TextureSplat: Per-Primitive Texture Mapping for Reflective Gaussian Splatting

Mae Younes Adnane Boukhayma
Inria France

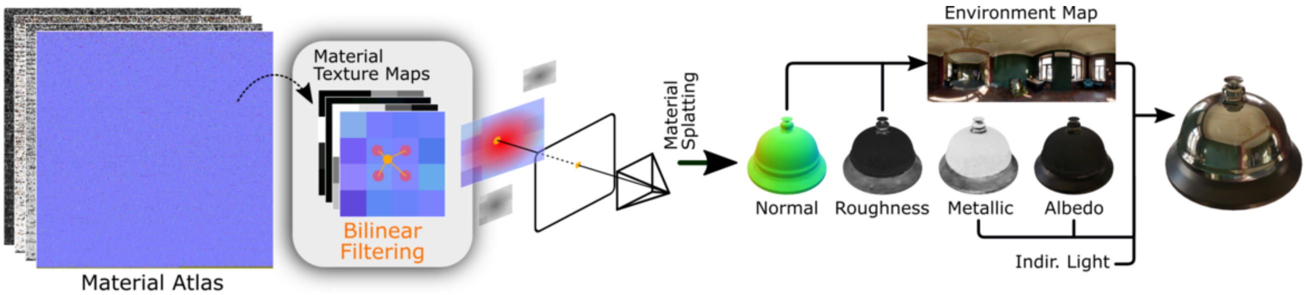

Figure 1. *Method Overview:* We introduce planar primitive material textures — as opposed to single attributes — within physically based Gaussian Splatting rendering optimization. The increased representation power from spatially varying normal and material in object space enables fidelity reconstruction of high frequency specular in highly reflective scenes. Our hardware-accelerated implementation using texture atlases improves rendering efficiency at test time.

Abstract

Gaussian Splatting have demonstrated remarkable novel view synthesis performance at high rendering frame rates. Optimization-based inverse rendering within complex capture scenarios remains however a challenging problem. A particular case is modelling complex surface light interactions for highly reflective scenes, which results in intricate high frequency specular radiance components. We hypothesize that such challenging settings can benefit from increased representation power. We hence propose a method that tackles this issue through a geometrically and physically grounded Gaussian Splatting borne radiance field, where normals and material properties are spatially variable in the primitive's local space. Using per-primitive texture maps for this purpose, we also propose to harness the GPU hardware to accelerate rendering at test time via unified material texture atlas. Code will be available at `TextureSplat.`

1. Introduction

3D reconstruction and inverse rendering from multi-view images are pivotal problems receiving constant interest and investigation from computer vision, graphics and machine learning research communities, with a myriad of direct applications in key industrial domains requiring high-quality 3D modeling and visualization.

Neural Radiance Fields (NeRF) [42] revolutionized the field by representing scenes as continuous neural implicit functions optimized through differentiable volume rendering. Building on this foundation, 3D Gaussian Splatting (3DGS) [27] reintroduced point-based graphics by replacing neural networks with explicit 3D Gaussian primitives, achieving both real-time rendering and state-of-the-art quality. These primitives are rendered through volume resampling [87], with their parameters optimized via gradient descent-based differentiable rendering. More recently, 2D Gaussian Splatting (2DGS) [18] improved multi-view consistency by using planar Gaussian primitives that better align with surfaces.

Despite these advances, accurately representing highly reflective surfaces remains challenging. Reflective objects exhibit complex view-dependent effects that depend on surface normals, material properties, and environmental lighting. Recent methods such as RefGaussian [73] and 3DGS-DR [76] attempt to model these effects using per-Gaussian material properties and physically-based rendering approaches. However, they often struggle with high-frequency specular highlights and sharp reflections. We hypothesize that this is due in part to the inherent resolution limitations of using a single attribute value per primitive.

In this paper, we ask the question: *How can we enhance the representation power of Gaussian splatting for reflective scenes while maintaining computational efficiency and leveraging hardware acceleration?* Our key insight is that the planar nature of 2D Gaussian primitives naturally defines a parameterization that can be exploited for texture mapping, enabling us to store spatially varying material properties per primitive.

Inspired by the distinction between Gouraud shading [15] (constant per-vertex attributes) and Phong shading [53] (interpolated attributes) in traditional computer graphics, we introduce per-primitive texture maps for material properties in 2D Gaussian splatting. This approach effectively decouples the geometric representation (Gaussian primitives) from the appearance representation (material textures), allowing us to model high-frequency material variations without increasing the number of primitives.

Our method leverages the closed-form ray-splat intersection of 2DGS to accurately map screen-space pixels to local texture coordinates, enabling proper texture filtering. Crucially, we transform tangential normal maps to world space using the primitive's rotation matrix, analogous to normal mapping in traditional rendering. This enables detailed normal variations across each primitive's surface, significantly enhancing the rendering of specular highlights and reflections.

For efficient rendering after optimization, we pack the primitive textures into atlases that leverage GPU hardware-accelerated texture filtering operations. Our approach is fully compatible with deferred shading pipelines, allowing us to incorporate physically-based rendering models for accurate light interactions.

Through experiments on standard benchmarks for reflective scene reconstruction, we demonstrate that our method outperforms state-of-the-art approaches in terms of both quantitative metrics and visual quality. Our method achieves more accurate reflections and sharper specular highlights while maintaining real-time rendering performance. The benefits extend beyond reflective scenes, as our approach improves rendering quality for standard scenes as well.

Our contributions include:

- A per-primitive texture mapping approach for 2D Gaussian splatting that enhances representation power while maintaining computational efficiency.
- Leveraging a normal mapping technique in the context of Gaussian Splatting that significantly improves the quality of specular reflections.
- A hardware-accelerated implementation using texture atlases that enhances real-time rendering at test time.
- State-of-the-art results on benchmarks for reflective scene reconstruction, demonstrating significant improvements in rendering quality and accuracy.

2. Related Work

2.1. Radiance Fields for 3D Scene Representations

Neural Radiance Fields [42] (NeRFs) have been dominating the 3D shape and appearance modelling recently, based on the astounding success of implicit representations combined with differentiable volume rendering [31, 41]. They represent scenes using view-dependent radiance and density fields parameterized by MLPs. When density is modeled as a function of a signed distance field, NeRF variants enable more accurate geometry reconstruction [21, 34, 64, 70, 74, 78]. However, multi-scale volume rendering demands frequent MLP evaluations, limiting real-time performance. Grid-based methods [7, 10, 11, 43, 58, 59] alleviate this but often struggle with large, unbounded scenes even with level-of-detail grids [38]. Implicit reconstruction has been made more robust to noise and sparse observations, whether from images or point clouds, through the use of generalizable data priors (*e.g.* [6, 19, 26, 32, 46, 47, 49, 52, 79]) and a variety of regularization strategies (*e.g.* [2, 8, 16, 20, 33, 45, 48–51, 72]). Gaussian splatting (3DGS) [27] emerged lately as a strong alternative to NeRFs, offering state-of-the-art novel view synthesis and real time rendering frame rates. It extends the elliptical weighted average (EWA) volume resampling framework [86, 87] to inverse rendering, modelling scenes with explicit Gaussian kernel primitives, that can be sorted and rasterized efficiently. Recent extensions of Gaussian splatting include building generalizable models [5, 23, 39, 61], bundle-adjustment–based formulations [12, 22, 75, 84], using higher-dimensional primitives [9], spatiotemporal models [69], in addition to several methods to improve density control [28, 65, 82], anti-alising [37, 77, 80], model compactness [30, 66] and training speed [17, 29, 83]. The 2DGS representation [18] leverages planar 2D primitives instead of volumetric ones (3DGS), and performs precise 2D kernel evaluation in object space as opposed to approximative ones in screen space (3DGS), thus leading to superior geometric modelling and multi-view consistency.

