technology Beijing, Beijing, China

2 National Centre for Computer Animation, Bournemouth University, Poole, UK

3 University of Groningen, Groningen, The Netherlands

4 Utrecht University, Utrecht, The Netherlands

5 Department of Computer Science and Technology, Tsinghua University, Beijing, China

Published online: 12 July 2026

which often leads to poor geometric convergence. Without explicit structural constraints, Gaussian primitives tend to "float" in empty space rather than conform to actual object surfaces, resulting in significant artifacts in geometric reconstruction. Recent advancements, such as 2DGS [3] and PGSR [4], have mitigated these issues by introducing view-consistent geometric priors and planar constraints. While achieving high-fidelity surface reconstruction for Lambertian scenes, these methods however consistently fail when handling highly specular surfaces. In such scenarios, high-frequency, view-dependent reflections confuse the optimization process, causing existing methods to struggle with the ambiguity between surface geometry and reflected radiance.

To address these challenges, we propose PGSR-DR, a reflection-aware framework that combines geometric optimization with deferred rendering to achieve both high-precision reconstruction and photorealistic appearance synthesis. Our approach is built upon three key technical insights:

1. We mitigate the depth estimation bias in conventional Gaussian rasterization by deriving a **depth calculation**. Under a planarity assumption, this yields more stable and accurate depth values.
2. We jointly enforce **depth-normal consistency** and **multi-view geometric supervision**, ensuring that Gaussian primitives adhere to the underlying surfaces with global coherence.
3. We overcome the representational limits of spherical harmonics in capturing high-frequency reflections by integrating a **learnable environment map** in a deferred rendering pipeline. By leveraging Nvdiffrast [5] for efficient sampling, we explicitly decouple intricate specularities from surface geometry.

Experimental results demonstrate that PGSR-DR achieves competitive rendering quality and consistently better geometric accuracy than prior real-time approaches on reflective surfaces. The planar-based primitives adhere closely to object surfaces, producing globally coherent geometry. This gain reflects a quality–speed trade-off: on synthetic scenes, our method runs at real-time rates comparable to leading methods; on real-world scenes, throughput drops as dense primitives raise the cost of per-pixel depth computation.

To summarize, our main contributions are:

- We build upon planar-based representations and augment 3D Gaussian primitives with additional attributes (normals, depth, reflection strength) enabling **a joint optimization framework** that combines depth-normal and multi-view consistency to accurately align Gaussians with the scene geometry.
- We design a **learnable environment map** sampled via Nvdiffrast and integrated into a deferred rendering framework that enables high-fidelity synthesis of complex, view-dependent specular appearances.
- We show via extensive **evaluations** that our method delivers a favorable quality-speed trade-off: it provides geometric and appearance reconstruction quality. In particular, compared to existing methods in the same class, our method offers better geometric accuracy and/or better visual quality on specular surfaces.

In the following, we first review relevant related work (Sect. 2). Section 3 overviews Gaussian splatting and its limitations. Section 4 details our method. Section 5 presents results and relevant comparisons. Finally, Sect. 6 concludes the paper.

2 Related work

2.1 Neural surface reconstruction

Early neural reconstruction frameworks primarily used voxels [6], point clouds [7], or implicit fields [8] to recover geometry. These pioneering methods often struggled with the trade-off between resolution and computational cost. NeRF [1] represents scenes with a Multi Layer Perceptron (MLP) that maps spatial locations to color and density, computed via volume rendering. Yet, its density-based representation often yields noisy, 'foggy' geometry. To address this, NeuS [9] and VolSDF [10] integrated Signed Distance Functions (SDFs) into the volume rendering pipeline, ensuring smoother surface extraction. Neuralangelo [11] pushed quality further by using multi-resolution hash encodings to capture fine-grained details.

However, the implicit nature of MLPs in NeRF-based models leads to relatively *slow inference*. 3D Gaussian Splatting (3DGS) [2] emerged as a breakthrough offering real-time rendering via explicit primitives. To bridge the gap between splatting and accurate surfaces, SuGaR [12] enforced Gaussians to align with surface geometry for Poisson reconstruction, while 2DGS [3] collapsed 3D volumes into view-oriented 2D disks to improve geometric consistency. Recent extensions such as GOF [13] and PGSR [4] introduced opacity fields and multi-view regularization, respectively. VCR-GauS [14] regularizes Gaussian primitives with geometric priors distilled from a pre-trained normal estimator. Despite these advances, existing 3DGS methods predominantly assume *Lambertian* reflectance and yield significant geometric distortion when applied to highly specular surfaces.

2.2 Specular and reflection modeling

Modeling reflective objects is inherently hard due to the high-frequency, view-dependent nature of specular highlights. Ref-NeRF [15] addressed this by re-parameterizing radiance as a function of the reflected direction and introducing a pred-normal constraint. NeRF-Casting [16] and ENVIDR [17] utilized neural ray tracing to simulate complex light transport. To handle metallic surfaces, NeRO [18] decomposed scenes into material and lighting components using physical-based rendering (PBR) principles.

In the 3DGS domain, recent work has attempted to replace simple spherical harmonics (SH) with more expressive reflection models. 3iGS [19] employs tensorial factorization to optimize incident illumination. GaussianShader [20] separately models view-dependent effects. To handle complex environment reflections, 3DGS-DR [21] and Ref-GS [22] adopted *deferred rendering* architectures. Specifically, 3DGS-DR uses a data-driven approach to optimize normal directions; however, this yields prolonged training time. Ref-GS uses a MLP to query environment colors, which incurs high inference-time overhead and makes real-time rendering difficult. In contrast to previous work, we introduce geometric constraints to regularize normal directions and leverage Nvdiffrast [5] for efficient environment map sampling, thereby achieving both fast training convergence and real-time rendering performance.

3 Gaussian splatting preliminaries

3D Gaussian Splatting (3DGS) [2] represents a 3D scene using a set of anisotropic 3D Gaussians. Each such Gaussian is described by its center $\boldsymbol{\mu} \in \mathbb{R}^3$ and a 3D covariance matrix Σ. To ensure Σ remains positive semi-definite during optimization, it is factorized into a scaling matrix S and a rotation matrix R via

$$\Sigma = RSS^\top R^\top. \tag{1}$$

The influence of a Gaussian $(\boldsymbol{\mu}, \Sigma)$ at a point $\mathbf{x}$ is then

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

For rendering, such 3D Gaussians are projected onto a 2D image plane. Given a viewing transformation W and the Jacobian J of the projective transformation, the covariance matrix in camera coordinates Σ' is given as follows:

$$\Sigma' = JW\Sigma W^\top J^\top. \tag{3}$$

Given a set of $\mathcal{N}$ such Gaussians, the appearance of each Gaussian $1 \le i \le \mathcal{N}$ is parameterized by an opacity $\alpha_i \in [0, 1]$ and color $\mathbf{c}_i$ typically represented by SH coefficients. 3DGS renders the color of a specific pixel using a tile-based rasterizer to sort Gaussians by depth. The final pixel color C is computed via front-to-back alpha blending

$$C = \sum_{i\in\mathcal{N}} \mathbf{c}_i \alpha_i \prod_{j=1}^{i-1}(1-\alpha_j), \tag{4}$$

Similarly, the rendered depth D in conventional 3DGS is typically computed as the weighted sum of the distances z_i from Gaussian centers $\boldsymbol{\mu}_i$ to the image plane as

$$D = \sum_{i\in\mathcal{N}} z_i \alpha_i \prod_{j=1}^{i-1}(1-\alpha_j). \tag{5}$$

Limitations: While the SH coefficients in Eq. (4) can model low-frequency view-dependent effects, they struggle to capture high-frequency reflections on specular surfaces. Also, the standard depth calculation in Eq. (5) often introduces estimation biases, as it treats Gaussian centers as discrete points rather than considering the continuous distribution along the ray, which we address in PGSR-DR as explained next.

4 Method

As explained in Sect. 2, reconstructing and rendering specular surfaces from multi-view images remains challenging due to their view-dependent appearance and complex light interactions. We address this by a joint training framework for geometry and appearance, which achieves efficient training and accurate reconstruction by modeling and optimizing geometric shape and appearance color separately (see Fig. 1). We first extend the properties of 3DGS to generate a G-Buffer that encodes geometric information (Sect. 4.1). Next, we introduce a learnable environment map to fetch reflective colors and render them in conjunction with the G-Buffer (Sect. 4.2). Finally, we impose depth-normal consistency regularization to enforce geometric constraints (Sect. 4.3).

4.1 Extension of Gaussian properties

To better reconstruct and render object surfaces, we introduce three per-Gaussian attributes: Two geometric attributes (normal and depth) enforce geometric constraints via corresponding normal and depth maps, and a reflectance attribute (reflect strength) to modulate the influence of the learned environment map during deferred shading. We detail these next.

