

# MGSR: 2D/3D Mutual-boosted Gaussian Splatting for High-fidelity Surface Reconstruction under Various Light Conditions

Qingyuan Zhou<sup>1</sup>, Yuehu Gong<sup>1</sup>, Weidong Yang<sup>1\*</sup>, Jiaze Li<sup>2</sup>, Yeqi Luo<sup>1</sup>, Baixin Xu<sup>2</sup>,  
 Shuhao Li<sup>1</sup>, Ben Fei<sup>3\*</sup>, Ying He<sup>2</sup>

<sup>1</sup>Fudan University, <sup>2</sup>Nanyang Technological University, <sup>3</sup>The Chinese University of Hong Kong

zhouqy23@m.fudan.edu.cn, wdyang@fudan.edu.cn, benfei@cuhk.edu.hk, yhe@ntu.edu.sg

## Abstract

*Novel view synthesis (NVS) and surface reconstruction (SR) are essential tasks in 3D Gaussian Splatting (3DGS). Despite recent progress, these tasks are often addressed independently, with GS-based rendering methods struggling under diverse light conditions and failing to produce accurate surfaces, while GS-based reconstruction methods frequently compromise rendering quality. This raises a central question: must rendering and reconstruction always involve a trade-off? To address this, we propose MGSR, a 2D/3D Mutual-boosted Gaussian Splatting for Surface Reconstruction that enhances both rendering quality and 3D reconstruction accuracy. MGSR introduces two branches—one based on 2DGS and the other on 3DGS. The 2DGS branch excels in surface reconstruction, providing precise geometry information to the 3DGS branch. Leveraging this geometry, the 3DGS branch employs a geometry-guided illumination decomposition module that captures reflected and transmitted components, enabling realistic rendering under varied light conditions. Using the transmitted component as supervision, the 2DGS branch also achieves high-fidelity surface reconstruction. Throughout the optimization process, the 2DGS and 3DGS branches undergo alternating optimization, providing mutual supervision. Prior to this, each branch completes an independent warm-up phase, with an early stopping strategy implemented to reduce computational costs. We evaluate MGSR on a diverse set of synthetic and real-world datasets, at both object and scene levels, demonstrating strong performance in rendering and surface reconstruction. Code is available at <https://github.com/TsingyuanChou/MGSR>.*

## 1. Introduction

3D Gaussian Splatting (3DGS) [9] has recently gained great attention in computer graphics and 3D vision [3]. By

representing scenes as collections of 3D Gaussian primitives, 3DGS offers a more flexible and adaptive representation, enabling accurate and efficient rendering and visualization without relying on neural networks. 3DGS effectively resolves the issues of low training efficiency and insufficient geometric accuracy in previous NeRF methods. Due to its impressive rendering efficiency and accuracy, 3DGS has been widely applied in fields such as SLAM [15], dynamic real-world or large-scale scene modeling [16, 20], reconstruction [11, 18], navigation [28], manipulation [13, 14, 19], 3D generation [2, 29], and 3D human simulation [17], etc.

Nevertheless, achieving high-fidelity surface reconstruction (SR) and improving the realism of novel view synthesis (NVS) under various light conditions are two main challenges in 3DGS. Specifically, considering that GS-based rendering methods are affected by different light conditions, previous methods [4, 8] proposed to decompose light to enhance the neural rendering in scenes containing reflective surfaces with the introduction of BRDF (Figure 1a). However, despite the effectiveness of illumination decomposition in rendering, these methods are time-consuming and still struggle to achieve meaningful mesh extraction due to inherent limitations in 3DGS. For instance, the centers of Gaussian primitives do not align well with the surface. On the other hand, GS-based SR methods [5, 6, 24] endeavor to constrain the Gaussian primitives close to the surface, inevitably sacrifice rendering quality and are sensitive to light conditions, which may cause surface artifacts on surfaces (Figure 1b). For example, 2DGS [6] utilize 2D Gaussian primitives for 3D scene representation, facilitating accurate and view-consistent geometry modeling. However, 2DGS still cannot effectively model surfaces when ambient lighting changes.

To solve these contradictions, we propose MGSR, a 2D/3D Mutual-boosted Gaussian splatting for Surface Reconstruction that enhances both rendering quality and 3D reconstruction accuracy (Figure 1c). Specifically, two branches are devised based on 2DGS and 3DGS, which are

\*Corresponding authors

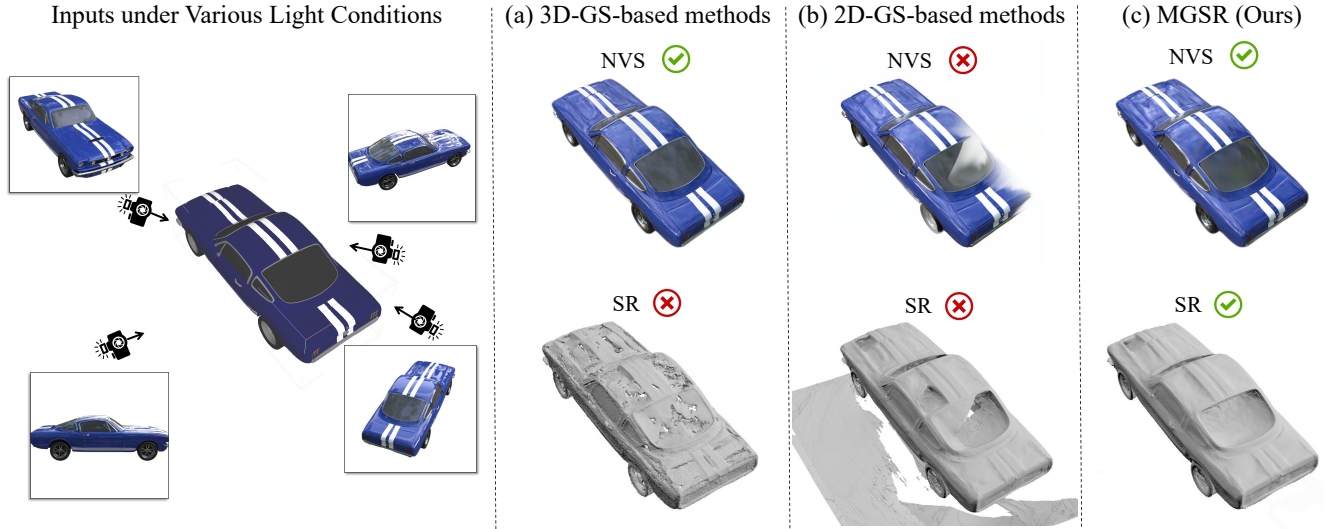


Figure 1. MGSR achieves strong NVS and SR results compared with methods based on 2DGS [6] and 3DGS [24]. The input consists of multi-view images captured from various camera positions and angles, under significantly varying light conditions. In some views, the object appears well-lit and clearly visible, while in others, it is poorly lit, resulting in shadows, reflections, and inconsistencies across the images.

