GaussianShader: 3D Gaussian Splatting with Shading Functions for Reflective Surfaces

Yingwenqi Jiang[1] Jiadong Tu[1] Yuan Liu[2] Xifeng Gao[3]
Xiaoxiao Long[2,*] Wenping Wang[4] Yuexin Ma[1,*]

[1]ShanghaiTech University [2]The University of Hong Kong [3]Tencent America [4]Texas A&M University

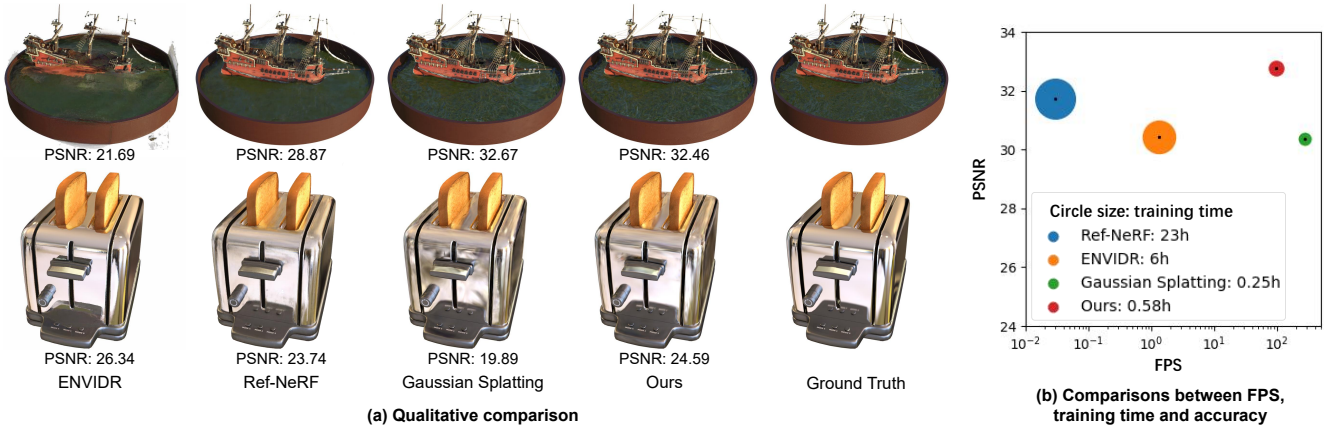

Figure 1. GaussianShader maintains real-time rendering speed and renders high-fidelity images for both general and reflective surfaces. Ref-NeRF[45] and ENVIDR[27] attempt to handle reflective surfaces, but they suffer from quite time-consuming optimization and slow rendering speed. 3D Gaussian splatting [21] keeps high efficiency but cannot handle such reflective surfaces.

Abstract

The advent of neural 3D Gaussians [21] has recently brought about a revolution in the field of neural rendering, facilitating the generation of high-quality renderings at real-time speeds. However, the explicit and discrete representation encounters challenges when applied to scenes featuring reflective surfaces. In this paper, we present ***GaussianShader****, a novel method that applies a simplified shading function on 3D Gaussians to enhance the neural rendering in scenes with reflective surfaces while preserving the training and rendering efficiency. The main challenge in applying the shading function lies in the accurate normal estimation on discrete 3D Gaussians. Specifically, we proposed a novel normal estimation framework based on the shortest axis directions of 3D Gaussians with a delicately designed loss to make the consistency between the normals and the geometries of Gaussian spheres. Experiments show that GaussianShader strikes a commendable balance between efficiency and visual quality. Our method surpasses Gaussian Splatting [21] in PSNR on specular object datasets, exhibiting an improvement of 1.57dB. When compared to prior works handling reflective surfaces, such as Ref-NeRF [45], our optimization time is significantly accelerated (23h vs. 0.58h). Please click on our project website to see more results.*

1. Introduction

In recent years, the field of 3D computer vision has witnessed remarkable advancements in the 3D reconstruction and visualization of 3D scenes. Innovations such as Neural Radiance Fields (NeRF) [32] have achieved substantial breakthroughs in generating novel views of 3D objects and scenes, presenting the potential for high-quality and photo-realistic renderings. Despite these advancements, NeRF-related methodologies [6, 27, 45], still contend with challenges such as computationally expensive optimization and slow rendering speed. These limitations restrict their application in real-time interactive scenarios.

*Corresponding author. This work was supported by NSFC (No.62206173), MoE Key Laboratory of Intelligent Perception and Human-Machine Collaboration (ShanghaiTech University), Shanghai Frontiers Science Center of Human-centered Artificial Intelligence, the Innovation and Technology Commission of the HKSAR Government under the InnoHK initiative, and Ref. T45-205/21-N of Hong Kong RGC.

More recently, 3D Gaussian Splatting [21] combines 3D Gaussian representation and tile-based splatting techniques to achieve high-quality 3D scene modeling and real-time rendering, making it possible to employ neural rendering techniques in real applications. However, it suffers from a performance drop on scenes featuring specular and reflective surfaces. This is because 3D Gaussian Splatting [21] does not explicitly model appearance properties, so that fails to capture significant view-dependent changes, particularly specular highlights. This constraint presents a substantial obstacle in the pursuit of achieving photorealistic rendering across a diverse array of materials, particularly those characterized by prominent reflective attributes.

Accurately modeling reflective surfaces is a challenging task. Ref-NeRF [45] and ENVIDR [27] combine the shading functions in implicit representations and present promising quality on reflective surfaces. However, they suffer from time-consuming optimization (hours) and slow rendering speed. Due to the limited flexibility of SDF, ENVIDR [27] even fails to model complex scenes and presents a significant performance drop on general objects. It is still an unexplored problem how to combine the shading functions in a 3D Gaussian Splatting framework to improve its ability to handle reflections while preserving the efficiency in training and rendering.