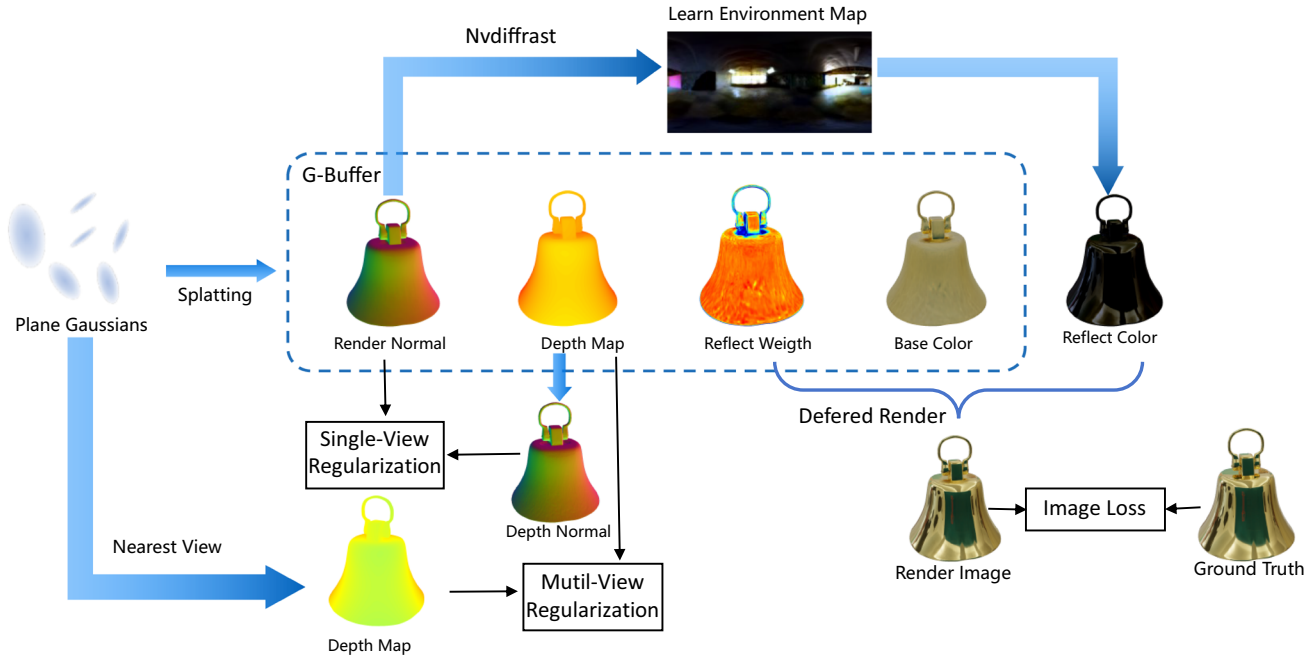


Fig. 1 Overview of PGSR-DR. During rendering, we first generate a per-pixel G-buffer via Gaussian splatting (Eq. 2). Based on this buffer, we adopt a deferred shading approach that uses the normal map (computed as explained in Sect. 4.1) to sample reflection colors from the environment map. Base and reflection colors are then blended via a reflection weight map. During training, we enforce depth-normal consistency constraints together with single-view and multi-view geometric regularization, which guides Gaussians to closely adhere to the object surface

Normal vector: Following NeuSG [23], the normal vector $\mathbf{n}_i$ of a Gaussian is defined as the row of its rotation matrix R_i that corresponds to the smallest scaling factor s_k of S_i. In the world coordinate system, this is expressed as

$$\mathbf{n}_i = R_i[k, :] \in \mathbb{R}^3, \quad k = \text{argmin}\big([s_1, s_2, s_3]\big) \in \mathbb{R}, \tag{6}$$

which aligns the normal $\mathbf{n}_i$ with the direction of the Gaussian's shortest axis.

Since $\mathbf{n}_i$ is constant over a given Gaussian, a normal map $\mathbf{n}$ can be computed directly by substituting the color $\mathbf{c}_i$ with $(\mathbf{n}_i + \mathbf{1})/2$ in Eq. (4).

Depth attribute: Accurate depth estimation requires computing the intersection between the camera ray and a Gaussian ellipsoid. Many existing methods approximate depth simply as the projected distance of the Gaussian center. This leads to biased estimates, especially near silhouettes and highly specular regions.

To simplify the ray-Gaussian intersection calculation, we adopt the scale regularization loss $\mathcal{L}_{\text{scale}}$ from NeuSG and assign a large weight to it so that it plays a dominant role during training. This loss 'compresses' the 3D Gaussian ellipsoids into extremely flat shapes by driving the smallest scaling component of each Gaussian toward zero. That is, we have

$$\mathcal{L}_{\text{scale}} = \big\|\min(s_1, s_2, s_3)\big\|_1. \tag{7}$$

As a result, each Gaussian can be well approximated by a plane defined by its center $\boldsymbol{\mu}_i$ and its normal $\mathbf{n}_i$.

Following VCR-GauS [14], with this planarity assumption, the intersection between a camera ray $\mathbf{y} = \mathbf{r}t$ (with unit direction $\mathbf{r}$ and distance along ray t) and the Gaussian satisfies $\mathbf{n}_i \cdot (\mathbf{y} - \boldsymbol{\mu}_i) = 0$. Solving for t and projecting onto the camera's z-axis yields the per-pixel depth value

$$d = \frac{\mathbf{n}_i \cdot \boldsymbol{\mu}_i}{\mathbf{n}_i \cdot \mathbf{r}} r_z, \tag{8}$$

where r_z is the z-component of $\mathbf{r}$ in camera coordinates.

Reflection strength: To accurately model the spatially varying specular contributions in deferred rendering, we introduce a scalar attribute called *reflection strength* $w_r \in [0, 1]$. It serves as a per-pixel weight during deferred shading, scaling the contribution of the learned environment map E based on local surface properties.

4.2 Deferred rendering for Gaussian splatting

Conventional view-dependent color representations, such as spherical harmonics (SH), primarily rely on observa-

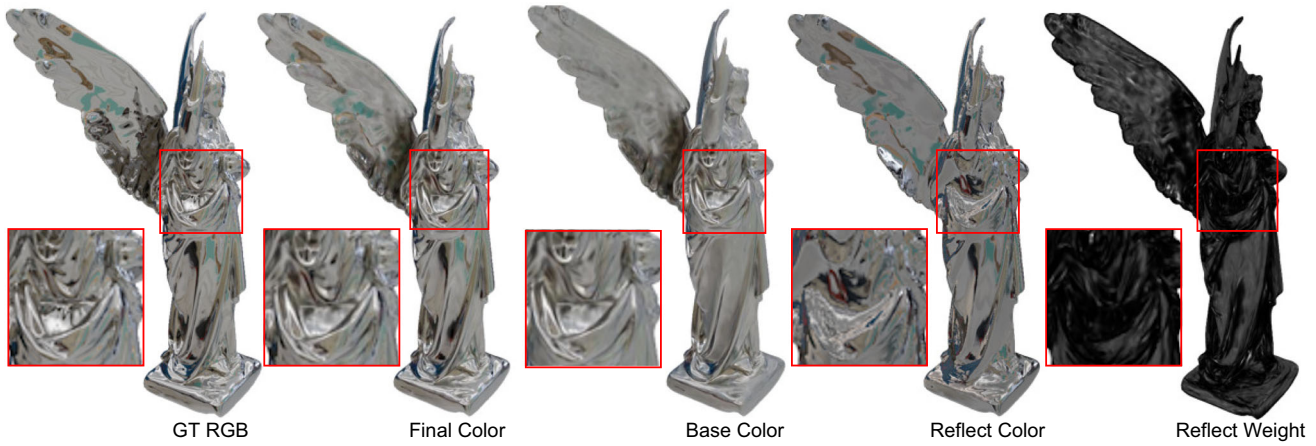


Fig. 2 Visualization of our deferred color composition. The final color is composed of a base color from Gaussian splatting and a specular reflection color from a learnable environment map. Insets marked in red show details

tion direction, which often fails to accurately reconstruct high-frequency specular reflections. In contrast, specular appearance is dominated by the viewing direction and environmental illumination rather than intrinsic object properties. Inspired by 3DGS-DR [21], we adopt a deferred rendering formulation. Instead of querying reflection colors *per Gaussian*, we compute them *per pixel*. This strategy reduces ambiguity and significantly improves computational efficiency. In our deferred pipeline, the final pixel color C is composed of a base term from Gaussian splatting and a specular term from a learnable environment map E as

$$C = \sum_{i=0}^{\mathcal{N}} \mathbf{c}_i \alpha_i \prod_{j=1}^{i-1} (1 - \alpha_j) + w_r E(\hat{\mathbf{r}} - 2(\hat{\mathbf{r}} \cdot \hat{\mathbf{n}})\hat{\mathbf{n}}), \tag{9}$$

where $\hat{\mathbf{r}}$ is the unit ray-incident direction at the pixel and $\hat{\mathbf{n}}$ is the unit normal of the surface rendered at that pixel. To efficiently query the environment map E in a differentiable manner – needed for the subsequent neural network training – we use **Nvdiffrast** [5] to accelerate the sampling process. Integrated within our deferred rendering pipeline, Nvdiffrast enables fast and memory-efficient gradient-based optimization of E. It rasterizes the reflected direction vector $\hat{\mathbf{r}} - 2(\hat{\mathbf{r}} \cdot \hat{\mathbf{n}})\hat{\mathbf{n}}$ for each pixel and returns the sampled color along with its analytical gradient with respect to both direction and texture parameters. This implementation not only ensures real-time performance during inference but also maintains full differentiability, allowing the environment map to be jointly optimized with the Gaussian attributes. Figure 2 shows the results of this process.