optimized synchronously and mutually enhance each other: the 2DGS branch provides accurate geometric information, with its limited rendering capabilities improved by 3DGS branch, while the 3DGS branch focuses on rendering, with the cost of geometric accuracy compensated by the 2DGS branch. With geometry information, a geometry-guided illumination decomposition module is devised in the 3DGS branch to obtain the reflected and transmitted components, achieving realistic rendering under various light conditions. To achieve this, an additional Spherical Harmonic (SH) is introduced to model reflected colors, along with two other parameters: reflected opacity and reflected confidence, to represent the reflected component accurately. Using the transmitted component as supervision, the 2DGS branch can achieve high-fidelity SR while avoiding the impact of illumination factors on surface estimation. Throughout the mutual-boosted supervision stage, the 2DGS and 3DGS branches engage in alternating optimization for mutual supervision. Prior to alternating optimization, the two modules undergo an independent warm-up stage, and an auto-stop strategy is introduced to reduce unnecessary computational burdens. To the best of our knowledge, MGSR is the first GS-based approach that investigates the simultaneous enhancement of rendering and reconstruction, as well as the first mutual-boosted work on GS involving both 2DGS and 3DGS. MGSR is thoroughly evaluated across a wide range of synthetic and real-world datasets, as well as object- and scene-level datasets, showcasing its superior performance in both rendering and mesh extraction. In summary, our main contributions are as follows:

- The first to explore the feasibility of the joint promotion between rendering and reconstruction, introducing a GS-

based 2D/3D mutual-boosted approach that ensures rendering quality while achieving high-fidelity SR under various light conditions.

- The 2DGS branch aims to provide geometry information to enhance the illumination decomposition of the 3DGS. The decomposed transmitted color will be in turn utilized to supervise the 2DGS branch for improved surface reconstruction, independent of varying light conditions.
- Leveraging geometric information from the 2DGS branch, we have developed a geometry-guided illumination decomposition module to enhance the accuracy of decomposing reflected and transmitted components, achieving more realistic rendering outcomes.
- To address the varying convergence speeds in the two branches during the warm-up stage, an auto-stop strategy has been devised. This strategy involves initiating alternating optimization once one branch has completed its warm-up stage.

## 2. Related work

**Lighting estimation and decomposition.** Estimating and decomposing light conditions in 3D scenes is a challenging task, further complicated by factors such as reflections, refractions, overexposure, and diverse material properties, resulting in problems related to multi-view inconsistency. NeRFactor [26] addresses spatially varying reflectance and environmental lighting using a re-rendering loss, smoothness priors, and data-driven BRDF prior learned from real-world measurements. NEILF [23] represents scene lighting as Neural Incident Light and models material properties as surface BRDF. NEILF++ [25] integrates incident and out-

going light fields through physically-based rendering and surface inter-reflections. Ref-NeRF [21] replaces view-dependent emitted radiance parameterization of NeRF with a representation of reflected radiance. However, NeRF-based methods are primarily constrained by their significant computational burdens and relatively slow rendering speed.

Recently, several attempts have also been made at lighting estimation and decomposition on 3DGS. R3DG [4] leverages 3DGS and NEILF [23] to create a ray-tracing and relighting-capable 3DGS representation. GShader [8] presents a simplified shading function for reflective surfaces in 3DGS, where a novel normal estimation is introduced that uses the shortest axis direction of the 3D Gaussian as an approximate normal, eliminating reliance on depth map priors and avoiding the flattening or 2D projection of 3D Gaussians. GS-IR [12] utilizes depth-derivation-based regularization for normal estimation and a baking-based method for modeling indirect lighting, enabling a precise decomposition of material attributes and illumination, thereby significantly improving the photorealism of the rendered outcomes. The prevailing focus in current efforts on lighting estimation and decomposition is predominantly on achieving photorealistic rendering, which consequently hinders the ability to carry out surface mesh extraction effectively.

**3DGS driven surface reconstruction.** The misalignment of the centroids (means) of 3D Gaussians with the actual surfaces presents a challenge for accurately reconstructing surfaces. A common strategy is to flatten the Gaussian spheres [5] or utilize 2D Gaussian disks [6], effectively pulling the Gaussian centers closer to the surfaces of the object. SuGaR [5] incorporates a regularization term that enhances the alignment between Gaussians and surfaces within the scene to improve the accuracy of normal estimation. 2DGS [6] adopts flattened 2D Gaussians to represent 3D scenes and defines the normal as the direction of the steepest change in density of the 2D Gaussian distribution. GOF [24] utilizes ray-tracing-based volume rendering and establishes a level set by opacities of 3D Gaussians. Surface normals of the Gaussians are approximated using the intersection plane between the ray and the Gaussian, enabling regularization that significantly improves the geometry. These methods solely focus on scenes with consistent light conditions, which often results in the reconstructed mesh exhibiting significant holes and surface inaccuracies. Unlike previous methods, PGSR [1] takes the light conditions into account and incorporates exposure compensation to enhance the accuracy. PGSR flattens the Gaussian into a planar shape and introduces unbiased depth estimation for extracting geometric parameters for surface reconstruction. However, it may result in over-smoothness in the highlight area, and be challenging in reconstructing reflective or mirror surfaces. Therefore, to reconstruct meshes under varying light conditions, we propose a method capable of accu-

rate textured mesh extraction in diverse light scenarios.

### 3. Method

#### 3.1. Overview

MGSR is a 2D/3D mutual-boosted framework that consists of two branches: improved 3DGS branch (Section 3.2) and 2DGS branch (Section 3.3). Initially, each branch undergoes an independent warm-up phase, after which they engage in alternating optimization through mutual-boosted supervision (Section 3.4).

The 3DGS branch is designed to perform illumination decomposition by introducing reflection-related parameters to disentangle reflections and handle overexposure. However, accurate illumination decomposition relies on reliable geometry, which the 3DGS branch alone cannot provide. To address this limitation, we introduce a geometry-guided illumination decomposition module, which leverages depth information from the 2DGS branch to enhance rendering performance under diverse light conditions.

Specifically, the 2DGS branch aligns 2D Gaussian disks with the surface, capturing initial and relatively accurate geometry information even under significantly varying light conditions. In the subsequent alternating optimization phase, the 3DGS branch is supervised by depth maps generated by the 2DGS branch, while reflection-free images rendered by the 3DGS branch concurrently supervise the 2DGS branch, facilitating a mutual-boosted alternating optimization. Finally, following to [6], we employ Truncated Signed Distance Fusion (TSDF) [27] to extract the reconstructed textured meshes.