2.2. Specular Reflection Modeling for Reflective Scenes

Both vanilla NeRFs and 3DGS assume low-frequency view dependency. Hence, they can struggle with highly reflective scenes. One strategy to improve in this department is using shading functions that are reflection direction aware [14, 25, 40, 57, 63]. For instance, Ref-NeRF [63] extends NeRFs with a new parameterization for view-dependent radiance and incorporates normal vector regularization. The shading function can be more physically grounded, and this enables additional application such as relighting and material editing. In this regard, other NeRF and GS based methods proposed to model light interaction using the explicit rendering equation with BRDF functions. ENVIDR [35] Uses environment maps to capture spatially-varying reflections in neural rendering. The next wave of work tackled shading for Gaussian Splatting (*e.g.* [13, 24, 36, 62, 73, 76, 85]). GShader [24] applies a simplified shading function on each fragment. 3DGS-DR [76] introduces deferred shading at pixel level, and stabilizes the optimization by smoothing out normal gradients.

Building on the 2DGS representation, our baseline Ref-Gaussian [73] decomposes the scene into geometry, material and lighting through the split-sum approximation of the rendering equation while incorporating an indirect light attribute, enabling inter-reflections while being fast to render. It achieves the state-of-the-art performance on the standard reflective scene novel view synthesis benchmarks. We propose to enhance its physical material and normal representations through the use of per-primitive textures.

2.3. Texture Attributes in Gaussian Splatting

Several works [4, 55, 56, 60, 68, 71] recently introduced the texture attribute representation for Gaussian Splatting as well. They use it to model a view independent component of the color. Differently, we explore this representation for rendering material properties and normals within 2DGS enabled physically based rendering to reconstruct challenging highly reflective scenes. We also propose leveraging texture atlases to enable hardware acceleration at test-time rendering. Closest to our context, concurrent work [1] manages to encode spatially varying primitive material attributes in a single compact texture map thanks to their pre-fitted template mesh.

3. Method

In this section, we present our approach to enhance 2D Gaussian Splatting (2DGS) [18] for representing highly reflective 3D scenes. We first provide background on 2DGS, then introduce our per-primitive texture mapping method that enables high-frequency detail on flat Gaussian primitives. Finally, we describe our hardware-accelerated implementation using texture atlases and the physically-based rendering model we use for reflective scenes.

3.1. Background: 2D Gaussian Splatting

We build upon 2D Gaussian Splatting (2DGS) [18], which represents a scene using oriented planar disks offering improved multi-view consistency compared to 3D Gaussian Splatting [27]. Each primitive k is characterized by its position $\boldsymbol{p}_k$, tangential vectors $\boldsymbol{t}_{u_k}$ and $\boldsymbol{t}_{v_k}$, and scaling factors (s_{u_k}, s_{v_k}). The primitive's normal is defined by the cross product $\boldsymbol{n}_k = \boldsymbol{t}_{u_k} \times \boldsymbol{t}_{v_k}$.

The 2D Gaussian is defined in a local tangent plane in world space with coordinates (u, v), parameterized as:

$$P_k(u, v) = \boldsymbol{p}_k + s_{u_k}\boldsymbol{t}_{u_k}u + s_{v_k}\boldsymbol{t}_{v_k}v. \tag{1}$$

For a point (u, v) in the local coordinate space corresponding to a given pixel (x, y) (*i.e.* ray-splat intersection), the 2D Gaussian value is evaluated as:

$$\mathcal{G}_k(u, v) = \exp\left(-\frac{u^2 + v^2}{2}\right). \tag{2}$$

Each splat has also a learnable opacity o_k. The final pixel color is computed by alpha-blending all primitives that contribute to the pixel in front-to-back order:

$$\boldsymbol{A}(x, y) = \sum_{i=1}^{N} \boldsymbol{a}_i\alpha_i(x, y)\prod_{j=1}^{i-1}(1 - \alpha_j(x, y)) \tag{3}$$

where $\alpha_i(x, y) = o_i\mathcal{G}_i(u_i(x, y), v_i(x, y))$ is the effective opacity of the i-th primitive at pixel (x, y) and $\boldsymbol{a}_i$ is its attribute (e.g., color), typically represented using spherical harmonics for view-dependent effects.

3.2. Per-Primitive Texture Mapping

While existing Gaussian splatting methods assign a single attribute value per primitive, we observe that the flat nature of 2D Gaussians naturally defines a local parameterization that can be leveraged for texture mapping. This enables encoding higher-frequency spatial detail without increasing the number of primitives, analogous to the distinction between Gouraud shading (constant per-primitive attributes) and Phong shading (interpolated attributes) in traditional rendering.

Rather than representing material properties with a single value per primitive, we define them as texture maps in the splat's local coordinate system:

$$\boldsymbol{a}_k(x, y) = \boldsymbol{\mathcal{T}}_k(u_k(x, y), v_k(x, y)) \tag{4}$$

where $\boldsymbol{\mathcal{T}}_k$ is the texture map associated with the k-th primitive, and $\boldsymbol{a}_k$ represents any reconstructed attribute.

This approach provides several advantages:

Decoupling of geometry and appearance: By separating the geometric representation (Gaussian primitives) from the appearance details (textures), we can represent complex visual features without increasing the number of primitives.

Higher fidelity appearance: Textures can capture high-frequency detail that would otherwise require many more primitives to represent.

Normal mapping: Instead of using a single normal per primitive, we can store detailed normal maps, significantly improving the rendering of specular effects.

The 2DGS representation is particularly well-suited for texture mapping because the ray-splat intersection already provides exact (u, v) coordinates in the primitive's local space and thus, enables accurate texture filtering.

3.3. Texture Mapping Implementation

To implement our texture mapping approach, we map the local splat coordinates (u, v) to texture coordinates (s, t) that account for the Gaussian kernel's support. Since the Gaussian kernel effectively drops to zero at approximately $S_\sigma = 3$ standard deviations, we scale the local coordinates to ensure that the effective support of the Gaussian ($[-S_\sigma, S_\sigma] \times [-S_\sigma, S_\sigma]$) maps to the texture space $[0, 1] \times [0, 1]$:

$$s = \frac{u + S_\sigma}{2S_\sigma}, \quad t = \frac{v + S_\sigma}{2S_\sigma}. \tag{5}$$

The attribute value $\boldsymbol{a}_k(x,y)$ for primitive k at pixel (x,y) is then obtained by bilinear filtering from its texture map $\boldsymbol{\mathcal{T}}_k$:

$$\boldsymbol{a}_k(x,y) = \text{BilinearFilter}(\boldsymbol{\mathcal{T}}_k, s(u_k(x,y)), t(v_k(x,y))). \tag{6}$$

3.4. Hardware Acceleration via Texture Atlases

To efficiently render optimized scenes at test time with per-primitive textures, we leverage hardware-accelerated texture filtering by packing individual primitive textures into texture atlases. This approach is inspired by methods like Ptex [3] and seamless texture atlases [54], adapted to the specific needs of Gaussian splatting. We provide details about Texture Atlas construction and texture sampling in the Supplementary Material (Section 1).

3.5. Physically-Based Deferred Rendering

Following the deferred rendering approach used in 3DGS-DR [76] and Ref-Gaussian [73], we first splat material attributes to screen-space buffers, then apply physically-based shading in a separate pass.