4.3 Depth normal supervision

Single-View Regularization: To enforce local geometric consistency, we adopt a normal-consistency constraint between the rendered normal map $\mathbf{n}$ and the depth gradient normals. Under a local-plane assumption, the depth at each pixel defines a surface whose normal can be estimated via finite differences. In more detail: Given the depth function $d(u, v)$ at a pixel with 2D coordinates (u, v) (as computed in Eq. 8), we can estimate the surface's unit normal at that pixel as

$$\mathbf{N}_i = \frac{\partial_u d \times \partial_v d}{\|\partial_u d \times \partial_v d\|}. \tag{10}$$

The single-view geometric loss is then defined as the cosine similarity between the rendered normal $\mathbf{n}_i$ and the depth-derived normal $\mathbf{N}_i$ over all pixels in a frame V to render as

$$\mathcal{L}_{\text{single}} = \sum_{i=1}^{|V|} \left(1 - \mathbf{n}_i^{\mathsf{T}} \mathbf{N}_i\right). \tag{11}$$

Minimizing this loss encourages the rendered normal map to align with the local geometry implied by the depth map, thereby improving local surface coherence.

Multi-View Regularization: While single-view regularization offers a reliable initial geometry, the discrete nature of Gaussian optimization often introduces inconsistencies across different views of the same scene. To ensure global structural coherence, we incorporate a *multi-view geometric consistency prior*. This is particularly important because photometric losses can be unreliable under noise, blur, or weak textures. Unlike SDF-based methods (e.g., NeuS [9] or NeRO [18]), we cannot rely on dense spatial smoothness constraints such as the Eikonal loss [24]. Our multi-view prior mitigates the impact of unreliable image-based updates and helps guide the Gaussians toward globally consistent geometry. We use the reprojection error of the same surface point as a constraint. First, the depth values d_r and d_n for the reference and neighboring views (with an angular separation no

Table 1 Mesh quality in terms of Chamfer Distance (CD)↓ multiplied by 10^2 on the Glossy Synthetic dataset

Method	Angel	Bell	Cat	Horse	Luyu	Potion	Tbell	Teapot	Mean	Time
NeuS	0.85	1.46	2.78	1.53	1.66	3.93	3.48	5.46	2.64	> 12 h
NeRO	**0.40**	**0.36**	**0.52**	**0.40**	**0.63**	**0.53**	**0.54**	**0.51**	**0.48**	> 12 h
2DGS	2.76	2.64	2.97	1.38	1.55	1.84	5.59	2.52	2.65	∼ 15 m
GOF	0.99	9.59	4.11	1.54	2.00	5.20	8.47	5.92	4.73	∼ 2 h
PGSR	0.51	3.00	4.00	0.76	1.03	4.94	6.26	4.74	3.15	∼ 30 m
Ref-GS	0.48	0.70	1.70	0.56	*0.87*	1.36	0.54	1.36	0.95	∼ 30 m
Ours	*0.47*	*0.49*	*1.54*	*0.46*	0.97	*1.03*	*0.51*	*1.29*	*0.85*	∼ 30 m

The best (that is, lowest) distance value is marked in Bold. The second best (second-lowest) is marked in Italic

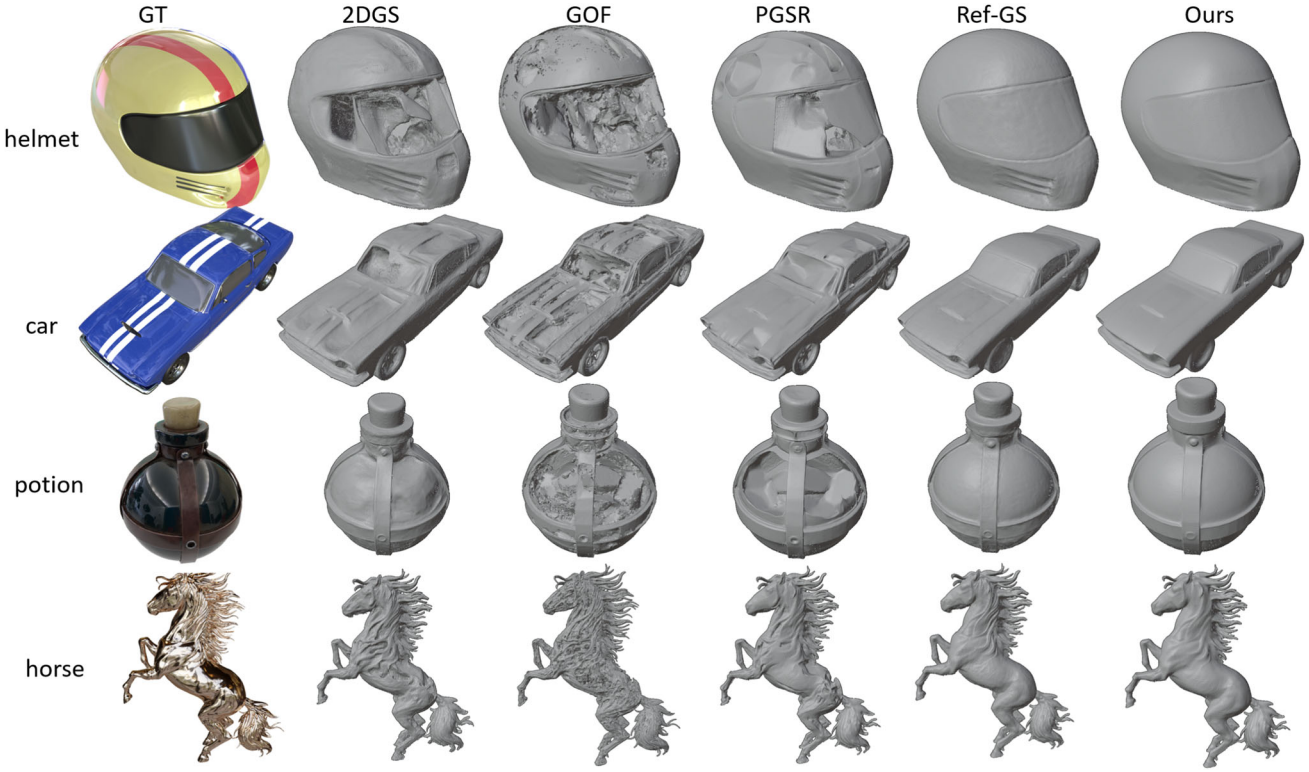


Fig. 3 Visualization of surface reconstruction for reflective objects. Compared to other reconstruction methods (e.g., Ref-GS and PGSR), our approach achieves more accurate surface geometry reconstruction for reflective objects

more than 30°) are obtained using Eq. 8. The depth value d_r of a pixel $\mathbf{p}_r$ is then transformed into a 3D point $\mathbf{P}$ in the world coordinate system as

$$\mathbf{P} = R_r^T (d_r \cdot K_r^{-1} \mathbf{p}_r - T_r), \tag{12}$$

where K_r, R_r and T_r denote the intrinsic matrix, translation vector, and rotation matrix of the reference view, respectively. The spatial point $\mathbf{P}$ is then projected into the neighboring view to obtain

$$\mathbf{p}_n = K_n (R_n \mathbf{P} + T_n), \tag{13}$$

where K_n, R_n, and T_n are the same as K_r, R_r, and T_r, but for the neighboring view. Since the projected coordinate $\mathbf{p}_n$ may not lie exactly on the pixel grid of the neighboring frame, we sample the via bilinear interpolation over the four nearest pixel neighbors of $\mathbf{p}_n$ in the neighboring depth map. Using Eqs. 12 and 13 again, the coordinate $\tilde{\mathbf{p}}_r$ in the reference view is then derived. Minimizing the forward and backward projection errors constitutes the multi-view geometric consistency regularization, computed over all pixels W in the image (excluding those with high forward and backward projection errors). This loss reads as

$$\mathcal{L}_{\text{multi}} = \frac{1}{|W|} \sum_{\mathbf{p}_r \in W} \phi(\mathbf{p}_r), \tag{14}$$

where $\phi(\mathbf{p}_r) = \|\mathbf{p}_r - \tilde{\mathbf{p}}_r\|$ is the forward and backward projection error of $\mathbf{p}_r$. When $\phi(\mathbf{p}_r)$ exceeds a certain threshold (empirically set to $\phi(\mathbf{p}_r) > 1$), we consider that the pixel is occluded or that there is a significant geometric error. To prevent occlusion errors, these pixels are not included in W. Note that, if such pixels are mistakenly identified as occluded due to geometric errors, this does not affect our final convergence. This is because the single-view regularization term and the use of sparse 3D Gaussians to represent dense scenes will gradually propagate high-precision geometry, eventually leading all Gaussians to converge to the correct positions. During training, we use the same combination of L_1 and D-SSIM loss L_{DSSIM} as in 3DGS to yield the final image-rendering loss

$$\mathcal{L}_{\text{rgb}} = L_1 + (1 - \lambda_1)L_{\text{DSSIM}}, \tag{15}$$

where, following 3DGS, we set $\lambda_1 = 0.2$. The combined loss used to train our entire framework is thus

$$\mathcal{L} = \mathcal{L}_{\text{rgb}} + \lambda_2\mathcal{L}_{\text{scale}} + \lambda_3\mathcal{L}_{\text{single}} + \lambda_4\mathcal{L}_{\text{multi}}. \tag{16}$$

Based on experiments, we set $\lambda_2 = 100$, which encourages the 3D Gaussians to become planar by penalizing the smallest scaling component; in order to reconstruct clean geometry, we set $\lambda_3 = 0.2$ and $\lambda_4 = 0.1$.