#### 3.2. Illumination decomposition with 3DGS

3DGS is constrained in modeling transparent or translucent materials, such as glass. To enhance the comprehensive modeling of scenes under various light conditions, the entire scene is modeled as composed of transmitted and reflected components. Specifically, we retain the original 3DGS as the transmitted component, while introducing three reflection-related parameters to represent the reflected component: reflected opacity  $\theta_{\text{ref}} \in \mathbb{R}$ , reflected confidence  $\beta \in [0, 1]$ , and reflected SH  $C_{\text{ref}} \in \mathbb{R}^k$ , where  $k$  refers to the degrees of freedom.

The reflected confidence  $\beta$  represents the probability that an individual Gaussian primitive captures the reflected component. When splatting 3D Gaussians to 2D images,  $\beta$  is accumulated as described in Equation (1) to obtain the pixel-wise reflected confidence  $W$ , and the transmitted color  $C_{\text{tran}}$  and reflected color  $C_{\text{ref}}$  are calculated with  $\alpha$ -blending from front to back, as depicted in:

$$W = \sum_{i=1}^K \beta^i \alpha_{\text{ref}}^i \prod_{j=1}^{k-1} (1 - \beta^j), \quad (1)$$

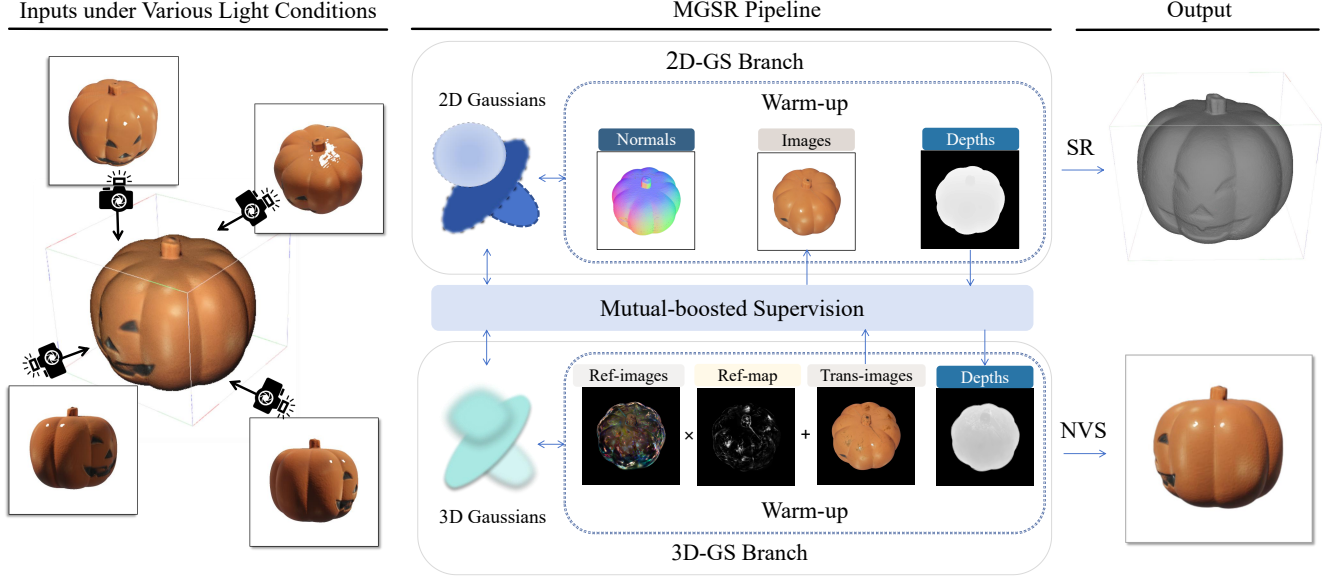


Figure 2. MGSR is a 2D/3D mutual-boosted framework with two branches: 2DGS branch (upper) for SR and 3DGS branch (bottom) for NVS. Each branch is enhanced by our specific designs. Upon receiving inputs from various light conditions, the two branches initially undergo a warm-up stage of initialization for mutual-boosted optimization. In the alternating optimization process under mutual-boosted supervision, the 3DGS branch is guided by depth maps generated by the 2DGS branch, while simultaneously, rendered transmitted images without reflections from the 3DGS branch provide reflection-removed supervision to the 2DGS branch.

$$C_{\text{tran}} = \sum_{i=1}^K c_{\text{tran}}^i \alpha_{\text{tran}}^i \prod_{j=1}^{i-1} (1 - \alpha_{\text{tran}}^j), \quad (2)$$

$$C_{\text{ref}} = \sum_{i=1}^K c_{\text{ref}}^i \alpha_{\text{ref}}^i \prod_{j=1}^{i-1} (1 - \alpha_{\text{ref}}^j), \quad (3)$$

where  $i$  denotes the index of the Gaussian sphere,  $\alpha_{\text{tran}}$  and  $\alpha_{\text{ref}}$  denote to the transmitted opacity  $\theta_{\text{tran}}$  and reflected opacity  $\theta_{\text{ref}}$  multiplied by the density of the splatted Gaussian at the pixel location,  $c_{\text{tran}}^i$  and  $c_{\text{ref}}^i$  represent the view-dependent transmitted and reflected color calculated from transmitted SH  $C_{\text{tran}}$  and reflected SH  $C_{\text{ref}}$ . The rendered color  $C$  is calculated according to Equation (4), combining the transmitted and reflected components, weighted by the reflected confidence  $W$ :

$$C = C_{\text{tran}} + W \times C_{\text{ref}}. \quad (4)$$

A total variation (TV) loss  $\mathcal{L}_{\text{trans-TV}}$  is utilized to smooth in local regions of the transmitted components. Subsequently, the rendering loss is applied to encourage rendered color  $C$  to be similar to the GT color  $C_{\text{GT}}$ .

$$\mathcal{L}_{\text{render}} = \lambda_1 \mathcal{L}_1(C, C_{\text{GT}}) + (1 - \lambda_1) \mathcal{L}_{\text{D-SSIM}}(C, C_{\text{GT}}), \quad (5)$$

where  $\lambda_1$  represents to the balance coefficient,  $\mathcal{L}_1$  computes the absolute error, while  $\mathcal{L}_{\text{D-SSIM}}$  refers to the differentiable structural similarity index measure (SSIM). The total loss is a weighted sum of the rendering loss and the TV loss of transmitted components,

$$\mathcal{L}_{3D} = \mathcal{L}_{\text{render}} + \lambda_2 \mathcal{L}_{\text{trans-TV}}, \quad (6)$$

where  $\lambda_2$  denotes to the weight.

However, due to the inherent limitation of 3DGS, specifically the inaccuracy of the depth map, although this 3D branch successfully performs illumination decomposition, it still cannot extract the surface mesh. Therefore, an additional 2DGS branch is introduced to provide reliable geometry supervision.