3.5.1 Material Properties

Each 2D Gaussian is associated with texture maps for the following material properties: Albedo $\boldsymbol{\lambda} \in [0,1]^3$, Metallic $m \in [0,1]$, Roughness $r \in [0,1]$, Tangent normal $\boldsymbol{n}^t \in [0,1]^3$ representing normal perturbations in the tangent space. For memory efficiency, we encode tangent normals using only two components (n_x^t, n_y^t) and reconstruct the third component at runtime using $n_z^t = \sqrt{\max(0, 1-(n_x^t)^2-(n_y^t)^2)}$.

3.5.2 Normal Mapping

A key factor in our approach is the use of normal mapping instead of a single normal per primitive. The tangent normal map encodes normal perturbations in the primitive's local coordinate system. These are transformed to world space using:

$$\boldsymbol{n}_k(x,y) = \boldsymbol{R}_k \cdot \boldsymbol{n}_k^t(x,y) \tag{7}$$

where $\boldsymbol{R}_k$ is the primitive's rotation matrix. This enables detailed normal variations across the surface of each primitive, critical for capturing high-frequency specular effects.

3.5.3 Attribute Splatting

We splat the material attributes to screen-space buffers using alpha-blending:

$$\boldsymbol{X}(x,y) = \sum_{i=1}^{N} \boldsymbol{x}_i(x,y)\alpha_i(x,y)\prod_{j=1}^{i-1}(1-\alpha_j(x,y)), \tag{8}$$

where $\boldsymbol{X}$ represents the combined screen-space buffers:

$$\boldsymbol{X} = \begin{bmatrix} \boldsymbol{\Lambda} \\ M \\ R \\ \boldsymbol{N} \\ \boldsymbol{L}_{\text{ind}} \end{bmatrix}, \quad \boldsymbol{x}_i = \begin{bmatrix} \boldsymbol{\lambda}_i(x,y) \\ m_i(x,y) \\ r_i(x,y) \\ \boldsymbol{n}_i(x,y) \\ \boldsymbol{l}_i^{\text{ind}}(x,y). \end{bmatrix} \tag{9}$$

This deferred approach treats alpha-blending as a smoothing filter, stabilizing the optimization of features sampled from textures and producing more cohesive rendering results compared to shading directly on the Gaussians [73, 76].

3.5.4 Physically-Based Shading

With the aggregated material maps, we apply the rendering equation to compute the outgoing radiance $\boldsymbol{L}_o(x,y,\omega_o)$ in the direction ω_o:

$$\boldsymbol{L}_o(x,y,\omega_o) = \boldsymbol{L}_d(x,y,\omega_o) + \boldsymbol{L}_s(x,y,\omega_o). \tag{10}$$

The diffuse term writes:

$$\boldsymbol{L}_d(x,y,\omega_o) = \frac{\boldsymbol{\Lambda}(x,y)}{\pi}(1-M(x,y))\mathcal{L}_{\text{env}}^{\text{diffuse}}(\boldsymbol{N}(x,y)), \tag{11}$$

where $\mathcal{L}_{\text{env}}^{\text{diffuse}}$ is the pre-integrated diffuse environment irradiance. Following the split-sum approximation, we compute the specular component efficiently:

$$\begin{aligned}&\boldsymbol{L}_s(x,y,\omega_o) \approx \text{BRDF}_{\text{LUT}}(\boldsymbol{N}(x,y)\cdot\omega_o, R(x,y))\cdot \\ &(V(x,y)\boldsymbol{L}_{\text{dir}}(x,y,\omega_r,R(x,y)) + (1-V(x,y))\boldsymbol{L}_{\text{ind}}(x,y)).\end{aligned} \tag{12}$$

The first term, $\text{BRDF}_{\textbf{LUT}}$, depends solely on the view angle and roughness, which is precomputed and stored in a 2D lookup texture. $\boldsymbol{L}_{\text{dir}}(x,y,\omega_r,R(x,y))$ is the direct environment lighting queried from a learnable environment map in the reflection direction ω_r, using roughness $R(x,y)$ for mipmap selection. $\boldsymbol{L}_{\text{ind}}(x,y)$ is the blended indirect lighting component. We follow the baseline method [73] in modeling inter-reflections by approximating visibility with ray tracing an extracted mesh and encoding indirect lighting with spherical harmonics. We follow prior work [73] and use [44] for the PBR shading.

3.6. Training

Splat parameters $\boldsymbol{p}_k, \boldsymbol{t}_{u_k}, \boldsymbol{t}_{v_k}, s_{u_k}, s_{v_k}, o_k$, their material texture maps $\mathcal{T}_k^{\boldsymbol{\lambda}}, \mathcal{T}_k^m, \mathcal{T}_k^r, \mathcal{T}_k^{\boldsymbol{n}^t}$, per-splat indirect lighting SH coefficients for $\boldsymbol{l}_k^{\text{ind}}$, and the environment maps $(\mathcal{L}_{\text{env}}^{\text{diffuse}}, \mathcal{L}_{\text{env}}^{\text{spec}})$, are optimized end-to-end using a composite loss:

$$\mathcal{L} = \mathcal{L}_{\text{img}} + \lambda_{\text{n}}\mathcal{L}_{\text{n}}, \tag{13}$$

where $\mathcal{L}_{\text{img}} = (1-\lambda)\mathcal{L}_1 + \lambda\mathcal{L}_{\text{D-SSIM}}$ is the RGB reconstruction loss with balancing weight $\lambda = 0.2$. The normal consistency loss $\mathcal{L}_n = 1 - \tilde{\boldsymbol{N}}(x,y)^{\mathrm{T}}\boldsymbol{N}(x,y)$ encourages alignment of the Gaussians with the surface by minimizing the cosine difference between the rendered normal $\boldsymbol{N}(x,y)$ and the surface normal $\tilde{\boldsymbol{N}}(x,y)$ derived from rendered depth. We use a single NVIDIA RTX A6000 GPU in our experiments.

4. Experiments

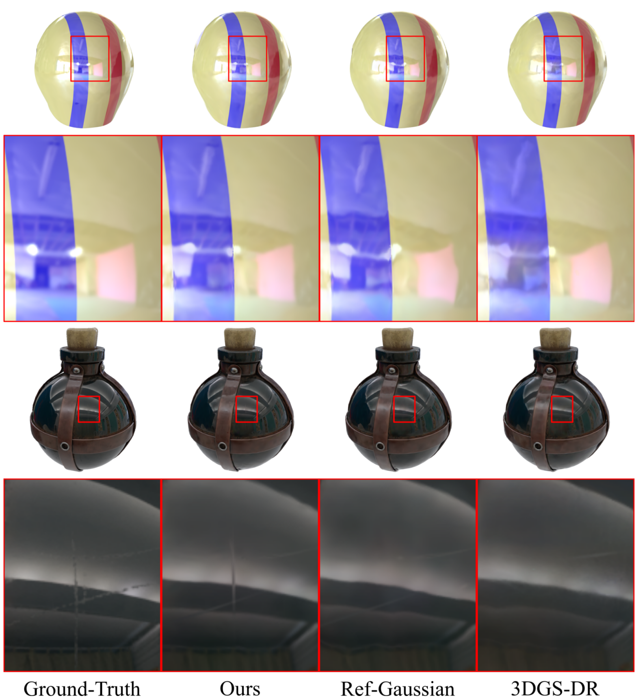

Figure 2. Qualitative comparisons of novel view synthesis on synthetic scenes. From top to bottom: helmet from Shiny Blender [63] and potion from Glossy Synthetic [40]. Notice how we reconstruct reflections with more fidelity and less distortion.