In this paper, we present GaussianShader, a novel method that enhances the neural rendering of 3D Gaussians within scenes that contain reflective surfaces by incorporating a shading function on 3D Gaussians. To ensure the efficiency of GaussianShader, evaluating the shading function cannot be too expensive while still retaining the ability to model the reflections. In light of this, we propose a novel simplified shading function that considers the diffuse colors and the direct reflections while putting all advanced complex reflections into a residual color term. In comparison with the shading function of Ref-NeRF which can only consider direct reflections, the utilization of this residual color enables GaussianShader to render more complex reflective appearances with efficiency.

Another challenge in computing a shading function is how to predict accurate normals on the discrete 3D Gaussian spheres. First, it is hard to get a locally-continuous surface from the 3D Gaussians to compute the surface normals. Second, associating multiple 3D Gaussians for normal computation would be very expensive using neighborhood searching. In GaussianShader, we address this problem by introducing a new normal representation, which is based on the shortest axis direction of a Gaussian sphere and learns a normal residual on this axis direction. Then, to enforce consistency between the estimated normals and the geometry formulated by Gaussian spheres, we introduce an efficient normal-geometry constraint between the predicted normals and the normals derived from the rendered depths. Both the normal representation and the constraint lead to an accurate normal estimation of Gaussian spheres, which helps us compute the shading function.

Built upon 3D Gaussian Splatting, GaussianShader maintains real-time rendering speed while still accommodating various materials like reflective surfaces. Experiments show that, compared to prior works, our method keeps a good balance between efficiency and robust performance on both general scenes and reflective surfaces. In summary, our method offers several significant advantages:

1. Our method explicitly approximates the rendering equation by a simplified shading function, significantly enhancing the realism of rendered scenes, particularly for highly specular and reflective surfaces.
2. We propose a new normal estimation framework on 3D Gaussians with a new regularization loss that allows precise normal estimation.
3. Leveraging the efficiency of Gaussian Splatting, our approach provides real-time rendering capabilities, making it suitable for interactive applications and scenarios that demand efficient rendering.

2. Related Work

2.1. Neural Radiance Fields

Neural Radiance Fields (NeRF) [32] gains remarkable progress in photo-realistic novel view synthesis using implicit representation and volume rendering. Recently, NeRF has inspired many follow-up works in various directions. [5, 6] improves NeRF in rendering quality by introducing 3D conical frustum, achieving state-of-art performance in NVS. [17, 36, 47, 52, 53] combine implicit surface representations with NeRF for more accurate geometric reconstruction. [43, 44] propose solutions for city-scale scene rendering using NeRF. [41, 45, 51, 57] targets a special kind of scenes with high specularities and reflections. Another important line of works [10, 14, 15, 28, 33, 42] focuses on acceleration due to the low training and rendering speed of NeRF using voxel grid or hash table. Although great progress has been made, NeRF-based methods still suffer from low rendering speed and high training-time memory usage due to their implicit nature [11]. Based on works on point-based neural rendering [2, 39, 50, 55], a recent milestone work [21] introduces anisotropic 3D Gaussian as an effective representation of the scene, and renders the image using a fast tile-based differentiable rasterizer, surpassing existing implicit neural representation methods in both quality and efficiency.

2.2. Reflective Object Rendering

Rendering views of reflective objects from multi-view images has been a challenging task due to the complex light interactions. Previous approaches rely on simple light field

Figure 2. GaussianShader initiates with the neural 3D Gaussian spheres that integrate both conventional attributes and the newly introduced shading attributes to accurately capture view-dependent appearances. We incorporate a differentiable environment lighting map to simulate realistic lighting. The end-to-end training leads to a model that reconstructs both reflective and diffuse surfaces, achieving high material and lighting fidelity.

interpolations to achieve high-fidelity rendering of novel perspectives [16, 25, 49], yet were constrained by the necessity for dense discrete captures. The accurate rendering of reflective surfaces hinges upon the precise estimation of scene illumination (e.g. environment light) and material properties (e.g. BRDF), which is the task of inverse rendering [4, 35]. Previous studies have demonstrated methodologies for predicting BRDF under known lighting conditions [1, 13, 31], as well as techniques for estimating lighting given known geometry [24, 37, 38]. Furthermore, [12, 19, 51, 58] incorporate indirect illumination, thereby enhancing the fidelity of the estimated BRDF.

Some NeRF related works attempt to model reflectance by disentangling the visual appearance into lighting and material properties, such as [7–9, 41, 54, 57], which can jointly predict environmental illumination and surface reflectance properties under unknown or varying lighting conditions. Ref-NeRF [45] introduces a new parameterization and structuring of view-dependent outgoing radiance, as well as a regularizer on normal vectors. Recent works [26, 27, 29] utilize SDF-based representation to learn geometry from high specular surfaces, obtaining more accurate normals for physically based rendering. However, these method suffer from extremely time-consuming optimization and slow rendering speed, which hinders their employments in real applications.

2.3. Preliminaries

2.3.1 3D Gaussian Splatting Rasterization

Our method builds upon Gaussian Splatting [21], which begins with a collection of images capturing a static scene, their corresponding camera parameters, and a sparse point cloud generated through Structure-from-Motion (SfM) [40]. These points construct a set of Gaussians, each defined by position (mean) $\mathbf{p}$ and a 3D covariance matrix Σ. While a direct optimization of the covariance matrix Σ might seem intuitive, it presents challenges due to the requirement of positive semi-definiteness. As an alternative, in Gaussian Splatting, a more intuitive yet equally expressive representation is adopted as an ellipsoid for optimization. The ellipsoid is separated into scaling and rotation components, represented by a scaling matrix S and a rotation matrix R, i.e., $\Sigma = RSS^TR^T$.