### 3.3. Surface reconstruction with 2DGS

2DGS represents the scene with flattened 2D Gaussian primitives, aligning the centers of the Gaussian disks with the surface. The aim of this section is to utilize these flattened Gaussian primitives to obtain initialized depth maps of scenes under various light conditions, which also serve as supervision for the 3DGS branch and are refined iteratively through the mutual-boosted stage (Section 3.4).

As an initial estimate, the rendering depth map  $Z$  is computed as a weighted sum of the normalized intersected depths  $z$ , as depicted in:

$$Z = \frac{\sum_i \omega_i z_i}{\sum_i \omega_i + \epsilon}, \quad (7)$$

where  $\omega_i = T_i \alpha_i$  is the contribution of 2D Gaussian disk to the rendering depth at the pixel location.  $\alpha_i$  is the opacity  $\theta$  multiplied by the density of the splatted Gaussian, and the visibility term  $T_i$  is calculated:

$$T_i = \prod_{j=1}^{i-1} (1 - \alpha_j), \quad (8)$$



where the definition of  $\alpha_j$  is same as above.

In the representation based on 2D Gaussian primitives, it is crucial to ensure that all 2D splats are locally aligned with the actual surfaces of the object. This alignment is particularly important in the context of volume rendering, where multiple semi-transparent surfels may be encountered along the ray. To accurately identify the actual surface, we follow [6] and consider the median point of intersection where the accumulated opacity reaches 0.5. The normals of the 2D splats are encouraged to be aligned with the gradients of the depth maps, as shown in:

$$\mathcal{L}_n = \sum_i \omega_i (1 - n_i^T N), \quad (9)$$

where  $i$  indexes the intersected splats along the ray,  $n_i$  represents the normal of the splat oriented towards the camera, and  $N$  is the normal estimated from the nearby depth point  $Z$ , and computed by:

$$N(x, y) = \frac{\nabla_x Z \times \nabla_y Z}{\|\nabla_x Z \times \nabla_y Z\|}. \quad (10)$$

Considering that our task is to reconstruct the surface meshes under various light conditions, a weighted normal loss is employed, which encourages the model to primarily learn the normals of the surfaces rather than those of the entire scenes. The normal loss from the camera's perspective will be determined based on the division of the mask into foreground and background, with a coefficient  $\gamma$  applied to weight the contributions. Two TV losses  $\mathcal{L}_{d-TV}$  and  $\mathcal{L}_{n-TV}$  are introduced as smooth terms on both rendered depths and normals. The overall loss of the 2DGS branch consists of a weighted combination:

$$\mathcal{L}_{2D} = \mathcal{L}_{render} + \lambda_3(\gamma\mathcal{L}_n + \lambda_4\mathcal{L}_{n-TV}) + \lambda_5\mathcal{L}_{d-TV}, \quad (11)$$

where  $\lambda_3$ ,  $\lambda_4$ , and  $\lambda_5$  represent the weights, and  $\mathcal{L}_{render}$  is defined in Equation (5).

Yet, the 2DGS branch has only accomplished modeling of illuminated scenes and relatively reliable depth estimation without eliminating the influence of light conditions, which is inconsistent with real unlit scenes. To address this issue, an alternating optimization approach is devised after the warm-up stage of both the 2DGS and 3DGS branches.

### 3.4. Alternating optimization of 2D & 3D Gaussians

Given that the warm-up 3DGS branch has performed a preliminary illumination decomposition in Section 3.2, and the warm-up 2DGS branch has provided an initial estimate of the depth maps for the scenes under various light conditions in Section 3.3, our alternating optimization approach utilizes the transmitted colors from the 3DGS branch to supervise the 2DGS branch for surface reconstruction, while the depth maps from the 2DGS branch provides geometry

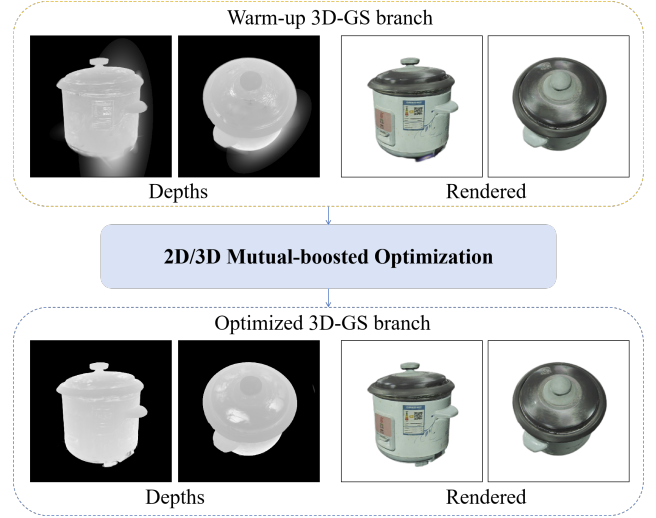


Figure 3. Geometry enhancement in 3DGS branch for realistic rendering through our mutual-boosted optimization.

information for the 3DGS branch for better illumination decomposition.

Specifically, since the illumination decomposition in the 3DGS branch is unsupervised, the transmitted components from 3DGS branch might not be completely disentangled from the reflected components after the warm-up stage, a weighted rendering loss is introduced for rendering supervision for the 2DGS branch. Our goal is for the images rendered by the 2DGS branch to not only approximate the transmitted images but also to retain a slight similarity to the illuminated images to enhance the stability of alternating optimization, as calculated by:

$$\mathcal{L}_{render-m} = \lambda_6\mathcal{L}_{mutual} + (1 - \lambda_6)\mathcal{L}_{2D-render}, \quad (12)$$

where  $\mathcal{L}_{render-m}$  represents the mutual rendering loss in alternating optimization, and  $\lambda_6$  is the weight.  $\mathcal{L}_{mutual}$  denotes the rendering loss between the 2DGS branch rendered images and the transmitted images from the 3DGS branch,  $\mathcal{L}_{2D-render}$  is the rendering loss between the 2DGS branch rendered images and the GT images, and both of the rendering loss are computed by Equation (5). Moreover, both TV losses on depths ( $\mathcal{L}_{d-TV-m}$ ) and normals ( $\mathcal{L}_{n-TV-m}$ ) in 2DGS branch are retained, but GT images are replaced with the transmitted images. In the alternating optimization stage, the loss function of 2DGS branch will be promoted to:

$$\mathcal{L}_{2D} = \mathcal{L}_{render-m} + \lambda_3(\gamma\mathcal{L}_n + \lambda_4\mathcal{L}_{n-TV-m}) + \lambda_5\mathcal{L}_{d-TV-m}, \quad (13)$$

where  $\gamma$  is the coefficient that balances the contribution of the foreground and background. The weights  $\lambda_2$ ,  $\lambda_3$ , and  $\lambda_4$  are the same as the warm-up stage.