Following the baseline method [73], we evaluate our work quantitatively and qualitatively under standard multi-view reconstruction benchmarks of challenging reflective scenes. We use the datasets: Shiny Blender [63] and Glossy Synthetic [40] for novel view synthesis of reflective objects, and dataset Ref-Real [63] to account for real world open reflective scenes. We also provide results on the Synthetic NeRF [42] dataset to showcase our method under non-reflective scenes and illustrate its practicality in the supplementary material (Section 2). We compare to state-of-the-art methods in the reflective scene setting, including the baseline Ref-Gaussian [73], other reflective Gaussian splatting based methods GShader [24] and 3DGS-DR [76], 3DGS [27] and 2DGS [18] for reference, and seminal NeRF based approaches such as Ref-NeRF [63] and ENVIDR [35]. We provide additional results and ablation studies in the supplementary material.

Figure 3. Qualitative comparisons of novel view synthesis on real scenes [63]. From left to right: garden spheres and sedan. Notice how we recover reflections with more fidelity.

4.1. Implementation Details

We follow a two-stage optimization for stability. We first train for half the total number of iterations using per-splat single attribute optimization. During the second stage, we start optimizing the material and normal textures initialized from the corresponding attributes from the first stage, and we freeze the positions of the primitives. In all our evaluations, we use texture resolution of 2×2. We use the same hyperparameters and training strategies defined by our baseline Ref-Gaussian [73]. We also follow the latter in replacing the integrated diffuse lighting by a spherical harmonics view dependent color for better fitting in the reflective setting. We implement efficient CUDA kernels on top of 2DGS and Ref-Gaussian for forward and backward operations involving material textures and normal mapping, as well as the texture atlases construction, packing and hardware bilinear filtering at test time.

4.2. Novel View Synthesis

Table 1 shows numerical results in the standard reflective benchmark. We report Peak Signal-to-Noise Ratio (PSNR), Structural Similarity Index Measure (SSIM) [67], and Learned Perceptual Image Patch Similarity (LPIPS)

Table 1. Per-scene image quality comparison in the reflective novel view synthesis setting.

Datasets		Shiny Blender [63]						Glossy Synthetic [40]								Real [63]		
		ball	car	coffee	helmet	teapot	toaster	angel	bell	cat	horse	luyu	potion	tbell	teapot	garden	sedan	toycar
PSNR ↑	Ref-NeRF	33.16	30.44	33.99	29.94	45.12	26.12	20.89	30.02	29.76	19.30	25.42	30.11	26.91	22.77	22.01	25.21	23.65
	ENVIDR	41.02	27.81	30.57	32.71	42.62	26.03	29.02	30.88	31.04	25.99	28.03	32.11	28.64	26.77	21.47	24.61	22.92
	3DGS	27.65	27.26	32.30	28.22	45.71	20.99	24.49	25.11	31.36	24.63	26.97	30.16	23.88	21.51	21.75	26.03	23.78
	2DGS	25.97	26.38	32.31	27.42	44.97	20.42	26.95	24.79	30.65	25.18	26.89	29.50	23.28	21.29	22.53	26.23	23.70
	GShader	30.99	27.96	32.39	28.32	45.86	26.28	25.08	28.07	31.81	26.56	27.18	30.09	24.48	23.58	21.74	24.89	23.76
	3DGS-DR	33.43	30.48	34.53	31.44	47.04	26.76	29.07	30.60	32.59	26.17	28.96	32.65	29.03	25.77	21.82	26.32	23.83
	Ref-Gaussian	36.07	31.32	34.2	32.3	47.15	28.28	30.55	28.57	33.04	26.76	30.1	33.39	30.1	25.97	23.09	26.23	24.74
	Ours	39.27	31.72	34.82	32.97	48.27	28.4	30.85	29.16	33.51	27.08	30.48	34.03	30.77	26.42	23.34	26.45	24.95
SSIM ↑	Ref-NeRF	0.971	0.950	0.972	0.954	0.995	0.921	0.853	0.941	0.944	0.820	0.901	0.933	0.947	0.897	0.584	0.720	0.633
	ENVIDR	0.997	0.943	0.962	0.987	0.995	0.922	0.934	0.954	0.965	0.925	0.931	0.960	0.947	0.957	0.561	0.707	0.549
	3DGS	0.937	0.931	0.972	0.951	0.996	0.894	0.792	0.908	0.959	0.797	0.916	0.938	0.900	0.881	0.571	0.771	0.637
	2DGS	0.934	0.930	0.972	0.953	0.997	0.892	0.918	0.911	0.958	0.909	0.918	0.939	0.902	0.886	0.609	0.778	0.597
	GShader	0.966	0.932	0.971	0.951	0.996	0.929	0.914	0.919	0.961	0.933	0.914	0.936	0.898	0.901	0.576	0.728	0.637
	3DGS-DR	0.979	0.963	0.976	0.971	0.997	0.942	0.942	0.959	0.973	0.933	0.943	0.959	0.958	0.942	0.581	0.773	0.639
	Ref-Gaussian	0.985	0.966	0.976	0.971	0.997	0.952	0.956	0.943	0.975	0.942	0.953	0.966	0.947	0.942	0.628	0.766	0.679
	Ours	0.992	0.969	0.977	0.975	0.998	0.954	0.958	0.947	0.977	0.946	0.956	0.97	0.97	0.947	0.631	0.772	0.688
LPIPS ↓	Ref-NeRF	0.166	0.050	0.082	0.086	0.012	0.083	0.144	0.102	0.104	0.155	0.098	0.084	0.114	0.098	0.251	0.234	0.231
	ENVIDR	0.020	0.046	0.083	0.036	0.009	0.081	0.067	0.054	0.049	0.065	0.059	0.072	0.069	0.041	0.263	0.387	0.345
	3DGS	0.162	0.047	0.079	0.081	0.008	0.125	0.088	0.104	0.062	0.077	0.064	0.093	0.102	0.125	0.248	0.206	0.237
	2DGS	0.156	0.052	0.079	0.079	0.008	0.127	0.072	0.109	0.060	0.071	0.066	0.097	0.125	0.101	0.254	0.225	0.396
	GShader	0.121	0.044	0.078	0.074	0.007	0.079	0.082	0.098	0.056	0.562	0.064	0.088	0.091	0.122	0.274	0.259	0.239
	3DGS-DR	0.105	0.033	0.076	0.050	0.006	0.082	0.052	0.050	0.042	0.057	0.048	0.068	0.059	0.060	0.247	0.208	0.231
	Ref-Gaussian	0.089	0.031	0.078	0.048	0.006	0.067	0.040	0.067	0.037	0.049	0.043	0.061	0.070	0.059	0.266	0.258	0.257
	Ours	0.074	0.028	0.075	0.042	0.004	0.063	0.038	0.064	0.034	0.045	0.040	0.055	0.041	0.055	0.283	0.257	0.27

[81] as metrics. Our method performs favorably compared to other methods, including baseline Ref-Gaussian, which is also the state-of-the-art method currently under this benchmark to the best of our knowledge. We provide qualitative results to accompany this table, for real scenes in Figure 3 and synthetic ones in Figure 2. Our method displays superior ability in capturing reflections on the surface with more fidelity, while Ref-Gaussian and 3DGS-DR suffer from distorted or missing reflections. We also show in the supplementary material numerical (Tab. 1 in Supp. Mat.) and qualitative (Fig. 1 in Supp. Mat.) comparisons under non-reflective data on Nerf Synthetic Scenes [42]. In that generic setting, we perform competitively with the state-of-the-art. The improvement brought by our method over the baseline Ref-Gaussian particularly, under both reflective and non-reflective scenes, is a testimony of the efficacy and versatility of our representation.