2.3.2 Rendering with 3D Gaussians

By projecting to 2D frame space according to the camera parameters, 3D Gaussian spheres are both differentiable and amenable to rapid rendering through 2D splatting with α-blending. For radiance field modeling, the directional appearance aspect (color) $\mathbf{c}$ is conveyed through spherical harmonics (SH). By tile-based rasterization, we could sum up the pixel color $\mathbf{C}$ after sorting $\mathbf{c}_i$ by depth:

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

where α_i is obtained by multiplying Gaussian weight with opacity α associated to Gaussian sphere. After rasterization, the color loss could be applied to the rendered image:

$$\mathcal{L}_{\text{color}} = ||\mathbf{C} - \mathbf{C}_{\text{gt}}||^2 \tag{2}$$

3. Method

The overview of our method is depicted in Fig. 2. Our approach begins by adopting 3D Gaussian spheres, encompassing shape attributes including covariance Σ, opacity α, and position $\mathbf{p}$. To enhance the representation ability on reflections, we compute the appearances of these Gaussian spheres with a shading function, which requires a set of shading attributes, including diffuse color, roughness, specular tint, normal, and residual color, as detailed in Sec. 3.1. Subsequently, we employ a differentiable environment light map to model the direct lighting, as elaborated in Sec. 3.2. The shading process relies significantly on accurate normal estimation, as discussed in Sec. 3.3. Finally, we introduce the losses used in the whole training process in Sec. 3.4.

3.1. Shading on 3D Gaussians

Gaussian Splatting [21] models the appearances of Gaussians with simple spherical harmonic functions without considering the light-surface interactions. Thus, Gaussian Splatting fails to accurately represent strong specular surfaces. However, accurately considering the light-surface interactions requires an exact evaluation of the Rendering Equation [20], which requires extensive computational time and complex BRDF parameters. We adopt a simplified approximation of the rendering equation which enables us to achieve high-quality rendering results on reflective surfaces in a considerably shorter time.

Specifically, for a Gaussian sphere, its rendered color $\mathbf{c}$ for the viewing direction ω_o is computed by

$$\mathbf{c}(\omega_o) = \gamma(\mathbf{c}_d + \mathbf{s} \odot L_s(\omega_o, \mathbf{n}, \rho) + \mathbf{c}_r(\omega_o)), \quad (3)$$

where γ is a gamma tone mapping function [3], $\mathbf{c}_d \in [0,1]^3$ is the diffuse color of this Gaussian sphere, $\mathbf{s} \in [0,1]^3$ is the specular tint defined on this sphere, $L_s(\omega_o, \mathbf{n}, \rho)$ is the direct specular light for this sphere in this direction, $\mathbf{n}$ is the normal of this Gaussian sphere, $\rho \in [0,1]$ is the roughness of the sphere, $\mathbf{c}_r : \mathbb{R}^3 \to \mathbb{R}^3$ is so-called residual colors, and $\odot$ is the element-wise multiplication.

Explanations on Eq. 3. We explain our motivation of this shading model in the following three aspects. **a**) Diffuse color $\mathbf{c}_d$ represents the consistent colors of this Gaussian sphere, which do not change with viewing directions. **b**) $\mathbf{s} \odot L_s(\omega_o, \mathbf{n}, \rho)$ describes the interactions between the surface intrinsic color $\mathbf{s}$ and the direct specular light L_s. This term enables us to represent most of the reflections in rendering. **c**) Since there are still some reflections that cannot be explained by the above reflections of direct lights, such as scattering and reflection on indirect lights, we add a residual color term $\mathbf{c}_r(\omega_o)$ to account for these complex reflections. In comparison, Ref-NeRF [45] adopts a similar shading function without such a residual color term, which makes it struggle to handle advanced complex reflections. $\mathbf{c}_r(\omega_o)$ is parameterized by spherical harmonic functions. **d**) $\mathbf{c}_d$, $\mathbf{s}$, ρ, and the coefficients of the spherical harmonics in $\mathbf{c}_r(\omega_o)$ all are trainable parameters associated with this Gaussian sphere. In the following, we will introduce how to compute $L_s(\omega_o, \mathbf{n}, \rho)$, and $\mathbf{n}$.

3.2. Specular Light

We compute the specular light L_s by integrating incoming radiance with the specular GGX [46] Normal Distribution Function D visualized in Fig. 3

$$L_s(\omega_o, \mathbf{n}, \rho) = \int_\Omega L(\omega_i) D(\mathbf{r}, \rho)(\omega_i \cdot \mathbf{n}) d\omega_i, \quad (4)$$

where Ω represents the whole upper semi-sphere, ω_i is the direction for the input radiance and D characterizes the specular lobe (effective integral range). When the surface is rough, the specular lobe will be larger around the reflective direction $\mathbf{r}$ while if the surface is smooth, the specular lobe will be smaller. The reflection direction $\mathbf{r}$ is calculated by view direction ω_o and normal $\mathbf{n}$ using $\mathbf{r} = 2(\omega_o \cdot \mathbf{n})\mathbf{n} - \omega_o$ In our approach, environment light $L(\omega_i)$ is represented by a trainable $6 \times 64 \times 64$ cube map. All pixel values in our lighting maps are uniformly initialized to 0.8 (within a range of 0 to 1). Note that the lighting map is global, with each scene being optimized using a single lighting map.