Scene geometry is essential for realistic physically-based rendering, and geometry estimation on Gaussian primitives is difficult due to the discrete structures. Therefore, on the

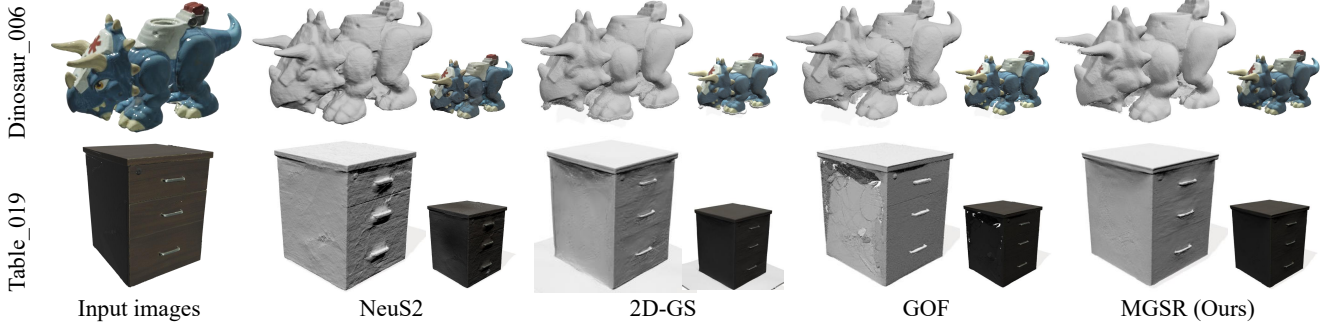


Figure 4. Visual comparisons on the OmniObject3D dataset [22].

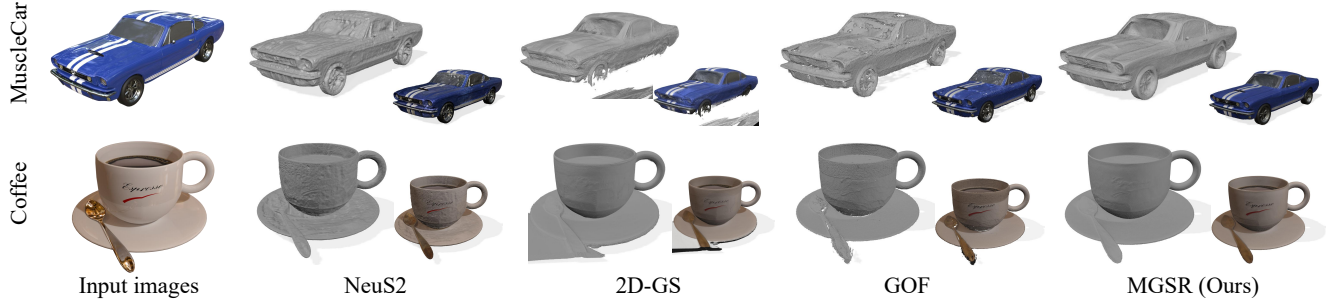


Figure 5. Visual comparisons on the Shiny Blender dataset [21].

other hand, a depth loss between 2DGS and 3DGS branches is introduced to improve the geometry estimation for the illumination decomposition of 3DGS branch (as shown in Figure 3). The calculation of depth maps  $Z_{3D}$  in 3DGS branch is the same as Equation (7). To encourage the alternating optimization to focus on the foreground part of the input images, masks are used to avoid the calculation of background by with a weight coefficient  $\gamma$ . The depth loss  $\mathcal{L}_Z$  is computed by:

$$\mathcal{L}_Z = \gamma \mathcal{L}_2(Z_{2D}, Z_{3D}), \quad (14)$$

where  $\mathcal{L}_2$  denotes the L2 loss, and  $Z_{2D}$  is the estimated depth map  $Z$  in the 2DGS branch.

The total loss  $\mathcal{L}_{total}$  of the alternating optimization is:

$$\mathcal{L}_{total} = w_{2D} \mathcal{L}_{2D} + w_{3D} \mathcal{L}_{3D} + w_{depth-mutual} \mathcal{L}_Z, \quad (15)$$

where the losses of the 3D module  $\mathcal{L}_{3D}$  and the 2D module  $\mathcal{L}_{2D}$  are calculated by Equation (6) and Equation (13), respectively, and  $w_{2D}$ ,  $w_{3D}$ , and  $w_{depth-mutual}$  are the weights. See Section 5 for their settings.

**Auto-stop strategy.** Moreover, due to the varying convergence speeds in each branch, we should not warm up both branches simultaneously at the same iterations. Therefore, to prevent the two branches from being overly optimized during the warm-up stage and to expedite the overall process, an auto-stop strategy is introduced in the warm-up process. Specifically, as one branch reaches convergence, it will initiate our alternating optimization process first. Subsequently, when the other branch also converges, our mutual-boosted stage starts.

Table 1. NVS results on Shiny Blender and OmniObject3D. The instance-level metrics are listed in Appendix.

Methods	Shiny Blender		OmniObject3D		
	SSIM $\uparrow$	PSNR $\uparrow$	SSIM $\uparrow$	PSNR $\uparrow$	Time $\downarrow$
3D-GS	0.9630	32.62	0.9859	33.64	<b>7min</b>
GS-IR	0.9409	32.09	0.9752	35.16	33min
GShader	<b>0.9679</b>	<u>34.35</u>	0.9840	34.14	105min
R3DG	<u>0.9646</u>	34.20	<b>0.9874</b>	<u>37.37</u>	21min
MGSR (Ours)	0.9645	<b>34.66</b>	0.9850	<b>37.69</b>	<u>13min</u>

## 4. Experiments

### 4.1. Datasets and evaluation metrics

**DTU** [7] is a large MVS dataset, where some scenes feature unfavorable light conditions for surface reconstruction, such as overexposure, underexposure, and metallic reflections. Similar to previous baselines [6, 24], we utilized the same 15 scans from the DTU dataset to validate our approach. **OmniObject3D** [22] contains objects with extensive areas of specular highlights on their surfaces. **Shiny Blender** [21] is introduced to assess the capability of surface mesh extraction under strong reflections. **Ref-NeRF Real Captured Scenes** [21] consists of three in-the-wild scenes with strong reflections. **Tanks and Temples (TnT)** [10] consists a large number of high-resolution real-world images under varying light conditions.