4.3. Scene Decomposition

Figure 4 shows a comparison of the decomposition with respect to our baseline. Notice that our material properties are sharper and display less noise. Our normals replicate the smooth sphere shape more faithfully. We provide further comparisons in supplementary material including environment map estimation and material decomposition (Figures 3 & 2 in Supp. Mat.) showcasing the superiority of our results.

We further evaluate our normal estimation through the benchmark of the Shiny Blender dataset [63]. Table 2 reports the mean angular error of normal maps, where we outperform our baseline. This result validates our tangential normal representation and the use of normal mapping. Figure 5 shows qualitative comparisons of normals, where we recover superior geometry compared to other methods.

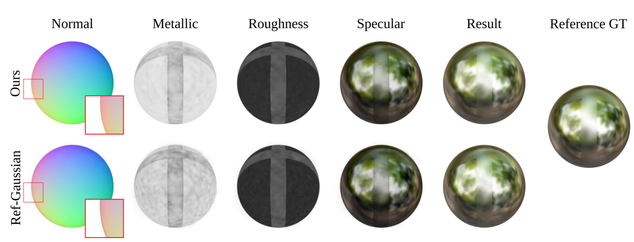

Figure 4. Comparison of scene decomposition between our method and the baseline.

4.4. Hardware Acceleration Performance

While our per-primitive texture mapping approach improves rendering quality for reflective scenes, a potential concern is the additional computational cost of tex-

Table 2. Normal quality evaluated by MAE°: comparisons on the Shiny Blender Dataset [63].

	GShader	NVDiffRec	ENVIDR	3DGS-DR	Ref-Gaussian	Ours
MAE° ↓	22.31	17.02	4.618	4.871	2.078	1.78

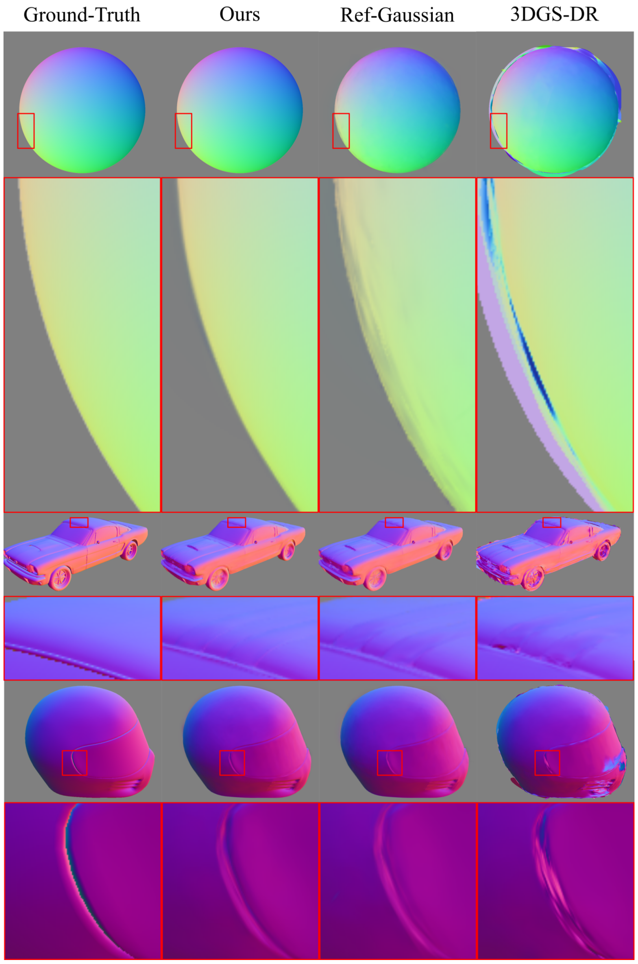

Figure 5. Qualitative comparisons of normal reconstruction by different methods.

ture sampling. To address this, we implemented hardware-accelerated texture filtering using texture atlases as described in Section 3.4. In this section, we evaluate the performance benefits of this implementation compared to software-based bilinear filtering.

We trained our model on the Shiny Blender dataset using different texture resolutions: 2×2, 4×4, 8×8 and 16×16. For each model, we then rendered the scenes using both the software implementation of bilinear filtering and our hardware-accelerated implementation with texture atlases. We measured the rendering performance in frames per second (FPS) and compared these values to the baseline method without textures.

As shown in Table 3, software-based bilinear filtering introduces some performance overhead compared to the baseline. This overhead increases with higher texture resolutions. In contrast, our hardware-accelerated implementation with texture atlases maintains rendering performance very close to the baseline method in comparison.

Table 3. Rendering performance comparison between software bilinear filtering and hardware-accelerated texture atlas filtering at different texture resolutions. Values represent the ratio of FPS compared to the baseline method without textures, averaged across the Shiny Blender dataset scenes [63].

	2×2 Textures	4×4 Textures	8×8 Textures	16×16 Textures
Baseline, No Textures	× 1.00	× 1.00	× 1.00	× 1.00
Software Bilinear	× 0.90	× 0.86	× 0.85	× 0.79
Hardware Texture Atlas	× 0.92	× 0.93	× 0.94	× 0.91

These results demonstrate that the texture atlas approach effectively leverages GPU hardware capabilities to minimize the performance impact of texture filtering. Several factors contribute to this efficiency including the use of dedicated texture units that are designed specifically for texture filtering operations, and are substantially faster than general-purpose compute.

We note that there is still room for improvement as we do not follow any strategy for packing the texture into the atlases, which could benefit from better locality if considering only primitives that are used for each frame or packing textures belonging to nearby primitives next to each other.

By leveraging hardware acceleration through texture atlases, we can achieve the best of both worlds: the improved rendering quality of per-primitive textures for reflective scenes while maintaining rendering speed comparable to methods without textures.

4.5. Ablation Study

To evaluate the effectiveness of our per-primitive texture mapping approach and understand the contribution of individual components, we conducted a series of ablation studies isolating different aspects of our method.

Additional experiments can be found in Sec. 3 of the supplementary material, notably a comparative analysis of Texture Mapping as opposed to increasing primitive count, and a comparative analysis of performance under reduced primitive count (Tab. 2 & Fig. 4 in Supp. Mat.).

Impact of Normal Mapping Our previous experiments suggested that normal mapping plays a particularly important role in the performance improvements observed with our method. To isolate this effect, we implemented a variant of the baseline that uses texture mapping only for normals while keeping other attributes (albedo, roughness, metallic) as per-primitive constants.

The results in Table 4 confirm that normal mapping alone accounts for a substantial portion of the performance improvement. This is particularly evident in the normal mean

Table 4. Ablation study: Effect of normal mapping on average image and normal quality. We evaluate on the Shiny Blender dataset [63].

Metric	Ref-Gaussian	+ Normal Mapping	Full Model(Ours)
PSNR ↑	34.89	35.33	35.90
SSIM ↑	0.974	0.975	0.978
LPIPS ↓	0.053	0.053	0.047
Normal MAE ↓	2.078	1.783	1.78

Table 5. Ablation study: Effect of texture resolution on storage, average image and normal quality. We evaluate on the Shiny Blender dataset [63].