5 Experiments

Datasets: To validate the effectiveness of our approach, we conduct experiments on three datasets featuring reflective surfaces: two synthetic datasets, Shiny Blender [15] and Glossy Synthetic [18]; and one real-world dataset, Shiny Real [15]. Shiny Blender contains objects with mixed diffuse and specular materials, testing generalization across varied reflectance. Glossy Synthetic focuses on highly specular objects under changing illumination, emphasizing reconstruction of strong view-dependent reflections. Shiny Real contains captured real scenes with complex reflective surfaces under natural lighting, which tests the robustness and practicality of our method in real-world scenarios.

Baselines and metrics: We measure the advantages of our method in terms of reconstruction and rendering quality. For reconstruction quality, we compare with NeRF-based methods, including NeuS [9] and NeRO [18], as well as 3DGS-based methods including 2DGS [3], GOF [13], PGSR [4] and Ref-GS [22]. For rendering quality, we compare with Gaussian-based methods, including 3DGS [2], GaussianShader [20], 3DGS-DR [21] and Ref-GS [22].

For quantitative comparison, we use the several established metrics. We measure rendering quality using PSNR, SSIM [25], and LPIPS [26]. We assess geometric accuracy by the Chamfer Distance, which directly evaluates the fidelity of the reconstructed 3D surface. We gauge our method's speed by reporting average training time and rendering speed in frames per second (FPS). ***Implementation Details:*** To ensure stable convergence, we use a phased training protocol that progressively introduces geometric and reflectance constraints. During the initial stage (typically around 3000 iterations, determined by the shape of point cloud expansion), we use only the original 3DGS RGB loss (Eq. 15) to allow stable point-cloud initialization. In the geometry-aware stage (until iteration 7000), we introduce the single-view consistency loss (Eq. 11) to regularize local surface structure, while also activating the learnable environment map to begin modeling specular reflections. Finally, in the refinement stage (after 7000 iterations), we add the multi-view consistency loss (Eq. 14) to enforce global shape coherence. In all scenes, we set the resolution of the environment map to $6 \times 128 \times 128$. Additionally, to ease the early learning of the environment map, we disable spherical harmonics (SH) by setting their order to 0 for the first 10k iterations, after which we set this to 3. All experiments are run on a single NVIDIA RTX 3090 GPU.

5.1 Comparison results

Geometric quality: Table 1 shows quantitative reconstruction results on the Glossy Synthetic dataset, measured by Chamfer Distance. Our method achieves an average CD of 0.0085 across eight scenes, outperforming all listed 3DGS-based counterparts (e.g., Ref-GS: $CD = 0.0095$). A noticeable gap remains compared to the best implicit model, NeRO ($CD = 0.0050$), which, however, requires over 12 h of training, whereas our full pipeline completes in only 30 min. PGSR-DR thus occupies a favorable accuracy–efficiency trade-off: it delivers markedly better geometry than prior real-time approaches and trains orders of magnitude faster than NeRO. Figure 3 shows the reconstruction of reflective objects by our method *vs* mainstream reconstruction approaches. Due to severe color variations on reflective surfaces, 2DGS, GOF, and PGSR struggle to make Gaussians adhere to the object surface. Ref-GS, which does not cater for multi-view consistency, creates non-smooth meshes. In contrast, our method produces globally consistent and smooth meshes while preserving more details. Figure 5 shows the generalization ability of our method on real-world datasets, where we compare predicted normals with those from PGSR, 3DGS-DR, and Ref-GS. Our method addresses the limitations of PGSR in reconstructing reflective surfaces and the issues of 3DGS-DR in handling real-world scenarios. Also, our method shows superior reconstruction capability over Ref-GS in certain scenes. For instance, in *gardenspheres*, Ref-GS produces blended normals at the object centers; our

Table 2 Quantitative novel view synthesis comparisons on Shiny Blender, Glossy Synthetic and Shiny Real datasets. Best and second best values of the PSNR, SSIM, and LPIPS errors, as well as the frames-per-second (FPS) are highlighted in Bold and Italic, respectively

Dataset	Shiny Blender Dataset							Glossy Synthetic Dataset									Shiny Real Dataset		
	Ball	Car	Coffee	Helmet	Teapot	Toaster	Mean	Angel	Bell	Cat	Horse	Luyu	Potion	Tbell	Teapot	Mean	Garden	Sedan	Toycar
PSNR ↑																			
3DGS	27.76	27.00	31.01	27.35	39.03	21.06	28.87	26.06	25.21	31.43	25.27	26.95	30.23	23.98	21.52	26.33	22.40	*25.56*	24.38
GaussianShader	29.81	28.49	31.32	28.81	43.68	24.94	31.18	28.44	30.35	31.49	26.90	27.53	29.71	25.32	23.42	28.01	20.64	22.82	22.59
3DGS-DR	31.45	29.87	*34.33*	30.96	**46.68**	25.68	33.16	29.28	*32.39*	*32.87*	25.04	28.97	31.21	28.21	25.3	28.89	21.82	24.88	23.56
Ref-GS	**38.71**	**31.47**	**34.87**	**35.84**	*46.18*	*26.97*	**35.67**	**30.02**	**32.40**	**32.89**	**27.74**	**29.50**	**32.34**	**30.84**	**26.74**	**30.31**	**24.14**	**26.50**	**25.34**
Ours	*33.09*	*30.34*	34.03	*31.38*	43.83	**27.08**	*33.29*	*29.37*	31.64	31.74	*27.46*	*29.22*	*31.76*	*28.79*	*25.51*	*29.44*	*22.67*	25.15	*24.64*
SSIM ↑																			
3DGS	0.937	0.924	0.965	0.946	0.994	0.891	0.943	0.908	0.908	0.960	0.903	0.915	0.938	0.900	0.880	0.914	0.590	*0.729*	0.661
GaussianShader	0.960	0.940	0.970	0.955	0.996	0.923	0.957	0.933	0.947	0.961	0.937	0.922	0.937	0.925	0.898	0.933	0.502	0.644	0.583
3DGS-DR	*0.972*	0.960	**0.976**	*0.970*	*0.997*	0.935	0.968	0.923	*0.967*	**0.977**	0.923	0.942	*0.958*	0.951	0.935	0.947	0.581	0.710	0.634
Ref-GS	**0.988**	**0.966**	*0.975*	**0.985**	**0.998**	*0.948*	**0.977**	**0.954**	**0.970**	*0.974*	**0.946**	**0.949**	**0.960**	**0.967**	**0.949**	**0.959**	**0.691**	**0.780**	**0.727**
Ours	*0.972*	*0.961*	0.971	*0.970*	0.996	**0.950**	*0.970*	*0.948*	0.960	0.967	*0.942*	*0.944*	0.953	*0.954*	*0.941*	*0.951*	*0.634*	0.713	*0.681*
LPIPS ↓																			
3DGS	0.159	0.063	0.107	0.102	0.020	0.145	0.099	0.077	0.106	0.062	0.071	0.067	0.092	0.126	0.102	0.088	*0.238*	*0.294*	*0.229*
GaussianShader	0.138	0.047	0.085	0.087	0.011	0.094	0.077	0.062	0.079	0.057	0.054	0.063	0.097	0.117	0.097	0.078	0.341	0.422	0.368
3DGS-DR	*0.123*	0.036	**0.077**	*0.054*	**0.006**	0.091	0.064	0.079	**0.042**	**0.038**	0.051	0.070	**0.065**	*0.066*	0.066	0.060	0.253	0.327	0.250
Ref-GS	**0.076**	**0.031**	*0.079*	**0.027**	**0.006**	*0.072*	**0.049**	**0.045**	**0.042**	*0.040*	**0.043**	**0.045**	*0.070*	**0.049**	**0.060**	**0.049**	**0.181**	**0.197**	**0.177**
Ours	*0.123*	*0.035*	0.084	0.058	0.008	**0.070**	*0.063*	*0.049*	0.052	0.049	*0.048*	*0.050*	0.078	0.075	*0.065*	*0.059*	0.241	0.333	0.240
FPS ↑																			
3DGS	**197.72**	**159.16**	**219.04**	**171.47**	**307.59**	**133.25**	**198.04**	**207.34**	**221.24**	**263.21**	**201.77**	**239.91**	**240.71**	**242.13**	**218.14**	**229.31**	**54.99**	**118.38**	**93.94**
GaussianShader	25.15	21.41	38.06	40.10	83.51	18.41	37.78	28.18	32.73	40.90	22.75	29.65	31.38	24.76	22.68	29.13	5.89	11.63	9.40
3DGS-DR	*148.18*	*130.35*	*112.23*	*146.05*	*164.46*	*89.78*	*131.68*	*134.36*	*130.09*	*128.43*	*130.69*	*138.82*	*80.32*	*138.39*	*135.14*	*127.03*	*75.50*	*64.68*	*80.09*
Ref-GS	1.24	1.85	1.52	1.44	3.26	1.28	1.77	1.45	1.41	1.30	1.42	1.40	1.17	1.16	1.36	1.33	1.06	0.98	1.02
Ours	111.87	86.25	110.13	115.52	118.73	78.77	103.54	88.02	102.50	114.31	73.50	87.02	102.90	107.98	90.41	95.83	19.25	35.52	28.07