We utilize SSIM and PSNR to evaluate the rendering quality, while reconstruction accuracy is validated by 10K sampled points with Normal Consistency (NC) and Chamfer Distance (CD) measurements. Due to the limitation of CD, we mainly focus on NC metric, which aligns better

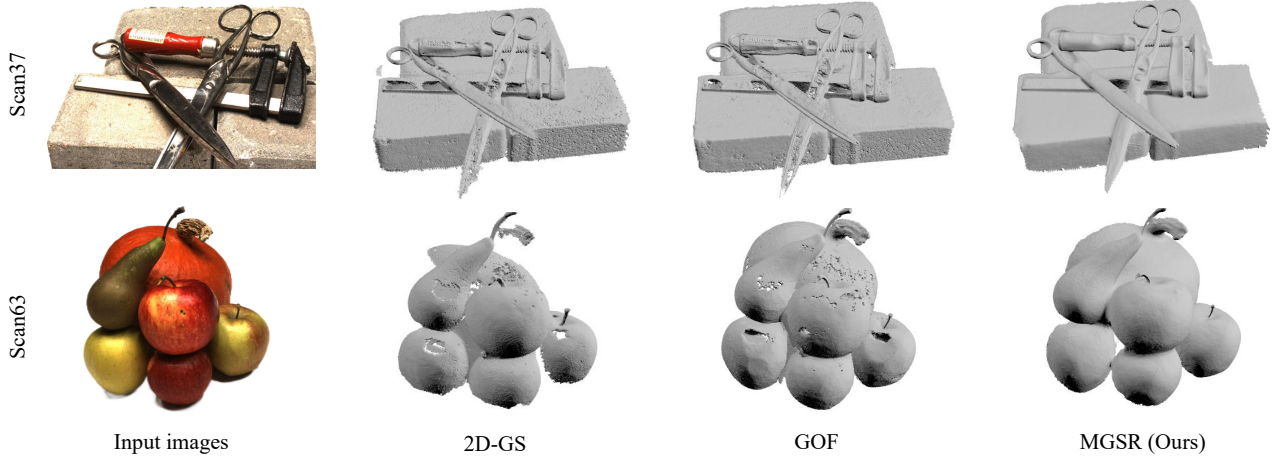


Figure 6. Visual comparisons on DTU dataset [7].

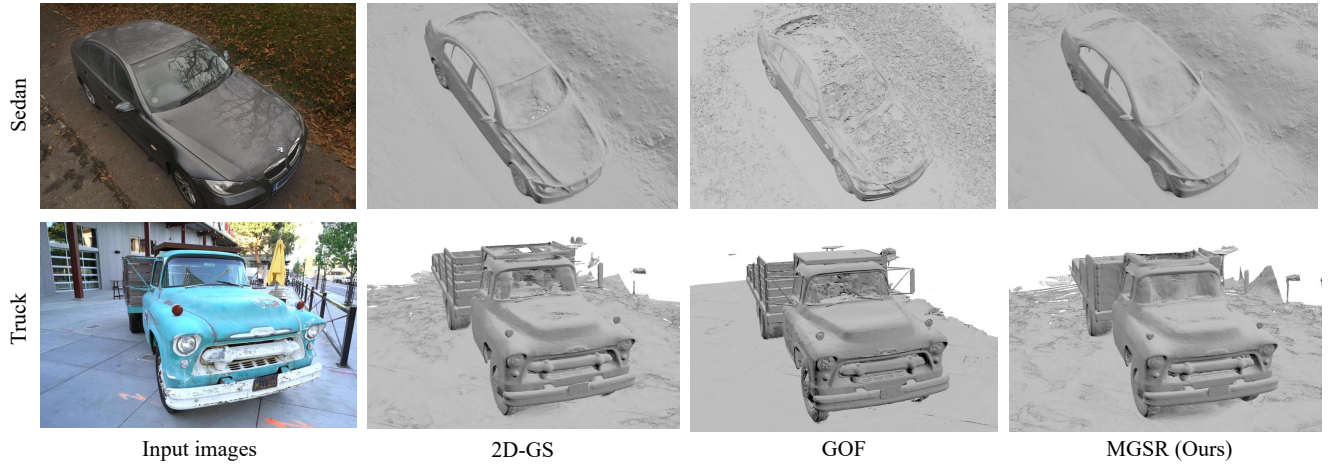


Figure 7. Visual comparisons on Ref-NeRF Real Captured Scenes dataset (Sedan) [21] and TnT dataset (Truck) [7].

Table 2. SR results on Shiny Blender and OmniObject3D. NC is multiplied by  $10^2$ , and CD is multiplied by  $10^3$ . Methods marked with \* fail on certain objects, which are excluded from the average metric values presented in this table, but are detailed in the list of instance-level metrics in the Appendix.

Methods	Shiny Blender				OmniObject3D				Time $\downarrow$
	NC $\uparrow$	CD $\downarrow$	SSIM $\uparrow$	PSNR $\uparrow$	NC $\uparrow$	CD $\downarrow$	SSIM $\uparrow$	PSNR $\uparrow$	
NeuS2	62.58	<b>0.51</b>	-	-	88.21	<b>0.48</b>	-	-	<b>5min</b>
2D-GS*	65.48	41.54	0.9597	33.46	86.34	0.75	0.9836	34.09	12min
GOF*	44.54	<u>2.11</u>	<b>0.9667</b>	<b>34.66</b>	<u>90.06</u>	0.84	<b>0.9885</b>	<b>38.18</b>	16min
MGSR (Ours)	<b>66.47</b>	2.85	0.9645	<b>34.66</b>	<b>90.60</b>	0.90	0.9850	<u>37.69</u>	13min

with human perception. Since GT mesh is unavailable for real-world data, only visual comparisons are provided.

## 4.2. Results

**Object-level data.** Three objects from the Shiny Blender with reflections and 30 objects from OmniObject3D with highlights are conducted on all methods for comparison. Table 1 and Table 2 present quantitative results on both NVS and SR. MGSR surpasses all baselines in terms of the PSNR metric for NVS and the NC metric for SR. As shown in visualization comparisons of Figure 4 and Figure 5, although NeuS2 exhibits superior CDs, its smoothness on surfaces with reflections is significantly poor. To address this issue,

NC is introduced as an evaluation metric for reconstruction, overcoming the limitations of CDs, which fail to capture surface holes or bumps. 2DGS produces excessive faces on reconstructed meshes on the exterior of objects while failing to accurately reconstruct thin surfaces. GOF tends to reconstruct rough or incomplete surfaces when handling lit surfaces. MGSR visually outperforms all baselines, resulting in the best NC, with smooth surfaces and accurate color modeling. Table 3 and Figure 6 show the quantitative and visual results compared to GS-based SR methods on DTU dataset. Under varying light conditions, MGSR successfully reconstructs realistic and intact surfaces compared to 2DGS and GOF.

**Scene-level data.** As shown in Figure 7, two in-the-wild scenes from Ref-NeRF [21] and TnT [10] are used for visual comparisons. More comparisons are shown in the Appendix. Previous GS-based methods fail to effectively reconstruct glass or mirror surfaces, resulting in damaged and inaccurate surfaces. In contrast, MGSR successfully reconstructs meshes in mirror material, driven by our devised 3DGS branch and mutual-boosted optimization, which perform illumination decomposition and guide the



Table 3. Instance-level Normal Consistency results of SR on DTU dataset. Results are multiplied by  $10^3$ .