Variants	PSNR ↑	SSIM ↑	LPIPS ↓	Normal MAE ↓	Storage
Ref-Gaussian	34.88	0.974	0.053	2.078	× 1.00
2×2 Textures, Same Storage	35.66	0.976	0.048	1.823	× 1.00
2×2 Textures	35.90	0.977	0.047	1.780	× 1.20
4×4 Textures, Same Storage	35.33	0.977	0.048	1.881	× 1.00
4×4 Textures	35.94	0.978	0.047	1.830	× 1.96

angular error (MAE) metric, where normal mapping significantly reduces the error compared to the baseline.

The effectiveness of normal mapping for reflective scenes can be attributed to the fact that reflective surfaces often exhibit high-frequency normal variations that are difficult to capture with a single normal per primitive and that more accurate and detailed normals lead to more precise reflection directions and therefore better specular highlights which is achieved by our spatially varying normals.

While normal mapping provides significant improvement, our full model with texture mapping for all material properties achieves the best overall performance, demonstrating that each component contributes to the final quality.

Impact of Texture Resolution and Storage The results in Table 5 show different configurations of our method in terms of the texture resolution, and whether we add them on top of the baseline or reduce the primitive count so that the total storage size is equal to the baseline. We perform this evaluation on the Shiny Blender dataset [63]. We experiment with resolutions 2×2 and 4×4 as we find that for the type of scenes in this dataset (object at the center of the scene and orbital views), enhancing primitives with much higher texture resolutions does not result in a desirable storage to quality tradeoff. Using higher resolutions can still be beneficial though for other scene types where some views are close to the object for instance.

We find that at a fixed storage budget, our method still achieves better performance compared to the baseline method in all metrics. This demonstrates the efficiency of our texture-based representation and that the memory overhead is not a fixed cost but a tunable parameter. Our framework allows for adapting texture resolutions or selectively texturing only the most critical attributes to fit different memory budgets.

Supplementary material includes additional comparisons to the baseline with increased primitive count as well as the impact on performance with reduced primitive count; These experiments show that for reflective scenes, investing the parameter budget in appearance/normal complexity (textures) is often more effective than investing it purely in geometric complexity (primitives).

5. Limitations and Discussion

Our method comes with some limitations that present opportunities for future work.

Uniform Texture Resolution In the current implementation, we assign textures of uniform resolution to all primitives regardless of their size or importance in the scene. This approach can be inefficient for large unbounded scenes where distant primitives occupy few pixels but still receive the same texture resolution as foreground elements.

Filtering Limitations Our implementation currently relies on bilinear filtering for texture sampling. While effective for our evaluated scenes, it does not fully resolve texture minification artifacts that might manifest in more challenging scenarios.

Despite these limitations, our experimental results demonstrate that per-primitive texture mapping significantly improves the visual quality of reflective scenes while maintaining real-time rendering performance.

6. Conclusion

We presented a method that enhances 2D Gaussian Splatting for reflective scenes by introducing per-primitive texture mapping. By leveraging the flat nature of 2D Gaussians to define textures of material properties, our approach enables high-frequency detail representation without increasing primitive count. Our hardware-accelerated implementation using texture atlases demonstrates that classical computer graphics techniques can be effectively integrated with modern differentiable rendering approaches. The results show that this representation significantly improves the quality of specular reflections, particularly through detailed normal mapping, while maintaining real-time performance. Our work bridges the gap between explicit primitive-based representations and high-quality material modeling, offering advantages of both approaches.

Acknowledgments

This work was granted access to the HPC resources of IDRIS under the allocation 20XX-AD010616156 made by GENCI.

References

[1] Kelian Baert, Mae Younes, Francois Bourel, Marc Christie, and Adnane Boukhayma. Gtavatar: Bridging gaussian splatting and texture mapping for relightable and editable gaussian avatars. *arXiv preprint*, 2025. 3

[2] Yizhak Ben-Shabat, Chamin Hewa Koneputugodage, and Stephen Gould. Digs: Divergence guided shape implicit neural representation for unoriented point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 19321–19330, 2022. 2

[3] Brent Burley and Dylan Lacewell. Ptex: Per-face texture mapping for production rendering. In *Proceedings of Eurographics Symposium on Rendering*. Eurographics Association, 2008. 4

[4] Brian Chao, Hung-Yu Tseng, Lorenzo Porzi, Chen Gao, Tuotuo Li, Qinbo Li, Ayush Saraf, Jia-Bin Huang, Johannes Kopf, Gordon Wetzstein, and Changil Kim. Textured gaussians for enhanced 3d scene appearance modeling, 2025. 3

[5] David Charatan, Sizhe Lester Li, Andrea Tagliasacchi, and Vincent Sitzmann. pixelsplat: 3d gaussian splats from image pairs for scalable generalizable 3d reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 19457–19467, 2024. 2

[6] Anpei Chen, Zexiang Xu, Fuqiang Zhao, Xiaoshuai Zhang, Fanbo Xiang, Jingyi Yu, and Hao Su. Mvsnerf: Fast generalizable radiance field reconstruction from multi-view stereo. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 14124–14133, 2021. 2

[7] Anpei Chen, Zexiang Xu, Andreas Geiger, Jingyi Yu, and Hao Su. Tensorf: Tensorial radiance fields. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2022. 2

[8] Kangle Deng, Andrew Liu, Jun-Yan Zhu, and Deva Ramanan. Depth-supervised nerf: Fewer views and faster training for free. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12872–12881, 2022. 2

[9] Stavros Diolatzis, Tobias Zirr, Alexandr Kuznetsov, Georgios Kopanas, and Anton Kaplanyan. N-dimensional gaussians for fitting of high dimensional functions. In *Proceedings of ACM SIGGRAPH (Conference Track)*. ACM, 2024. 2

[10] Sara Fridovich-Keil, Alex Yu, Matthew Tancik, Qinhong Chen, Benjamin Recht, and Angjoo Kanazawa. Plenoxels: Radiance fields without neural networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5501–5510, 2022. 2

[11] Sara Fridovich-Keil, Giacomo Meanti, Frederik Rahbæk Warburg, Benjamin Recht, and Angjoo Kanazawa. K-planes: Explicit radiance fields in space, time, and appearance. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023. 2

[12] Yang Fu, Sifei Liu, Amey Kulkarni, Jan Kautz, Alexei A. Efros, and Xiaolong Wang. Colmap-free 3d gaussian splatting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 20796–20805, 2024. 2

[13] Jian Gao, Chun Gu, Youtian Lin, Zhihao Li, Hao Zhu, Xun Cao, Li Zhang, and Yao Yao. Relightable 3d gaussians: Realistic point cloud relighting with brdf decomposition and ray tracing. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2024. 2

[14] Wenhang Ge, Tao Hu, Haoyu Zhao, Shu Liu, and Ying-Cong Chen. Ref-neus: Ambiguity-reduced neural implicit surface learning for multi-view reconstruction with reflection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 4251–4260. IEEE, 2023. 2

[15] Henri Gouraud. Continuous shading of curved surfaces. In *Proceedings of the 2nd conference on Computer graphics and interactive techniques*, pages 747–755. ACM, 1971. 1

[16] Amos Gropp, Lior Yariv, Niv Haim, Matan Atzmon, and Yaron Lipman. Implicit geometric regularization for learning shapes. In *Proceedings of Machine Learning and Systems (MLSys) 2020*, pages 3569–3579. 2020. 2

[17] Aaron Hanson, Zhihao Li, Yufei Wang, et al. Speedy-splat: Fast 3d gaussian splatting with sparse pixels and sparse gaussians. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025. 2