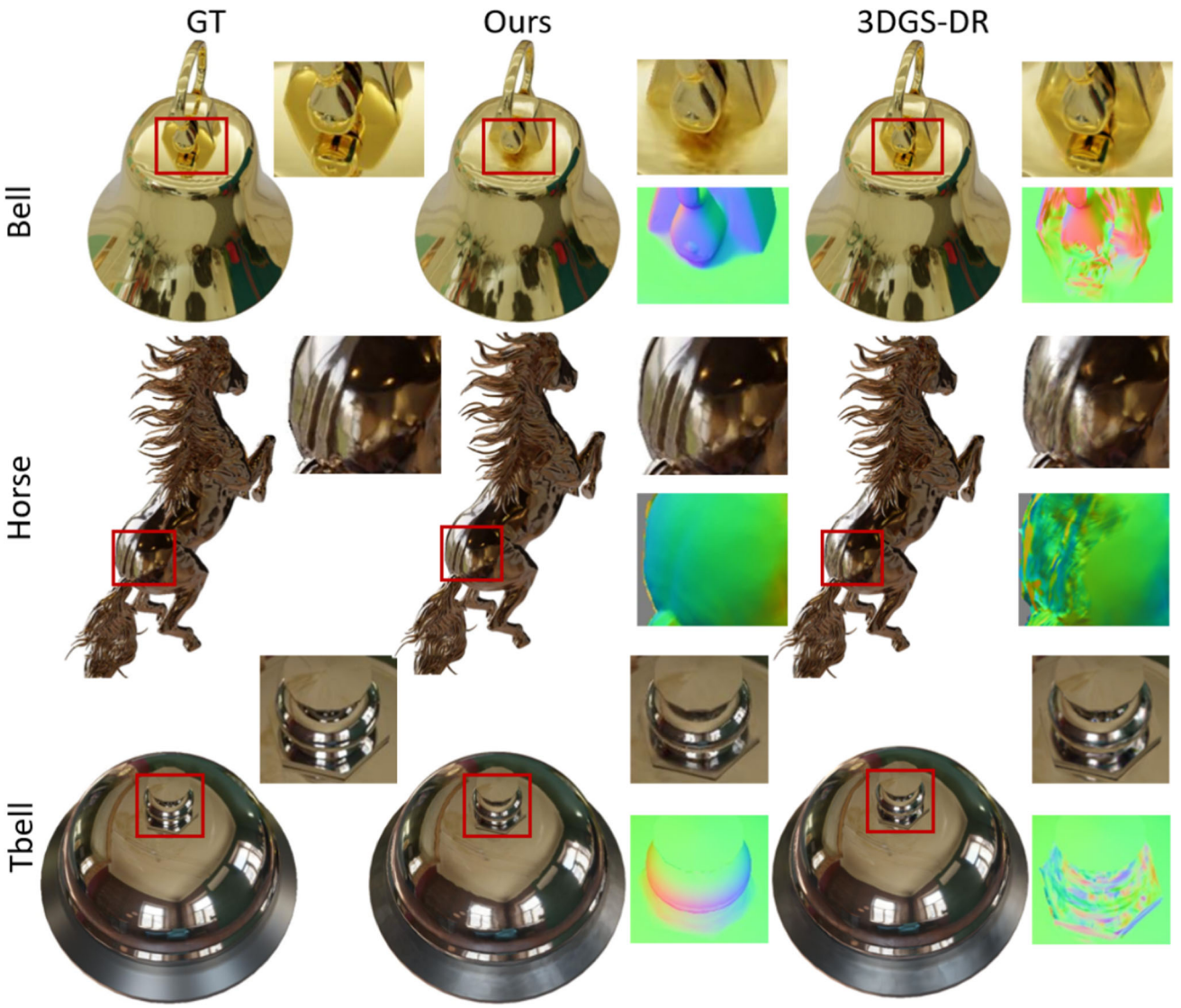


Fig. 4 Qualitative comparisons on reflective scenes. Compared to 3DGS-DR methods, our approach yields more accurate surface geometry predictions for reflective objects

method successfully distinguishes the normals of individual objects.

Novel View Synthesis: We present novel view synthesis results on the Glossy Synthetic, Shiny Blender, and Shiny Real datasets, comparing against 3DGS-based methods. Table 2 reports the quantitative measurements.

Concerning rendering quality, the MLP-based implicit representation of Ref-GS achieves the best PSNR, SSIM, and LPIPS scores, confirming the advantage of implicit methods in modeling high-frequency reflective details. Yet, Ref-GS yields a low speed (FPS), failing real-time rendering requirements. In contrast, the explicit 3DGS-DR representation does real-time rendering but its rendering quality is poor, especially in terms of normal estimation. Our method achieves slightly higher average PSNR than 3DGS-DR across the evaluated datasets, indicating improved reconstruction of view-dependent specular details. This gain comes at the cost of additional computational overhead from our depth-normal consistency constraints and deferred rendering steps. On the Shiny Real dataset, however, our method experiences a notable drop in rendering speed (28.07 vs. 80.09 FPS for 3DGS-DR). This degradation stems from the computational cost of the depth calculation in Eq. 8, which becomes substantial when the number of Gaussian primitives is large.

Figure 4 shows the normal estimation results obtained during rendering, where normal directions are color-coded for interpretability. During rendering, 3DGS-DR estimates normals that exhibit discontinuities and artifacts as it prioritizes image fitting. In contrast, our method consistently reconstructs smooth and well-defined normal fields, yield-

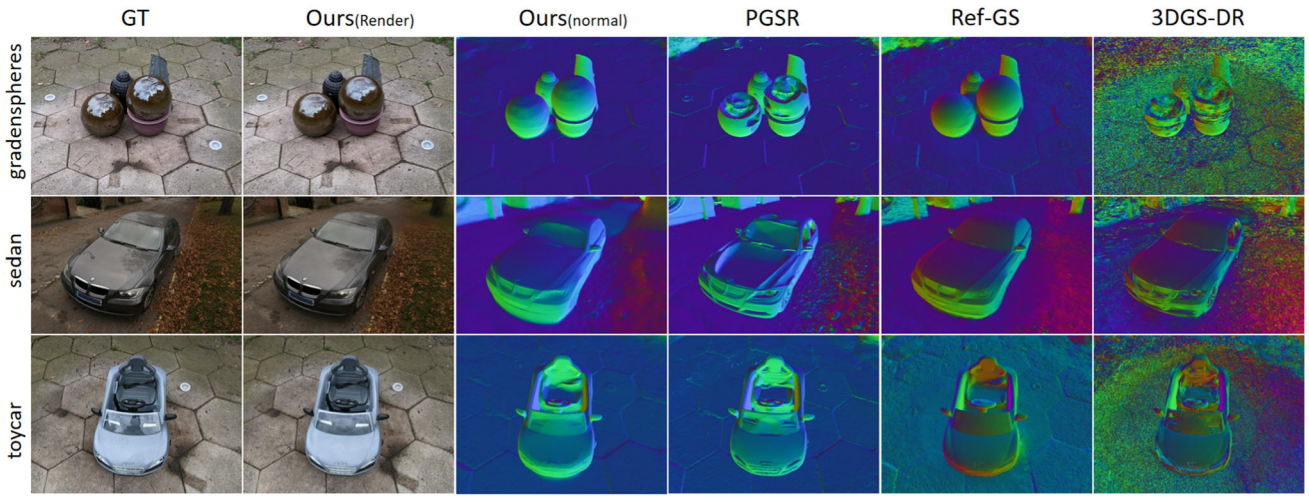


Fig. 5 Visual comparisons of reconstruction results on the Shiny Real dataset. Compared to 3DGS-DR and PGSR, our method successfully reconstructs reflective surfaces, producing results nearly identical to those of Ref-GS

Fig. 6 Visualization of rendering components on a specular object

ing rendered outputs that more closely match the ground truth on scenes such as Horse and tbell. However, this also introduces a notable limitation: the depth-normal consistency constraints can limit the expressiveness of the appearance model in regions with complex occlusions. For instance, in the Bell scene, the pull ring introduces heavy near-field occlusions. To maintain depth-normal consistency, our method sacrifices some flexibility in fitting the reflected colors, whereas 3DGS-DR, with its less constrained normals, more easily overfits the photometric appearance and thus produces visually sharper reflections in these areas. A dynamic visualization is available in the supplementary video. Figure 6 visualizes the decomposition of rendering components on specular objects. The explicit decomposition demonstrates that our method successfully separates view-dependent specular reflections from diffuse surfaces while maintaining geometric accuracy through normal prediction.