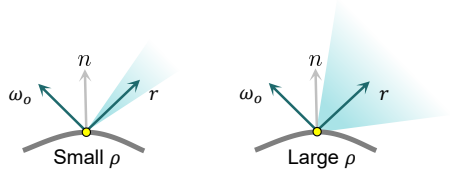

Figure 3. Normal Distribution Function D in Eq. 4 is determined by the roughness ρ and reflective direction $\mathbf{r}$. Surface with small ρ has a smaller specular lobe and that with large ρ has a larger specular lobe.

The light modeling is based on NVDiffRec [34]. Specifically, we begin by preprocessing the environment lighting map into multiple mipmap levels, each storing the integrated light across different reflective directions and roughness. When calculating light integration, we interpolate between these mipmaps based on the input of specific reflection directions and roughness, to obtain the required light integrals. This mipmap-based framework, in contrast to Ref-NeRF's [45] approach using integrated directional encoding, enhances training efficiency. The reflection direction r is calculated by combining the observation direction ω_o with the normal n.

3.3. Normal Estimation

Normal estimation on Gaussian spheres is difficult. Gaussian spheres are a collection of discrete entities, each representing a localized point in space without a continuous surface or defined edge. This discrete structure makes it inherently difficult to directly calculate normals, which typically requires a continuous surface. In the following, we observed that the shortest axis direction of a Gaussian can serve as an approximated normal and we further associate a predicted normal residual on it.

Shortest axis direction. In experiments, we made an interesting observation that the aspect ratio of the 3D Gaussian sphere—specifically, the ratios of their longest, intermediate, and shortest axis—gradually increased during optimization, as shown in Fig. 4. This observation suggests that Gaussian spheres gradually become more flattened and approach a planar shape. This observation inspires us to select the shortest axis as the normal of this "planarized" Gaussian sphere, denoted by $\mathbf{v}$.

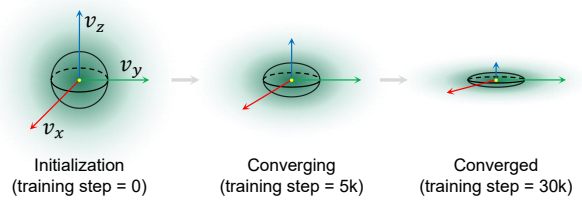

Figure 4. The geometric evolving process of a 3D Gaussian sphere in the optimization, which gradually becomes planar.

Predicted normal residual. The shortest axis $\mathbf{v}$ only serves as an approximated normal. To make the normal computation more accurate, we further introduce a trainable normal residual $\Delta\mathbf{n}$ on every Gaussian sphere. However, the orientation of the shortest axis $\mathbf{v}$ has an ambiguity because the direction of the shortest axis could either point outward or inward from the surface. To handle this ambiguity, we optimize two separate normal residuals to accommodate both scenarios. Given a specific viewing direction ω_o, we first select the direction aligned with the viewing direction $\omega_\mathbf{o}$ as the active normal direction for this viewing direction and then apply the corresponding normal residual to the active normal. This process is described by

$$\mathbf{n} = \begin{cases} \mathbf{v} + \Delta\mathbf{n}_1 & \text{if } \omega_\mathbf{o} \cdot \mathbf{v} > 0, \\ -(\mathbf{v} + \Delta\mathbf{n}_2) & \text{otherwise.} \end{cases} \tag{5}$$

To prevent the normal residual from deviating too much from the shortest axis, we add a penalty towards normal residual, making sure it is small enough.

$$\mathcal{L}_{\text{reg}} = ||\Delta\mathbf{n}||^2 \tag{6}$$

The normal residual is shown on the left of Fig. 5.

Normal-geometry consistency. The above shortest axis direction and normal residuals are defined on each Gaussian sphere separately. However, a noticeable problem is that a normal reveals the gradient of the local geometry, which is supposed to be associated with all the Gaussian spheres in a local region. We find that simply applying a color loss to train the aforementioned normal residuals leads to inconsistency between the local geometry and the estimated normals. The main reason is that every Gaussian sphere learns its normal residuals separately without knowing the local geometry formulated by its neighbor Gaussian spheres. Thus, we have to correlate multiple Gaussian spheres in a local region with their normals to ensure normal-geometry consistency. A straightforward and naive solution is to search for its K neighborhoods in space and estimate a coarse normal from all the neighboring spheres. However, such KNN searching would be extremely expensive during training because all the Gaussian spheres are dynamically moving in the optimization process. Instead, we propose a simple yet effective way to ensure normal-geometry consistency as follows.

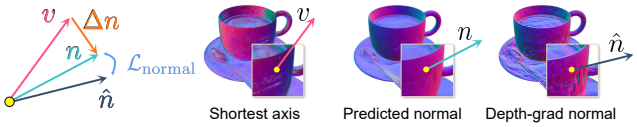

Figure 5. Visualization of the relationship between shortes axis $\mathbf{v}$, normal residual $\Delta\mathbf{n}$, normal $\mathbf{n}$ and depth-grad normal $\hat{\mathbf{n}}$. The supervision $\mathcal{L}_{normal}$ enforces the normal-geometry consistency.

We associate the local geometry with predicted normals by minimizing the difference between the grad normals derived from the rendered depth map and the rendered normal maps using the predicted normals

$$\mathcal{L}_{\text{normal}} = ||\bar{\mathbf{n}} - \hat{\mathbf{n}}||^2, \tag{7}$$

where $\bar{\mathbf{n}}$ is the rendered normal map and $\hat{\mathbf{n}}$ is computed by applying the Sobel-like operator on the rendered depth map. $\hat{\mathbf{n}}$ reveals the local geometry formulated by multiple Gaussian spheres because it is computed from the rendered depth maps. $\bar{\mathbf{n}}$ contains the information from the separately-defined normal on each Gaussian sphere. By minimizing the difference, we enforce the consistency between the local geometry and the estimated normals.