Methods	24	37	40	55	63	65	69	83	97	105	106	110	114	118	122	Mean
2D-GS	50.58	<b>49.89</b>	45.73	46.09	<b>68.10</b>	<b>54.62</b>	54.36	54.72	60.33	33.12	<b>49.59</b>	71.81	60.83	44.95	47.00	52.78
GOF	38.13	47.76	45.47	41.17	61.47	49.84	52.35	<b>55.00</b>	55.85	42.75	48.70	73.11	55.65	52.01	<b>47.05</b>	51.09
MGSR (Ours)	<b>52.46</b>	48.59	<b>47.47</b>	<b>51.66</b>	55.17	45.34	<b>63.27</b>	40.71	<b>61.51</b>	<b>52.40</b>	42.59	<b>74.08</b>	<b>65.04</b>	<b>52.16</b>	43.81	<b>53.08</b>

Table 4. Ablations of loss weights (Models A-F), iterations of mutual-boosted optimization (Models G-J), bidirectional BP and auto-stop warm-up strategy (Models K-L) on OmniObject3D dataset. NC is multiplied by  $10^2$ , and CD is multiplied by  $10^3$ . The best-performing model in each ablation study is highlighted.

Model	Mutual-boosted Iterations	Bidirectional BP	Auto-stop Warm-up	$w_{2D}$	$w_{3D}$	$w_{depth-mutual}$	SSIM $\uparrow$	PSNR $\uparrow$	NC $\uparrow$	CD $\downarrow$
A	20k	No	Yes	0	1	0	0.9755	35.10	-	-
B	20k	No	Yes	1	0	0	-	-	90.16	0.92
C	20k	No	Yes	0.3	0.7	0.01	<b>0.9861</b>	<b>37.88</b>	90.54	0.92
D	20k	No	Yes	0.7	0.3	0.01	0.9825	36.99	<b>90.62</b>	0.92
E	20k	No	Yes	0.5	0.5	0.01	<u>0.9850</u>	<u>37.69</u>	<u>90.60</u>	<b>0.90</b>
F	20k	No	Yes	0.5	0.5	0.1	0.9803	36.42	<u>90.60</u>	0.91
G	5k	No	Yes	0.5	0.5	0.01	0.9820	36.76	90.40	<b>0.89</b>
H	10k	No	Yes	0.5	0.5	0.01	0.9842	37.37	90.50	0.91
I	20k	No	Yes	0.5	0.5	0.01	<b>0.9850</b>	<u>37.69</u>	<b>90.60</b>	<u>0.90</u>
J	30k	No	Yes	0.5	0.5	0.01	<b>0.9850</b>	<b>37.71</b>	<b>90.60</b>	0.91
K	20k	Yes	Yes	0.5	0.5	0.01	0.9813	36.64	90.57	0.91
L	20k	No	No	0.5	0.5	0.01	0.9831	37.01	90.56	0.92
M	20k	No	Yes	0.5	0.5	0.01	<b>0.9850</b>	<b>37.69</b>	<b>90.60</b>	<b>0.90</b>

2DGS branch for enhanced SR.

**Optimization time.** In Table 1 and Table 2, we additionally present the optimization time for all compared methods. By eliminating the dependence on BRDF, MGSR is faster than all illumination decomposition baselines. Among methods supporting both NVS and SR, 2DGS is the fastest. MGSR, supported by an auto-stop warm-up strategy, outperforms GOF and achieves a comparable speed with 2DGS. Specifically, each of the two branches in MGSR has an average warm-up optimization time of around 3.5-4 minutes. The mutual-boosted optimization time is approximately 8-9 minutes.

## 5. Ablation studies

All ablation studies (Table 4) are conducted on the OmniObject3D dataset due to its various light conditions, with the experimental setup consistent with Section 4. Additional results are provided in the Appendix.

**Auto-stop warm-up.** Experimental results (Models G-J) show that the auto-stop strategy during warm-up outperforms both MVS and SR, demonstrating the effectiveness of the auto-stop strategy in preventing overfitting during the warm-up stage. **Bidirectional back-propagation (BP).** In alternating optimization (Section 3.4), the outputs from one branch are utilized to supervise the other branch. During the supervision, the gradient of loss can be back-propagated to one branch (unidirectional BP) or both branches (bidirectional BP). Experimental results (Models K and M) show that unidirectional BP outperforms bidirectional BP in terms of both NVS and SR results. It is mainly due to the bidirectional BP influences optimization

of the branch which provides supervision. **Mutual-boosted iterations.** We observed that when the mutual-boosted iterations exceed 20k, there is no significant improvement in the NVS and SR metrics (Models L and M). Therefore, 20k iterations are optimal for MGSR. **Loss weights.** The cases without mutual-boosted optimization are firstly validated, where only the 3DGS branch is used for NVS or the 2DGS branch for SR (Models A and B). The results were inferior compared to the case with mutual-boosted optimization. Furthermore, it is found that the branch with a higher weight (Models C and D) during the mutual-boosted optimization stage tends to favor the task it excels at, leading to better performance. To balance both NVS and SR, the weights of the two branches should be equal.

## 6. Conclusion

We introduce MGSR, a 2D/3D Mutual-boosted Gaussian splatting for SR that enhances both rendering quality and 3D reconstruction accuracy under various light conditions. Moreover, the auto-stop strategy is proposed in the warm-up stage, while geometry-guided illumination decomposition is devised in the mutual-boosted optimization stage. Extensive experiments across various synthetic and real-world datasets at both object and scene levels have validated the superiority of MGSR.

**Limitations and future work.** Although MGSR achieves reliable reconstruction under reflective conditions, over-smoothness may occur in low-light scenarios, leading to the absence of details. A possible way for addressing this issue is to incorporate exposure compensation for input images, which we will investigate as a future work.



## Acknowledgment

This project was partially supported by the Ministry of Education, Singapore, under its Academic Research Fund Grant (RT19/22). The computational experiments were conducted using the CFFF platform at Fudan University.