5.2 Ablation studies

To analyze the contribution of our various loss terms in Eq. 16, we next perform ablation studies on a scene (Bell)

Fig. 7 Qualitative ablation, Bell scene. See also Table 3

Table 3 Quality metrics for the ablation study, Bell scene. Lower CD values, and higher PSNR values, are better

Model settings	CD ↓	PSNR ↑
w/o deferred shading	0.94	25.03
w/o depth compute	1.37	28.61
w/o $\mathcal{L}_{\text{single}}$	0.96	29.01
w/o $\mathcal{L}_{\text{multi}}$	1.01	30.16
Full model	0.85	29.43

from the Glossy Synthetic dataset. Table 3 reports quantitative results for removing deferred shading, single loss, and multi loss respectively. Corresponding visual comparisons are provided in Fig. 7.

Deferred Shading: Removing deferred shading and reverting to spherical harmonics (SH) for rendering yields a significant performance drop (see Fig. 7), where the reconstructed mesh exhibits noticeable surface fragmentation and reduced geometric coherence. The model without deferred shading yields a PSNR of 25.03 and a CD of 0.94, substantially worse than the full model (PSNR = 29.43, CD = 0.85). This confirms that our deferred rendering mechanism is crucial for accurately modeling the view-dependent colors of reflective objects and for achieving coherent geometric reconstruction.

Depth Compute: Removing our depth computation in Eq. (8) and reverting to the conventional alpha-blended depth in Eq. (5) causes the most severe geometric degradation among all ablations, with CD increasing from 0.85 to 1.37 (see Table 3). This confirms that the conventional depth estimator, which treats Gaussian centers as discrete point samples, introduces substantial bias that misaligns primitives from the true surface. The resulting geometric errors propagate to appearance modeling, as reflected by the PSNR drop from 29.43 to 28.61. We conclude that accurate depth estimation serves as the foundation of our joint optimization framework. Without it, the subsequent single-view and multi-view regularizations lack a reliable geometric signal to operate on.

Single-View Regularization: The single-view regularization enforces depth–normal consistency under a local-plane assumption, aligning rendered normals with geometry derived from depth gradients. As Table 3 shows, removing this constraint raises the CD from 0.85 to 0.96 while PSNR remains almost unchanged, indicating that the $\mathcal{L}_{\text{single}}$ term (Eq. 11) improves local surface accuracy without affecting rendering fidelity. In Fig. 7, this geometric decline is evident as surface perturbations appear at the handle-ring junction, confirming that the single-view prior is necessary for recovering locally consistent geometry.

Multi-View Regularization: Removing the multi-view consistency loss $\mathcal{L}_{\text{multi}}$ (Eq. 14) leads to a sharp CD increase (0.85 to 1.01, see Table 3), telling severe degradation in global structural coherence. This is further visible in Fig. 7, where the corresponding mesh exhibits visible surface concavity and discontinuity, especially around joint regions such as the handle-ring connection. These results confirm that our multi-view prior is essential for maintaining consistent geometry across viewpoints and for recovering globally smooth surfaces.

6 Conclusion

In this paper, we presented PGSR-DR, a framework that extends 3D Gaussian Splatting with planar-based geometry regularization and deferred rendering to tackle reflective surface reconstruction. Our method builds on PGSR's planar representation to improve depth estimation, enforces joint depth-normal and multi-view consistency for geometric coherence, and incorporates a learnable environment map to capture view-dependent specular effects. Experimental results demonstrate that PGSR-DR successfully reconstructs reflective objects, overcoming the failure of PGSR [4] on specular surfaces. Compared to 3DGS-DR [21] and Ref-GS [22], our method achieves a favorable balance between rendering quality and speed, delivering real-time performance with competitive visual fidelity.

Several limitations motivate future work. A primary limitation lies in the material-lighting ambiguity, where our framework captures complex reflections without explicitly decomposing the spatially-varying BRDF parameters from environmental illumination. Future work could incorporate more rigorous physically-based rendering models to further decouple surface roughness and metallicity. Additionally, modeling indirect illumination and inter-reflections between multiple metallic objects remains a non-trivial task for Gaussian-based representations, potentially requiring the integration of recursive sampling or irradiance fields. Finally, as shown in Fig. 5, the reconstruction of transparent objects also presents a significant challenge that remains difficult for 3DGS-based methods to address.

Appendix A Surface approximation via planar Gaussians

To assess how the planarity assumption affects curved geometry, we compare two variants of our method on a metallic ball and a car scene from the GlossyBlender dataset: **PGSR-DR (normal)**, which retains the full 3D covariance without flattening, and **PGSR-DR (planar)**, which compresses the smallest scaling axis toward zero. Unconstrained 3DGS serves as a baseline. Table 4 reports the results.

On the Ball scene, the planar variant requires more primitives than the normal variant (186K *vs.* 159K), as each flattened Gaussian covers less angular extent on the spherical surface. This yields a marginal PSNR gain (33.11 *vs.* 32.99). On the Car scene, which represents common objects with mixed curvature, both variants use comparable primitive counts (288K *vs.* 281K) and achieve nearly identical quality. In both scenes, our variants substantially outperform 3DGS in PSNR. We conclude that the planar assumption incurs a controllable overhead only on surfaces with sustained high curvature such as the ball. For typical objects like the car, planarity introduces no meaningful primitive increase while preserving improved rendering fidelity.

Appendix B Reconstruction quality: comparison with NeRO

We conduct a direct visual comparison with NeRO [18], the strongest offline baseline in Table 1. NeRO achieves a Chamfer Distance of 0.0050 on the Glossy Synthetic dataset, outperforming our method (CD 0.0085) and Ref-GS (CD 0.0095) by a clear margin. As shown in Fig. 8, NeRO recovers finer surface details, particularly around concave regions and sharp edges where our planar-based representation loses some fidelity. Our method, while coarser than NeRO, produces results that are visibly smoother and more globally coherent than those of Ref-GS, which exhibits surface noise due to its lack of multi-view consistency constraints. This comparison places PGSR-DR at an intermediate position: it sacrifices some geometric precision relative to state-of-the-art offline methods in exchange for substantially faster training (30 min *vs.* 12+ hours).

Table 4 Approximation efficiency of planar Gaussians on different surface types. The planar variant requires more primitives on the highly curved ball but shows no notable increase on the car

Scene	Method	#Gaussians	PSNR	SSIM	FPS
Ball	3DGS	97K	27.76	0.936	197.71
	PGSR-DR (normal)	159K	32.99	0.972	116.40
	PGSR-DR (planar)	186K	33.11	0.972	114.61
Car	3DGS	292K	27.00	0.923	159.16
	PGSR-DR (normal)	281K	30.21	0.961	89.09
	PGSR-DR (planar)	288K	30.34	0.961	89.60