[18] 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

[19] Jiahui Huang, Zan Gojcic, Matan Atzmon, Or Litany, Sanja Fidler, and Francis Williams. Neural kernel surface reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4369–4379, 2023. 2

[20] Ajay Jain, Matthew Tancik, and Pieter Abbeel. Putting nerf on a diet: Semantically consistent few-shot view synthesis. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021. 2

[21] Shubhendu Jena, Franck Multon, and Adnane Boukhayma. Geotransfer: Generalizable few-shot multi-view reconstruction via transfer learning. In *Proceedings of the ECCV 2024 Workshop on (insert full workshop name here)*, 2024. 2

[22] Shubhendu Jena, Amine Ouasfi, Mae Younes, and Adnane Boukhayma. Sparfels: Fast reconstruction from sparse unposed imagery. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2025. 2

[23] Shubhendu Jena, Shishir Reddy Vutukur, and Adnane Boukhayma. Sparsplat: Fast multi-view reconstruction with generalizable 2d gaussian splatting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2025. 2

[24] Yingwenqi Jiang, Jiadong Tu, Yuan Liu, Xifeng Gao, Xiaoxiao Long, Wenping Wang, and Yuexin Ma. Gaussianshader: 3d gaussian splatting with shading functions for reflective surfaces. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1185–1194. IEEE, 2024. 2, 5

[25] Zijie Jiang, Tianhan Xu, and Hiroharu Kato. Rethinking directional parameterization in neural implicit surface recon-

struction. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 127–142. Springer, 2024. 2

[26] Mohammad Mahdi Johari, Yann Lepoittevin, and François Fleuret. Geonerf: Generalizing nerf with geometry priors. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18365–18375, 2022. 2

[27] 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, 2, 3, 5

[28] Shakiba Kheradmand, Daniel Rebain, Gopal Sharma, Weiwei Sun, Yang-Che Tseng, Hossam Isack, Abhishek Kar, Andrea Tagliasacchi, and Kwang Moo Yi. 3d gaussian splatting as markov chain monte carlo. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024. Spotlight Presentation. 2

[29] Lei Lan, Tianjia Shao, Zixuan Lu, Yu Zhang, Chenfanfu Jiang, and Yin Yang. 3dgs2: Near second-order converging 3d gaussian splatting. *arXiv preprint arXiv:2501.13975*, 2025. 2

[30] Joo Chan Lee, Daniel Rho, Xiangyu Sun, Jong Hwan Ko, and Eunbyung Park. Compact 3d gaussian representation for radiance field. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 21719–21728, 2024. 2

[31] Marc Levoy. Display of surfaces from volume data. *IEEE Computer Graphics and Applications*, 8(3):29–37, 1988. 2

[32] Qian Li, Franck Multon, and Adnane Boukhayma. Learning generalizable light field networks from few images. In *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5. IEEE, 2023. 2

[33] Qian Li, Franck Multon, and Adnane Boukhayma. Regularizing neural radiance fields from sparse rgb-d inputs. In *Proceedings of the IEEE International Conference on Image Processing (ICIP)*, pages 2320–2324, 2023. 2

[34] Zhaoshuo Li, Thomas Müller, Alex Evans, Russell H Taylor, Mathias Unberath, Ming-Yu Liu, and Chen-Hsuan Lin. Neuralangelo: High-fidelity neural surface reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8456–8465, 2023. 2

[35] Ruofan Liang, Huiting Chen, Chunlin Li, Fan Chen, Selvakumar Panneer, and Nandita Vijaykumar. Envidr: Implicit differentiable renderer with neural environment lighting. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 79–89. IEEE, 2023. 2, 5

[36] Zhihao Liang, Qi Zhang, Ying Feng, Ying Shan, and Kui Jia. Gs-ir: 3d gaussian splatting for inverse rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 21644–21653, 2024. 2

[37] Zhihao Liang, Qi Zhang, Wenbo Hu, Lei Zhu, Ying Feng, and Kui Jia. Analytic-splatting: Anti-aliased 3d gaussian splatting via analytic integration. In *European conference on computer vision*, pages 281–297. Springer, 2024. 2

[38] Lingjie Liu, Jiatao Gu, Kyaw Zaw Lin, Tat-Seng Chua, and Christian Theobalt. Neural sparse voxel fields. In *NeurIPS*, 2020. 2

[39] Tianqi Liu, Guangcong Wang, Shoukang Hu, Liao Shen, Xinyi Ye, Yuhang Zang, Zhiguo Cao, Wei Li, and Ziwei Liu. Mvsgaussian: Fast generalizable gaussian splatting reconstruction from multi-view stereo. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 2662–2678, 2024. 2

[40] Yuan Liu, Peng Wang, Cheng Lin, Xiaoxiao Long, Jiepeng Wang, Lingjie Liu, Taku Komura, and Wenping Wang. Nero: Neural geometry and brdf reconstruction of reflective objects from multiview images. *ACM Trans. Graph.*, 42(4), 2023. 2, 5, 6

[41] Nelson Max. Optical models for direct volume rendering. *IEEE Transactions on Visualization and Computer Graphics*, 1(2):99–108, 1995. 2

[42] Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *ECCV*, pages 405–421. Springer, 2020. 1, 2, 5, 6

[43] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a multiresolution hash encoding. *ACM transactions on graphics (TOG)*, 41(4):1–15, 2022. 2

[44] Jacob Munkberg, Jon Hasselgren, Tianchang Shen, Jun Gao, Wenzheng Chen, Alex Evans, Thomas Müller, and Sanja Fidler. Extracting Triangular 3D Models, Materials, and Lighting From Images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8280–8290, 2022. 4

[45] Michael Niemeyer, Jonathan T. Barron, Ben Mildenhall, Mehdi S. M. Sajjadi, Andreas Geiger, and Noha Radwan. Regnerf: Regularizing neural radiance fields for view synthesis from sparse inputs. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 2

[46] Amine Ouasfi and Adnane Boukhayma. Few'zero level set'-shot learning of shape signed distance functions in feature space. In *ECCV*, 2022. 2

[47] Amine Ouasfi and Adnane Boukhayma. Mixing-denoising generalizable occupancy networks. *3DV*, 2024. 2

[48] Amine Ouasfi and Adnane Boukhayma. Few-shot unsupervised implicit neural shape representation learning with spatial adversaries. In *Proceedings of the 41st International Conference on Machine Learning (ICML)*, pages 38905–38918, 2024. 2

[49] Amine Ouasfi and Adnane Boukhayma. Robustifying generalizable implicit shape networks with a tunable nonparametric model. *Advances in Neural Information Processing Systems*, 36, 2024. 2

[50] Amine Ouasfi and Adnane Boukhayma. Unsupervised occupancy learning from sparse point cloud. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 21729–21739, 2024.