3.4. Losses

In addition to $\mathcal{L}_{\text{color}}$, $\mathcal{L}_{reg}$, and $\mathcal{L}_{\text{normal}}$, we make use of sparse loss [30, 50] to encourages Gaussian spheres' opacity values α to approach either 0 or 1 by

$$\mathcal{L}_{\text{sparse}} = \frac{1}{|\alpha|} \sum_{\alpha_i} \left[\log(\alpha_i) + \log(1 - \alpha_i)\right]. \tag{8}$$

This sparsity loss helps the geometry of Gaussian spheres converge to a single thin plate and improves the rendering quality. In summary, the total training loss $\mathcal{L}$ is

$$\mathcal{L} = \mathcal{L}_{\text{color}} + \lambda_{\text{n}}\mathcal{L}_{\text{normal}} + \lambda_{\text{s}}\mathcal{L}_{\text{sparse}} + \lambda_{\text{r}}\mathcal{L}_{\text{reg}} \tag{9}$$

where $\lambda_{\text{n}} = 0.01, \lambda_{\text{s}} = 0.001, \lambda_{\text{r}} = 0.001$.

4. Experiments

4.1. Datasets

To comprehensively validate the effectiveness of GaussianShader, we conduct evaluation on various datasets: **a)** widely-used NVS dataset: NeRF Synthetic [32]. **b)** reflective objects datasets: Shiny Blender [45] and Glossy Synthetic [29]. **c)** real-world large-scale scenes: Tanks and Temples [23].

4.2. Baselines and Metrics

We compare our method against the following baselines: **a)** 3D Gaussian Splatting [21]: a real-time radiance field rendering method based on efficient 3D Gaussian representation; **b)** VolSDF [52]: a classical neural implicit surface

Table 1. The quantitative comparisons (PSNR / SSIM / LPIPS) on NeRF Synthetic dataset [32].

	NeRF Synthetic [32]								
	Chair	Drums	Lego	Mic	Materials	Ship	Hotdog	Ficus	Avg.
					PSNR↑				
NeRF [32]	33.00	25.01	32.54	32.91	29.62	28.65	36.18	30.13	31.01
VolSDF [52]	30.57	20.43	29.46	30.53	29.13	25.51	35.11	22.91	27.96
Ref-NeRF [45]	33.98	25.43	35.10	33.65	27.10	29.24	37.04	28.74	31.29
ENVIDR [27]	31.22	22.99	29.55	32.17	29.52	21.57	31.44	26.60	28.13
Gaussian Splatting [21]	35.82	26.17	35.69	35.34	30.00	30.87	37.67	34.83	33.30
Ours	35.83	26.36	35.87	35.23	30.07	30.82	37.85	34.97	33.38
					SSIM↑				
NeRF [32]	0.967	0.925	0.961	0.980	0.949	0.856	0.974	0.964	0.947
VolSDF [52]	0.949	0.893	0.951	0.969	0.954	0.842	0.972	0.929	0.932
Ref-NeRF [45]	0.974	0.929	0.975	0.983	0.921	0.864	0.979	0.954	0.947
ENVIDR [27]	0.976	0.930	0.961	0.984	0.968	0.855	0.963	0.987	0.956
Gaussian Splatting [21]	0.987	0.954	0.983	0.991	0.960	0.907	0.985	0.987	0.969
Ours	0.987	0.949	0.983	0.991	0.960	0.905	0.985	0.985	0.968
					LPIPS↓				
NeRF [32]	0.046	0.091	0.050	0.028	0.063	0.206	0.121	0.044	0.081
VolSDF [52]	0.056	0.119	0.054	0.191	0.048	0.191	0.043	0.068	0.096
Ref-NeRF [45]	0.029	0.073	0.025	0.018	0.078	0.158	0.028	0.056	0.058
ENVIDR [27]	0.031	0.080	0.054	0.021	0.045	0.228	0.072	0.010	0.067
Gaussian Splatting [21]	0.012	0.037	0.016	0.006	0.034	0.106	0.020	0.012	0.030
Ours	0.012	0.040	0.014	0.006	0.033	0.098	0.019	0.013	0.029