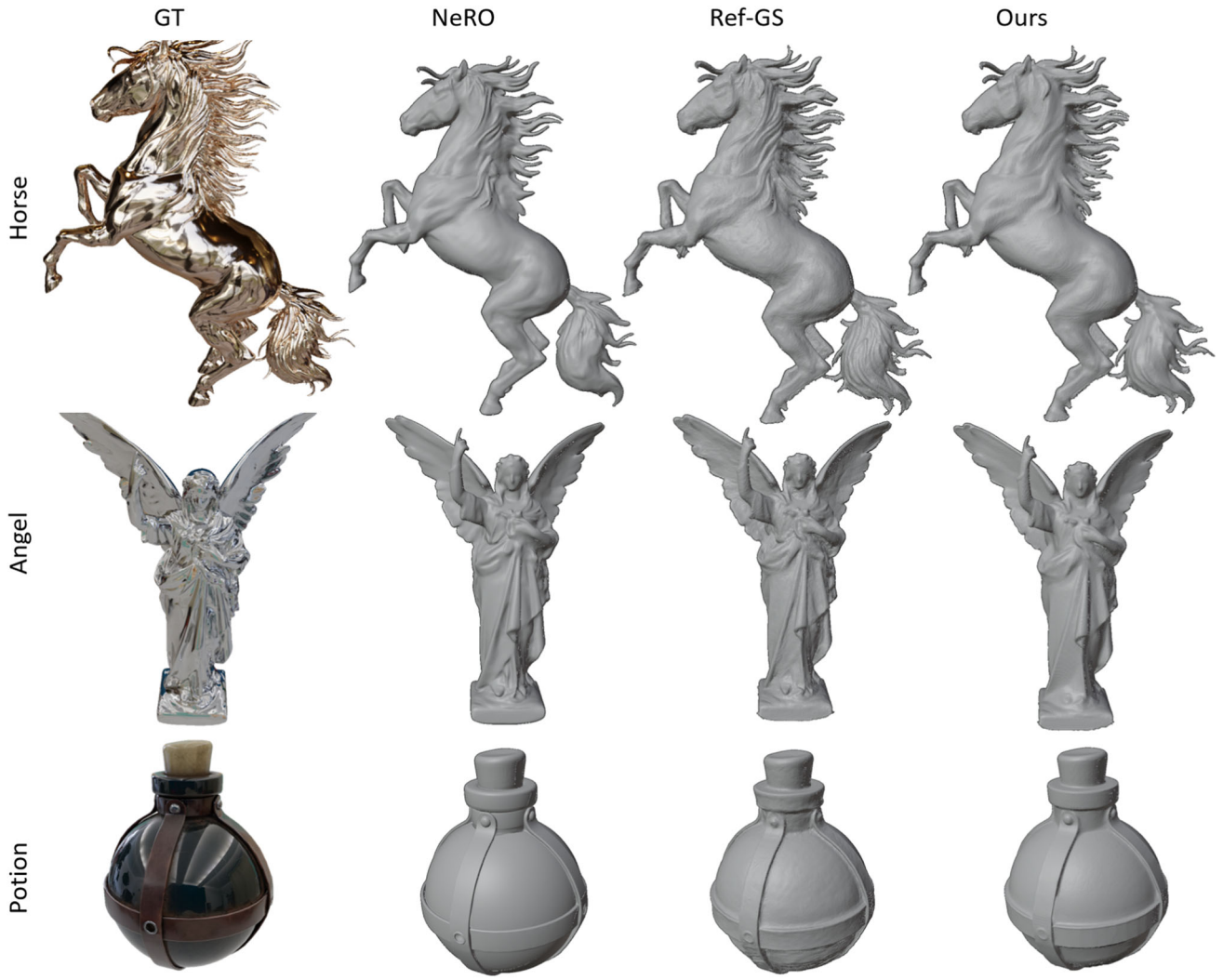


Fig. 8 Visual comparison with NeRO on the Glossy Synthetic dataset. NeRO captures finer surface details but requires over 12 h of training; our method produces smoother surfaces than Ref-GS

Appendix C Failure cases

We presented two cases of reconstruction failure, illustrated in Fig. 9. First, without explicit occlusion reasoning, Gaussians near self-occluded concave boundaries may converge to incorrect positions that partially explain specular highlights, distorting the geometry. As shown in Fig. 9a, near concave boundaries of metallic objects, Gaussians may converge to incorrect spatial positions that partially explain the observed specular highlights without recovering the true occluding surface. Second, the opacity assumption fails on transparent surfaces such as the *toycar* windshield. From different viewpoints, the windshield alternately appears reflective or transparent, causing the accumulated depth via alpha-blending to vary inconsistently across views. Our depth formulation, which assumes a single opaque surface along each ray, cannot correctly represent this view-dependent behavior, leading to erroneous depth estimates and, consequently, poorly converged normals. Both failure modes highlight the need for occlusion-aware and transparency-capable Gaussian representations.

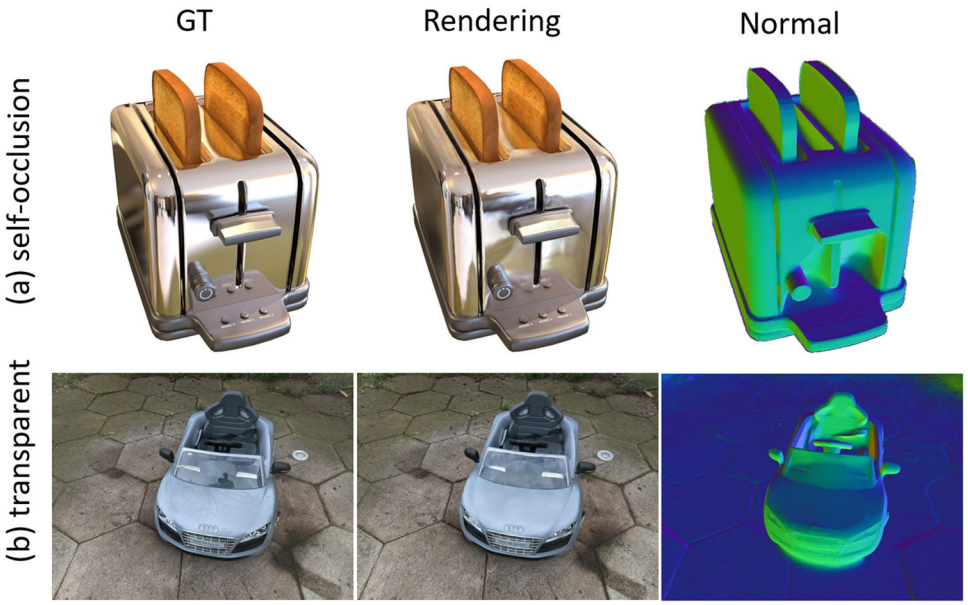


Fig. 9 Representative failure case. **a** Self-occlusions cause Gaussians to converge to incorrect positions, resulting in distorted geometry. **b** Transparent objects lead to inaccurate depth estimation and prevent normal convergence

Appendix D Multi-scene ablation study

To analyze the contribution of each component, we conduct ablation studies on two additional scenes (Potion and Teapot) from the Glossy Synthetic dataset. Figure 10 provides visual comparisons. Removing depth computation (reverting to conventional alpha-blended depth) leads to the most severe geometric degradation across all scenes, confirming that

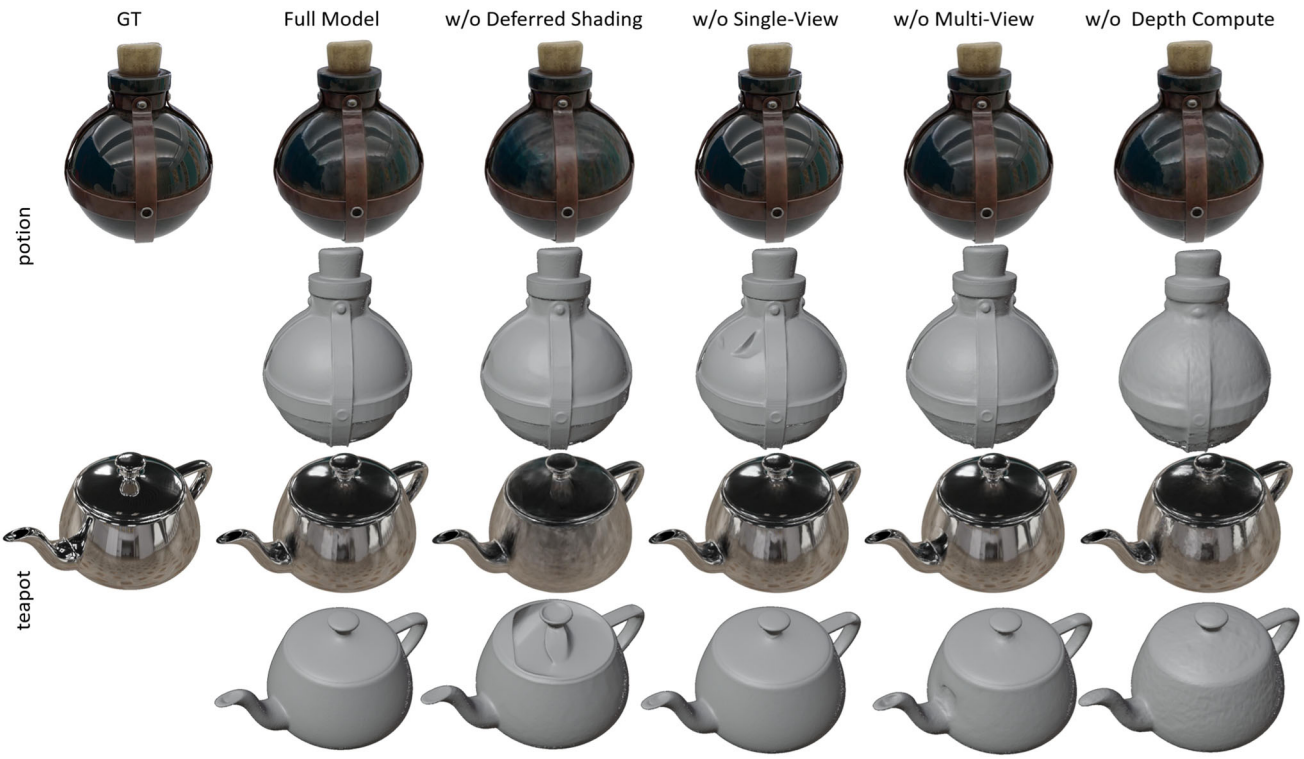


Fig. 10 Qualitative ablation on Potion and Teapot scenes

accurate depth estimation is the foundation of our joint optimization framework. Both single-view and multi-view regularizations contribute to surface smoothness and global coherence, with the multi-view term playing a particularly important role in preventing surface concavity and discontinuity. Deferred shading mainly affects rendering fidelity: its removal consistently causes a sharp drop in image quality, although the geometric impact varies by scene. Overall, the relative contribution of each component remains consistent across scenes, supporting the generality of our design.

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/s00371-026-04610-y.

Acknowledgements This research was funded by the New Generation Artificial Intelligence-National Science and Technology Major Project (Grant No. 2025ZD0123900), the National Natural Science Foundation of China (Grant Nos. 62376025 and 62332017), and the Key Research and Development Program of Hainan Province (Grant No. ZDYF2024GXJS032). The authors sincerely acknowledge the financial support from these funding agencies and thank them for their contribution to this work.

Author Contributions J.L. performed the main research work, including methodology design, algorithm implementation, experiments, data analysis, and drafted the manuscript. X.W., J.K., A.C.T., and Y.Z. supervised the research, contributed to the conceptualization and methodology, and critically revised the manuscript. H.Z., X.Y., and Y.X. assisted in data collection, experimental validation, and preparation of figures. All authors reviewed and approved the final manuscript.