## References

- [1] Danpeng Chen, Hai Li, Weicai Ye, Yifan Wang, Weijian Xie, Shangjin Zhai, Nan Wang, Haomin Liu, Hujun Bao, and Guofeng Zhang. PGSR: Planar-based Gaussian splatting for efficient and high-fidelity surface reconstruction. *IEEE Transactions on Visualization and Computer Graphics*, 2024. 3
- [2] Zilong Chen, Feng Wang, Yikai Wang, and Huaping Liu. Text-to-3D using Gaussian splatting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 21401–21412, 2024. 1
- [3] Ben Fei, Jingyi Xu, Rui Zhang, Qingyuan Zhou, Weidong Yang, and Ying He. 3D Gaussian splatting as new era: A survey. *IEEE Transactions on Visualization and Computer Graphics*, 31(8):4429–4449, 2025. 1
- [4] Jian Gao, Chun Gu, Youtian Lin, Hao Zhu, Xun Cao, Li Zhang, and Yao Yao. Relightable 3D Gaussian: Real-time point cloud relighting with BRDF decomposition and ray tracing. *arXiv preprint arXiv:2311.16043*, 2023. 1, 3
- [5] Antoine Guédon and Vincent Lepetit. Sugar: Surface-aligned Gaussian splatting for efficient 3D mesh reconstruction and high-quality mesh rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5354–5363, 2024. 1, 3
- [6] 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*, pages 1–11, 2024. 1, 2, 3, 5, 6
- [7] Rasmus Jensen, Anders Dahl, George Vogiatzis, Engin Tola, and Henrik Aanæs. Large scale multi-view stereopsis evaluation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 406–413, 2014. 6, 7
- [8] Yingwenqi Jiang, Jiadong Tu, Yuan Liu, Xifeng Gao, Xiaoxiao Long, Wenping Wang, and Yuexin Ma. Gaussian-Shader: 3D Gaussian splatting with shading functions for reflective surfaces. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5322–5332, 2024. 1, 3
- [9] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3D Gaussian splatting for real-time radiance field rendering. *ACM Trans. Graph.*, 42(4):139–1, 2023. 1
- [10] Arno Knapitsch, Jaesik Park, Qian-Yi Zhou, and Vladlen Koltun. Tanks and temples: Benchmarking large-scale scene reconstruction. *ACM Transactions on Graphics*, 36(4):1–13, 2017. 6, 7
- [11] Jiaze Li, Zhengyu Wen, Luo Zhang, Jiangbei Hu, Fei Hou, Zhebin Zhang, and Ying He. GS-Octree: Octree-based 3D Gaussian splatting for robust object-level 3d reconstruction under strong lighting. *Computer Graphics Forum*, 43(7): e15206, 2024. 1
- [12] 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*, pages 21644–21653, 2024. 3
- [13] Kunhao Liu, Fangneng Zhan, Muyu Xu, Christian Theobalt, Ling Shao, and Shijian Lu. StyleGaussian: Instant 3D style transfer with Gaussian splatting. *arXiv preprint arXiv:2403.07807*, 2024. 1
- [14] Weijie Lyu, Xueting Li, Abhijit Kundu, Yi-Hsuan Tsai, and Ming-Hsuan Yang. Gaga: Group any Gaussians via 3D-aware memory bank. *arXiv preprint arXiv:2404.07977*, 2024. 1
- [15] Hidenobu Matsuki, Riku Murai, Paul HJ Kelly, and Andrew J Davison. Gaussian splatting slam. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18039–18048, 2024. 1
- [16] Marko Mihajlovic, Sergey Prokudin, Siyu Tang, Robert Maier, Federica Bogo, Tony Tung, and Edmond Boyer. Splatfields: Neural Gaussian splats for sparse 3d and 4d reconstruction. *arXiv preprint arXiv:2409.11211*, 2024. 1
- [17] Haokai Pang, Heming Zhu, Adam Kortylewski, Christian Theobalt, and Marc Habermann. Ash: Animatable Gaussian splats for efficient and photoreal human rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1165–1175, 2024. 1
- [18] Zhiyin Qian, Shaofei Wang, Marko Mihajlovic, Andreas Geiger, and Siyu Tang. 3DGS-avatar: Animatable avatars via deformable 3D Gaussian splatting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5020–5030, 2024. 1
- [19] Minghan Qin, Wanhua Li, Jiawei Zhou, Haoqian Wang, and Hanspeter Pfister. Langsplat: 3D language Gaussian splatting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20051–20060, 2024. 1
- [20] Sara Sabour, Lily Goli, George Kopanas, Mark Matthews, Dmitry Lagun, Leonidas Guibas, Alec Jacobson, David J Fleet, and Andrea Tagliasacchi. Spotlessplats: Ignoring distractors in 3D Gaussian splatting. *arXiv preprint arXiv:2406.20055*, 2024. 1
- [21] Dor Verbin, Peter Hedman, Ben Mildenhall, Todd Zickler, Jonathan T Barron, and Pratul P Srinivasan. Ref-NeRF: Structured view-dependent appearance for neural radiance fields. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5481–5490. IEEE, 2022. 3, 6, 7
- [22] Tong Wu, Jiarui Zhang, Xiao Fu, Yuxin Wang, Jiawei Ren, Liang Pan, Wayne Wu, Lei Yang, Jiaqi Wang, Chen Qian, Dahua Lin, and Ziwei Liu. Omniobject3D: Large-vocabulary 3D object dataset for realistic perception, reconstruction and generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 803–814, 2023. 6
- [23] Yao Yao, Jingyang Zhang, Jingbo Liu, Yihang Qu, Tian Fang, David McKinnon, Yanghai Tsin, and Long Quan.

- NEILF: Neural incident light field for physically-based material estimation. In *European Conference on Computer Vision*, pages 700–716. Springer, 2022. [2](#), [3](#)
- [24] Zehao Yu, Torsten Sattler, and Andreas Geiger. Gaussian opacity fields: Efficient and compact surface reconstruction in unbounded scenes. *arXiv preprint arXiv:2404.10772*, 2024. [1](#), [2](#), [3](#), [6](#)
- [25] Jingyang Zhang, Yao Yao, Shiwei Li, Jingbo Liu, Tian Fang, David McKinnon, Yanghai Tsin, and Long Quan. NEILF++: Inter-reflectable light fields for geometry and material estimation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3601–3610, 2023. [2](#)
- [26] Xiuming Zhang, Pratul P Srinivasan, Boyang Deng, Paul Debevec, William T Freeman, and Jonathan T Barron. NeRFactor: Neural factorization of shape and reflectance under an unknown illumination. *ACM Transactions on Graphics*, 40(6):1–18, 2021. [2](#)
- [27] Qian-Yi Zhou, Jaesik Park, and Vladlen Koltun. Open3D: A modern library for 3D data processing. *arXiv preprint arXiv:1801.09847*, 2018. [3](#)
- [28] Xiaoyu Zhou, Zhiwei Lin, Xiaojun Shan, Yongtao Wang, Deqing Sun, and Ming-Hsuan Yang. DrivingGaussian: Composite Gaussian splatting for surrounding dynamic autonomous driving scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 21634–21643, 2024. [1](#)
- [29] Xiaoyu Zhou, Xingjian Ran, Yajiao Xiong, Jinlin He, Zhiwei Lin, Yongtao Wang, Deqing Sun, and Ming-Hsuan Yang. Gala3D: Towards text-to-3D complex scene generation via layout-guided generative Gaussian splatting. *arXiv preprint arXiv:2402.07207*, 2024. [1](#)