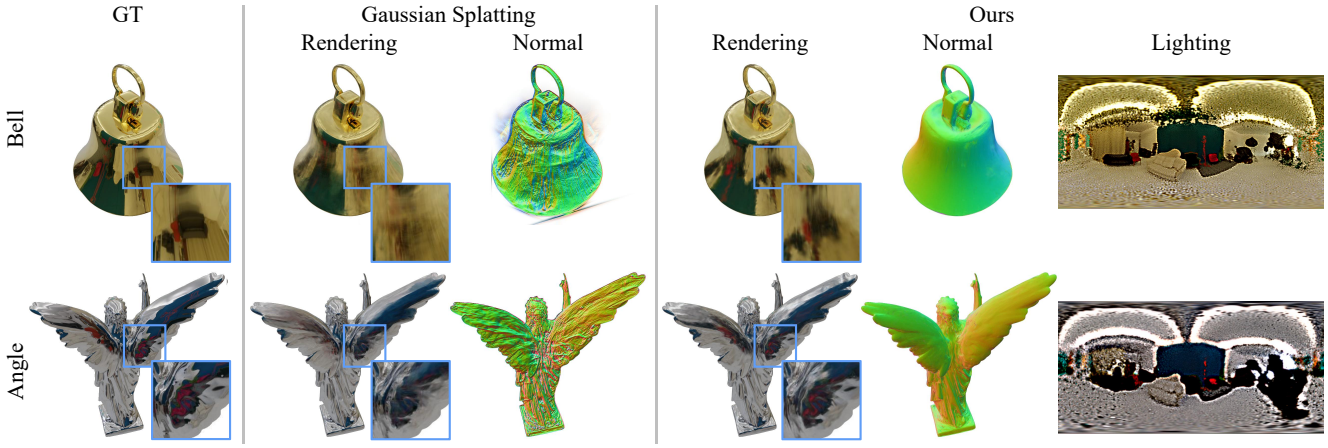

Figure 6. The qualitative comparisons with 3D Gaussian Splatting [21] on Glossy dataset [29]. Our method not only renders the objects with high fidelity but also provides detailed normal and lighting maps. Some areas are zoomed in for better visualization.

reconstruction method based on SDF; **c)** Ref-NeRF [45]: state-of-the-art method in novel view synthesis of reflective objects; **d)** NVDiffRec [34], NVDiffRecMC [18]: top-performing neural inverse rendering methods; **e)** ENVIDR [27], NeRO [29]: top-performing SDF-based neural implicit methods for reconstructing reflective objects. The evaluation metrics to measure rendering quality are reported in PSNR, SSIM [48], and LPIPS [56].

4.3. Implementation Details

All experiments are conducted on a Nvidia RTX 3090 graphics card. We optimize our models using Adam [22] optimizer for 30,000 iterations. For Ref-NeRF [45], ENVIDR [27], and 3D Gaussian Splatting [21], we retrain them using their official codes keeping all the original settings (training iterations, sampled rays, etc.) and obtain results from the retrained models. Other works' results are imported from their original papers.

4.4. Comparisons

NeRF Synthetic dataset [32]. We first evaluate our model on the NeRF Synthetic dataset, which contains objects with complex geometry and realistic non-Lambertian materials. We show the quantitative results in Tab. 1 and visual comparisons in Fig. 8. Our approach achieves numerically and visually comparable results with both Gaussian Splatting and neural rendering methods, demonstrating the effectiveness of our method in rendering general objects.

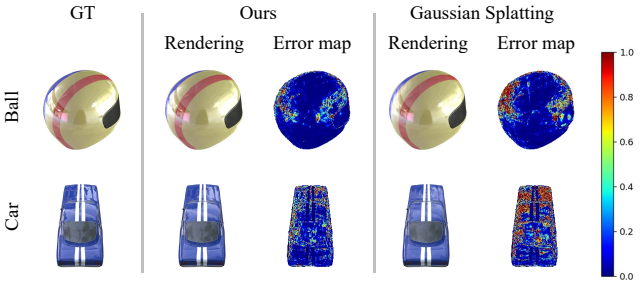

Figure 7. Qualitative comparisons on Shiny Blender dataset [45]. Error maps corresponding to each rendering result demonstrate the enhanced fidelity of our approach in highly specular areas.

Note that SDF-based ENVIDR [27] cannot perform well on shadowed regions of complex objects, as shown in Fig. 8.

Shiny Blender dataset [45]. Quantitative results on the Shiny Blender dataset are reported in Tab. 2. With the ability to model light-surface interactions, our method outperforms original Gaussian Splatting in all scenes while slightly underperforms ENVIDR and Ref-NeRF. ENVIDR capitalizes on the continuous properties of Signed Distance Fields to naturally create smoother surfaces and normals, resulting in high rendering quality, particularly for reflective objects. Fig. 7 shows our comparisons with Gaussian Splatting. We can see that our method correctly renders surfaces with strong specular appearances. High-quality modeling of reflections is contingent upon the accurate estimation of normals as shown in Fig. 9.

Additionally, we report the average PSNR, training time, and rendering FPS in Tab. 3. The reported results are averaged among all objects of the Shiny Blender and NeRF synthetic datasets. On the same hardware, our method only takes about 0.5 hour for training while MLP-based methods, like ENVIDR and Ref-NeRF, require 6 hours and 23 hours for optimizing. Due to the introduce of extra lighting and shading attributes, our method is a bit slower than 3D Gaussian Splatting [21] but still keeps reasonable efficiency and real-time rendering speed.

Tanks and Temples dataset [23]. To explore our method's scalability in larger-scale environments instead of only small objects, we conducted experiments using Tanks and Temples [23]. The qualitative results in Fig. 10 show that our model outperforms Gaussian Splatting [21].

Glossy Synthetic dataset [29]. We present a comparative visualization between Gaussian Splatting and our method on the Glossy dataset shown in Fig. 6. Our method not only renders objects with exceptional fidelity but also provides more accurate normal and lightings, capturing the nuances of reflective surfaces with greater precision. These maps reveal our method's ability to approximate the rendering function, resulting in a more lifelike portrayal of specular lighting. Please refer to the supplementary materials provided for an extensive showcase of our results.

Table 2. Qualitative comparisons on Shiny Blender dataset [45]. Our method is comparable with both Gaussian Splatting [21] and prior reflective object reconstruction methods.