Data Availability The datasets that support the findings of this study are publicly available. Shiny Blender and Shiny Real datasets can be accessed at https://dorverbin.github.io/refnerf/. Glossy Synthetic dataset can be accessed at https://liuyuan-pal.github.io/NeRO/.

Declarations

Conflict of interest The authors declare no conflict of interest.

References

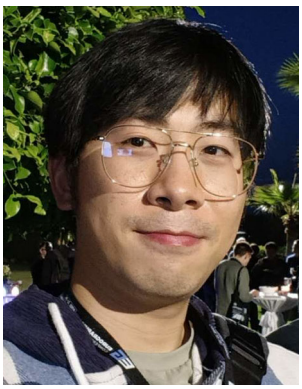
1. Mildenhall, B., Srinivasan, P.P., Tancik, M., Barron, J.T., Ramamoorthi, R., Ng, R.: NeRF: representing scenes as neural radiance fields for view synthesis. Commun. ACM **65**(1), 99–106 (2021)
2. Kerbl, B., Kopanas, G., Leimkühler, T., Drettakis, G., et al.: 3D Gaussian splatting for real-time radiance field rendering. ACM Trans. Graph. **42**(4), 139–1 (2023)
3. Huang, B., Yu, Z., Chen, A., Geiger, A., Gao, S.: 2D Gaussian splatting for geometrically accurate radiance fields. In: ACM SIGGRAPH 2024 Conference Papers, pp. 1–11 (2024)
4. Chen, D., Li, H., Ye, W., Wang, Y., Xie, W., Zhai, S., Wang, N., Liu, H., Bao, H., Zhang, G.: PGSR: planar-based Gaussian splatting for efficient and high-fidelity surface reconstruction. IEEE Trans. Visual Comput. Graphics **31**(9), 6100–6111 (2024)
5. Laine, S., Hellsten, J., Karras, T., Seol, Y., Lehtinen, J., Aila, T.: Modular primitives for high-performance differentiable rendering. ACM Trans. Graph. (ToG) **39**(6), 1–14 (2020)
6. Choy, C.B., Xu, D., Gwak, J., Chen, K., Savarese, S.: 3D-R2N2: a unified approach for single and multi-view 3D object reconstruction. In: European Conference on Computer Vision, pp. 628–644 (2016). Springer
7. Fan, H., Su, H., Guibas, L.J.: A point set generation network for 3d object reconstruction from a single image. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 605–613 (2017)
8. Park, J.J., Florence, P., Straub, J., Newcombe, R., Lovegrove, S.: DeepSDF: learning continuous signed distance functions for shape representation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 165–174 (2019)
9. Wang, P., Liu, L., Liu, Y., Theobalt, C., Komura, T., Wang, W.: NeuS: learning neural implicit surfaces by volume rendering for multi-view reconstruction. arXiv preprint arXiv:2106.10689 (2021)
10. Yariv, L., Gu, J., Kasten, Y., Lipman, Y.: Volume rendering of neural implicit surfaces. Adv. Neural. Inf. Process. Syst. **34**, 4805–4815 (2021)
11. Li, Z., Müller, T., Evans, A., Taylor, R.H., Unberath, M., Liu, M.-Y., Lin, C.-H.: Neuralangelo: High-Fidelity Neural Surface Reconstruction. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 8456–8465 (2023)
12. Guédon, A., Lepetit, V.: SuGaR: surface-aligned Gaussian splatting for efficient 3D Mesh Reconstruction and High-Quality Mesh Rendering. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 5354–5363 (2024)
13. Yu, Z., Sattler, T., Geiger, A.: Gaussian opacity fields: efficient adaptive surface reconstruction in unbounded scenes. ACM Trans. Graph. (ToG) **43**(6), 1–13 (2024)
14. Chen, H., Wei, F., Li, C., Huang, T., Wang, Y., Lee, G.H.: VCR-GauS: view consistent depth-normal regularizer for Gaussian surface reconstruction. Adv. Neural. Inf. Process. Syst. **37**, 139725–139750 (2024)
15. Verbin, D., Hedman, P., Mildenhall, B., Zickler, T., Barron, J.T., Srinivasan, P.P.: Ref-NeRF: structured view-dependent appearance for neural radiance fields. IEEE Trans. Pattern Anal. Mach. Intell. **47**(11), 9426–9437 (2024)
16. Verbin, D., Srinivasan, P.P., Hedman, P., Mildenhall, B., Attal, B., Szeliski, R., Barron, J.T.: NeRF-casting: improved view-dependent appearance with consistent reflections. In: SIGGRAPH Asia 2024 Conference Papers, pp. 1–10 (2024)
17. Liang, R., Chen, H., Li, C., Chen, F., Panneer, S., Vijaykumar, N.: ENVIDR: implicit differentiable renderer with neural environment lighting. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 79–89 (2023)
18. Liu, Y., Wang, P., Lin, C., Long, X., Wang, J., Liu, L., Komura, T., Wang, W.: NeRO: neural geometry and BRDF reconstruction of reflective objects from multiview images. ACM Trans. Graph. (ToG) **42**(4), 1–22 (2023)
19. Tang, Z.J., Cham, T.-J.: 3iGS: factorised tensorial illumination for 3D Gaussian splatting, pp. 143–159. Springer (2024). (**European Conference on Computer Vision**)
20. Jiang, Y., Tu, J., Liu, Y., Gao, X., Long, X., Wang, W., Ma, Y.: 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)
21. Ye, K., Hou, Q., Zhou, K.: 3D Gaussian splatting with deferred reflection. In: ACM SIGGRAPH 2024 Conference Papers, pp. 1–10 (2024)
22. Zhang, Y., Chen, A., Wan, Y., Song, Z., Yu, J., Luo, Y., Yang, W.: Ref-GS: directional factorization for 2D Gaussian splatting. In: Proceedings of the Computer Vision and Pattern Recognition Conference, pp. 26483–26492 (2025)

23. Chen, H., Li, C., Wang, Y., Lee, G.H.: NeuSG: neural implicit surface reconstruction with 3D Gaussian splatting guidance (2023). arXiv preprint arXiv:2312.00846
24. Gropp, A., Yariv, L., Haim, N., Atzmon, M., Lipman, Y.: Implicit geometric regularization for learning shapes (2020). arXiv preprint arXiv:2002.10099
25. Wang, Z., Bovik, A.C., Sheikh, H.R., Simoncelli, E.P.: Image quality assessment: from error visibility to structural similarity. IEEE Trans. Image Process. **13**(4), 600–612 (2004)
26. Zhang, R., Isola, P., Efros, A.A., Shechtman, E., Wang, O.: The unreasonable effectiveness of deep features as a perceptual metric. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 586–595 (2018)

Jingfeng Li received the undergraduate diploma from Zhongyuan University of Technology in 2024. He is currently working toward the M.S. degree in computer science at the University of Science and Technology Beijing. His research interests include 3D reconstruction and computer graphics.

Xiaokun Wang received the PhD degree in computer science and technology from the University of Science and Technology Beijing, in 2017. He is a professor in intelligence science and technology from the University of Science and Technology Beijing, China. From 2021 to 2023, he worked with the National Centre for Computer Animation, Bournemouth University, supported by the EU Horizon 2020 Marie Curie Individual Fellowship. His research interests include computer graphics, virtual reality, and human–computer interaction.

Haokai Zeng received the undergraduate diploma from the University of Science and Technology Beijing, in 2023. He is currently a postgraduate student in artificial intelligence science and engineering at the University of Science and Technology Beijing. His research interests include computer graphics, especially physically based fluid simulation.



Xingyu Ye received the BEng degree in vehicle engineering from Chongqing University, in 2019, and the MEng degree in vehicle engineering from Zhejiang University, in 2022. He is currently working toward the PhD degree with National Centre for Computer Animation, Bournemouth University, UK. His research interests include physics-based fluid simulation and neural fluid simulation.

Jiří Kosinka received the PhD and MSc degrees from Charles University, in Prague, Czech Republic, in2006 and 2002, respectively. He is currently an associate (adjunct) professor with the Bernoulli Institute, University of Groningen, the Netherlands, where he leads the Scientific Visualization and Computer Graphics research group. His research interests include geometric modeling, computer-aided design, computer graphics, and image processing.

Alexandru C. Telea received the PhD degree in computer science from the Eindhoven University of Technology, 2000. He was assistant professor in visualization and computer graphics with the Eindhoven University of Technology (until 2007) and then full professor of visualization with the University of Groningen. Since 2019, he is full professor of visual data analytics with Utrecht University. His interests include high dimensional visualization, visual analytics, and image-based information visualization.

Yalan Zhang received the PhD degree in computer science and technology from the University of Science and Technology Beijing. She is an associate professor with the School of Intelligence Science and Technology, University of Science and Technology Beijing. Her research interests include computer graphics, artificial intelligence, and 3D visualization.

Yanrui Xu received the PhD degree in computer science and technology from the University of Science and Technology Beijing, in 2025. He is a postdoctoral researcher with Tsinghua University. His research interest includes focus on physics-based fluid simulation.