[51] Amine Ouasfi, Shubhendu Jena, Eric Marchand, and Adnane Boukhayma. Toward robust neural reconstruction from

sparse point sets. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025. 2

[52] Songyou Peng, Michael Niemeyer, Lars Mescheder, Marc Pollefeys, and Andreas Geiger. Convolutional occupancy networks. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part III 16*, pages 523–540. Springer, 2020. 2

[53] Bui Tuong Phong. Illumination for computer generated pictures. *Communications of the ACM*, 18(6):311–317, 1975. 1

[54] Budirijanto Purnomo, Jonathan D Cohen, and Subodh Kumar. Seamless texture atlases. In *IEEE Visualization, 2004. VIS 2004.*, pages 273–280. IEEE, 2004. 4

[55] Victor Rong, Jingxiang Chen, Sherwin Bahmani, Kiriakos N Kutulakos, and David B Lindell. Gstex: Per-primitive texturing of 2d gaussian splatting for decoupled appearance and geometry modeling. In *2025 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 3508–3518. IEEE, 2025. 3

[56] Yunzhou Song, Heguang Lin, Jiahui Lei, Lingjie Liu, and Kostas Daniilidis. Hdgs: Textured 2d gaussian splatting for enhanced scene rendering. *arXiv preprint arXiv:2412.01823*, 2024. 3

[57] Pratul P. Srinivasan, Boyang Deng, Xiuming Zhang, Matthew Tancik, Ben Mildenhall, and Jonathan T. Barron. Nerv: Neural reflectance and visibility fields for relighting and view synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7495–7504, 2021. 2

[58] Cheng Sun, Jaesung Choe, Charles Loop, Wei-Chiu Ma, and Yu-Chiang Frank Wang. Sparse voxels rasterization: Real-time high-fidelity radiance field rendering. *arXiv preprint arXiv:2412.04459*, 2024. 2

[59] Jingxiang Sun, Yiming Gao, Xuan Wang, Qi Zhang, Hujun Bao, and Xiaowei Zhou. Improved direct voxel grid optimization for radiance fields, 2022. 2

[60] David Svitov, Pietro Morerio, Lourdes Agapito, and Alessio Del Bue. Billboard splatting (bbsplat): Learnable textured primitives for novel view synthesis. *arXiv preprint arXiv:2411.08508*, 2024. 3

[61] Shengji Tang, Weicai Ye, Peng Ye, Weihao Lin, Yang Zhou, Tao Chen, and Wanli Ouyang. Hisplat: Hierarchical 3d gaussian splatting for generalizable sparse-view reconstruction. In *Proceedings of the 2025 International Conference on Learning Representations (ICLR)*, 2025. 2

[62] Zhe Jun Tang and Tat-Jen Cham. 3igs: Factorised tensorial illumination for 3d gaussian splatting. In *Proceedings of the European Conference on Computer Vision (ECCV)*. Springer, 2024. 2

[63] 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 *Proceedings of the IEEE/CVF CVPR*, pages 5481–5490. IEEE, 2022. 2, 5, 6, 7, 8

[64] Peng Wang, Lingjie Liu, Yuan Liu, Christian Theobalt, Taku Komura, and Wenping Wang. Neus: Learning neural implicit surfaces by volume rendering for multi-view reconstruction. *arXiv preprint arXiv:2106.10689*, 2021. 2

[65] Peng Wang, Zhihao Li, Jong Hwan Ko, and Eunbyung Park. Steepest descent density control for compact 3d gaussian splatting. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025. 2

[66] Yufei Wang, Zhihao Li, Lanqing Guo, Wenhan Yang, Alex Kot, and Bihan Wen. Compact 3d gaussian splatting with anchor level context model. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024. 2

[67] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 13(4):600–612, 2004. 5

[68] Sebastian Weiss and Derek Bradley. Gaussian billboards: Expressive 2d gaussian splatting with textures. *arXiv preprint arXiv:2412.12734*, 2024. 3

[69] Guanjun Wu, Taoran Yi, Jiemin Fang, Lingxi Xie, Xiaopeng Zhang, Wei Wei, Wenyu Liu, Qi Tian, and Xinggang Wang. 4d gaussian splatting for real-time dynamic scene rendering. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 20310–20320, 2024. 2

[70] Tong Wu, Jiaqi Wang, Xingang Pan, Xudong Xu, Christian Theobalt, Ziwei Liu, and Dahua Lin. Voxurf: Voxel-based efficient and accurate neural surface reconstruction. *arXiv preprint arXiv:2208.12697*, 2022. 2

[71] Rui Xu, Wenyue Chen, Jiepeng Wang, Yuan Liu, Peng Wang, Lin Gao, Shiqing Xin, Taku Komura, Xin Li, and Wenping Wang. Supergaussians: Enhancing gaussian splatting using primitives with spatially varying colors. *arXiv preprint arXiv:2411.18966*, 2024. 3

[72] Jiawei Yang, Marco Pavone, and Yue Wang. Freenerf: Improving few-shot neural rendering with free frequency regularization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8254–8263, 2023. 2

[73] Yuxuan Yao, Zixuan Zeng, Chun Gu, Xiatian Zhu, and Li Zhang. Reflective gaussian splatting. In *ICLR*, 2025. 1, 2, 4, 5

[74] Lior Yariv, Jiatao Gu, Yoni Kasten, and Yaron Lipman. Volume rendering of neural implicit surfaces. *Advances in Neural Information Processing Systems*, 34:4805–4815, 2021. 2

[75] Botao Ye, Sifei Liu, Haofei Xu, Xueting Li, Marc Pollefeys, Ming-Hsuan Yang, and Songyou Peng. No pose, no problem: Surprisingly simple 3d gaussian splats from sparse unposed images. In *Proceedings of the 2025 International Conference on Learning Representations (ICLR)*, 2025. 2

[76] Keyang Ye, Qiming Hou, and Kun Zhou. 3d gaussian splatting with deferred reflection. In *ACM SIGGRAPH Conference Proceedings*, 2024. 1, 2, 4, 5

[77] Mae Younes and Adnane Boukhayma. Anti-aliased 2d gaussian splatting. In *Proceedings of the 39th Conference on Neural Information Processing Systems (NeurIPS)*, 2025. Poster. 2

[78] Mae Younes, Amine Ouasfi, and Adnane Boukhayma. Sparsecraft: Few-shot neural reconstruction through stereopsis

guided geometric linearization. In *Proceedings of the European Conference on Computer Vision (ECCV)*. Springer, 2024. 2

[79] Alex Yu, Vickie Ye, Matthew Tancik, and Angjoo Kanazawa. pixelnerf: Neural radiance fields from one or few images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4578–4587, 2021. 2

[80] Zehao Yu, Anpei Chen, Binbin Huang, Torsten Sattler, and Andreas Geiger. Mip-splatting: Alias-free 3d gaussian splatting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 19447–19456, 2024. 2

[81] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 586–595, 2018. 6

[82] Zheng Zhang, Wenbo Hu, Yixing Lao, Tong He, and Hengshuang Zhao. Pixel-gs: Density control with pixel-aware gradient for 3d gaussian splatting. In *European Conference on Computer Vision (ECCV)*, 2024. 2

[83] Haoyang Zhao, Zhihao Li, Yufei Wang, et al. Grendel: On scaling up 3d gaussian splatting training. In *International Conference on Learning Representations (ICLR)*, 2025. 2

[84] Lingzhe Zhao, Peng Wang, and Peidong Liu. Bad-gaussians: Bundle adjusted deblur gaussian splatting. In *European Conference on Computer Vision*, pages 233–250. Springer, 2024. 2

[85] Zuo-Liang Zhu, Beibei Wang, and Jian Yang. Gs-ror: 3d gaussian splatting for reflective object relighting via sdf priors. *arXiv preprint arXiv:2406.18544*, 2024. 2

[86] Matthias Zwicker, Hanspeter Pfister, Jeroen Van Baar, and Markus Gross. Ewa volume splatting. In *Proceedings of the IEEE Conference on Visualization (VIS)*, pages 29–36. IEEE, 2001. 2

[87] Matthias Zwicker, Hanspeter Pfister, Jeroen Van Baar, and Markus Gross. Ewa splatting. *IEEE Transactions on Visualization and Computer Graphics*, 8(3):223–238, 2002. 1, 2