	Shiny Blender [45]						
	Car	Ball	Helmet	Teapot	Toaster	Coffee	Avg.
	PSNR↑						
NVDiffRec [34]	27.98	21.77	26.97	40.44	24.31	30.74	28.70
NVDiffMC [18]	25.93	30.85	26.27	38.44	22.18	29.60	28.88
Ref-NeRF [45]	30.41	29.14	29.92	45.19	25.29	33.99	32.32
NeRO [29]	25.53	30.26	29.20	38.70	26.46	28.89	29.84
ENVIDR [27]	28.46	38.89	32.73	41.59	26.11	29.48	32.88
Gaussian Splatting [21]	27.24	27.69	28.32	45.68	20.99	32.32	30.37
Ours	27.90	30.98	28.32	45.86	26.21	32.39	31.94
	SSIM↑						
NVDiffRec [34]	0.963	0.858	0.951	0.996	0.928	0.973	0.945
NVDiffMC [18]	0.940	0.940	0.940	0.995	0.886	0.965	0.944
Ref-NeRF [45]	0.949	0.956	0.955	0.995	0.910	0.972	0.956
NeRO [29]	0.949	0.974	0.971	0.995	0.929	0.956	0.962
ENVIDR [27]	0.961	0.991	0.980	0.996	0.939	0.949	0.969
Gaussian Splatting [21]	0.930	0.937	0.951	0.996	0.895	0.971	0.947
Ours	0.931	0.965	0.950	0.996	0.929	0.971	0.957
	LPIPS↓						
NVDiffRec [34]	0.045	0.297	0.118	0.011	0.169	0.076	0.119
NVDiffMC [18]	0.077	0.312	0.157	0.014	0.225	0.097	0.147
Ref-NeRF [45]	0.051	0.307	0.087	0.013	0.118	0.082	0.109
NeRO [29]	0.074	0.094	0.050	0.012	0.089	0.110	0.072
ENVIDR [27]	0.049	0.067	0.051	0.011	0.116	0.139	0.072
Gaussian Splatting [21]	0.047	0.161	0.079	0.007	0.126	0.078	0.083
Ours	0.045	0.121	0.076	0.007	0.079	0.078	0.068

Table 3. Comparisons over average PSNR, training time and FPS. Our method achieves the best rendering quality and competitive training/rendering speed.

	PSNR	Training Time	FPS
Ref-NeRF [45]	31.73	23h	0.03
ENVIDR [27]	30.04	6h	1.33
Gaussian Splatting [21]	32.05	**0.25h**	**274**
Ours	**32.76**	0.58h	97

4.5. Ablation Study

In this section, we conduct ablation studies of our model on Shiny Blender dataset. The numerical results are reported in Tab. 4. All ablation experiments are done on Shiny Blender dataset with $1/2$ image resolution.

Loss functions. We trained our model serperately under the setting "without sparse loss $\mathcal{L}_{\text{sparse}}$" and "without normal loss $\mathcal{L}_{\text{normal}}$". The incorporation of these losses, especially the direct regularization on normals, proved critical, yielding a marked improvement in rendering quality by effectively guiding the normal optimization process as shown in Tab. 4.

Residual color. Residual color, comprising 3rd order SH coefficients, is used to compensate for indirect light. As shown in Table 4, the introduce of residual color enhances our model, allowing it to simulate a wide array of real-world intricate lighting situations and significantly improve rendering quality.

Normal formulation. In our full model, we use the shortest axis direction $\mathbf{v}$ with a learnable residual $\Delta\mathbf{n}$ to formulate normal, as described by Eqn. 3.3. In Tab. 4, we compared this novel normal formation with a naive normal prediction manner (indicated by "w/o axis $\mathbf{v}$"). Instead of

Figure 8. We present a qualitative comparisons of our method on NeRF Synthetic dataset [32] against previous techniques and corresponding ground truth images from test views. For clarity, we zoom in the images where differences in quality are especially notable.

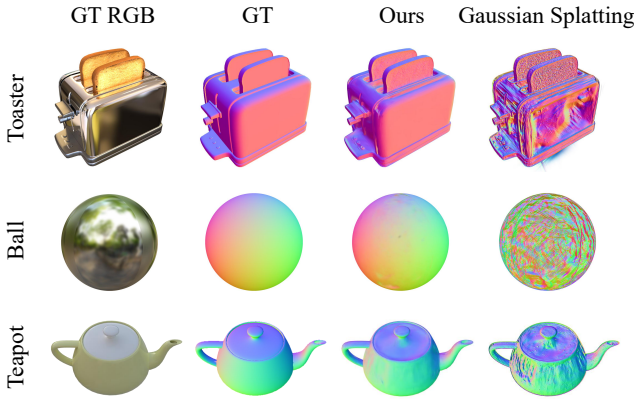

Figure 9. Visualized normal results on Shiny Blender dataset [45], compared with Gaussian Splatting [21]. It illustrates the superior normal estimation achieved by our method.

Table 4. Ablation studies on loss function, lighting representation, normal axis and residual color using Shiny Blender dataset.

	PSNR↑	SSIM↑	LPIPS↓
w/o $\mathcal{L}_{\text{sparse}}$	31.79	0.952	0.056
w/o $\mathcal{L}_{\text{normal}}$	30.93	0.941	0.060
w/o $\mathbf{c}_r$	31.49	0.948	0.060
w/o $\mathbf{v}$	31.47	0.951	0.058
MLP Lighting	29.73	0.936	0.075
Full Model	**32.09**	**0.953**	**0.054**

correlating with the geometry of 3D gaussians, we directly optimize a single normal attribute. The result shows our full model can achieve higher rendering quality, since our predicted normals are correlated with the true geometry of Gaussian spheres.

Lighting representation. A crucial component of our approach is the modeling of lighting. "MLP Lighting" means we use a directional MLP similar to Ref-NeRF to implicitly optimize lighting. In our full model, we use "Envmap Lighting" representation as illustrated in Sec. 3.2. The result proves that our explicit lighting representation is better suited for 3D Gaussian rendering framework to correctly render reflective objects.

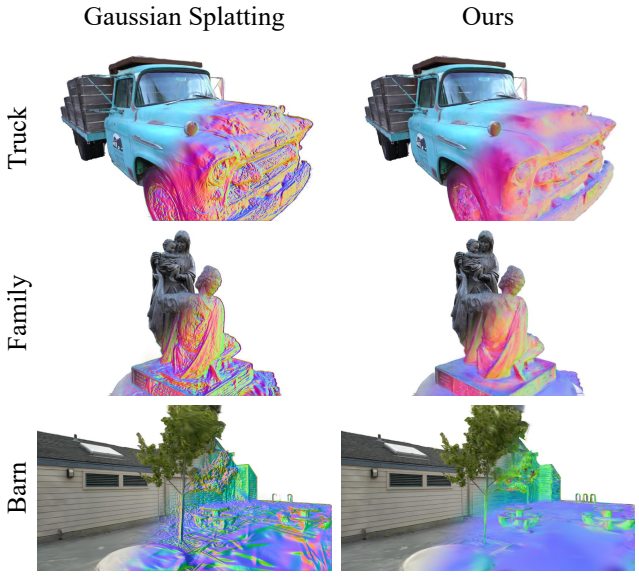

Figure 10. Qualitative comparisons with Gaussian Splatting [21] on the Tanks and Temples dataset [23]. Our approach yields a smoother and more reasonable normal.

5. Conclusions

In summary, we propose GaussianShader, advancing the rendering of both reflective and general objects through an extended 3D Gaussian model. Specifically, our method integrates shading functions with 3D Gaussian representation for handling view-dependent appearances, successfully handling reflective surfaces. Moreover, a novel normal prediction method is proposed to achieve high-quality normals, which correlates the shading parameters and true geometry of 3D Gaussians. The results show our approach achieves high-quality renderings, marking a substantial improvement in efficient, realistic 3D object rendering.

References

[1] Miika Aittala, Timo Aila, and Jaakko Lehtinen. Reflectance modeling by neural texture synthesis. *ACM Transactions on Graphics (ToG)*, 35(4):1–13, 2016. 3

[2] Kara-Ali Aliev, Artem Sevastopolsky, Maria Kolos, Dmitry Ulyanov, and Victor Lempitsky. Neural point-based graphics. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXII 16*, pages 696–712. Springer, 2020. 2

[3] Matthew Anderson, Ricardo Motta, Srinivasan Chandrasekar, and Michael Stokes. Proposal for a standard default color space for the internet—srgb. In *Color and imaging conference*, pages 238–245. Society for Imaging Science and Technology, 1996. 4

[4] Jonathan T Barron and Jitendra Malik. Shape, illumination, and reflectance from shading. *IEEE transactions on pattern analysis and machine intelligence*, 37(8):1670–1687, 2014. 3

[5] Jonathan T Barron, Ben Mildenhall, Matthew Tancik, Peter Hedman, Ricardo Martin-Brualla, and Pratul P Srinivasan. Mip-nerf: A multiscale representation for anti-aliasing neural radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5855–5864, 2021. 2

[6] Jonathan T Barron, Ben Mildenhall, Dor Verbin, Pratul P Srinivasan, and Peter Hedman. Mip-nerf 360: Unbounded anti-aliased neural radiance fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5470–5479, 2022. 1, 2

[7] Sai Bi, Zexiang Xu, Pratul Srinivasan, Ben Mildenhall, Kalyan Sunkavalli, Miloš Hašan, Yannick Hold-Geoffroy, David Kriegman, and Ravi Ramamoorthi. Neural reflectance fields for appearance acquisition. *arXiv preprint arXiv:2008.03824*, 2020. 3

[8] Mark Boss, Raphael Braun, Varun Jampani, Jonathan T Barron, Ce Liu, and Hendrik Lensch. Nerd: Neural reflectance decomposition from image collections. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 12684–12694, 2021.

[9] Mark Boss, Varun Jampani, Raphael Braun, Ce Liu, Jonathan Barron, and Hendrik Lensch. Neural-pil: Neural pre-integrated lighting for reflectance decomposition. *Advances in Neural Information Processing Systems*, 34: 10691–10704, 2021. 3

[10] Anpei Chen, Zexiang Xu, Andreas Geiger, Jingyi Yu, and Hao Su. Tensorf: Tensorial radiance fields. In *European Conference on Computer Vision*, pages 333–350. Springer, 2022. 2

[11] Zilong Chen, Feng Wang, and Huaping Liu. Text-to-3d using gaussian splatting. *arXiv preprint arXiv:2309.16585*, 2023. 2

[12] Youming Deng, Xueting Li, Sifei Liu, and Ming-Hsuan Yang. Dip: Differentiable interreflection-aware physics-based inverse rendering. *arXiv preprint arXiv:2212.04705*, 2022. 3

[13] Valentin Deschaintre, Miika Aittala, Fredo Durand, George Drettakis, and Adrien Bousseau. Single-image svbrdf capture with a rendering-aware deep network. *ACM Transactions on Graphics (ToG)*, 37(4):1–15, 2018. 3

[14] 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

[15] Stephan J Garbin, Marek Kowalski, Matthew Johnson, Jamie Shotton, and Julien Valentin. Fastnerf: High-fidelity neural rendering at 200fps. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 14346–14355, 2021. 2

[16] Steven J Gortler, Radek Grzeszczuk, Richard Szeliski, and Michael F Cohen. The lumigraph. In *Seminal Graphics Papers: Pushing the Boundaries, Volume 2*, pages 453–464. 2023. 3

[17] Haoyu Guo, Sida Peng, Haotong Lin, Qianqian Wang, Guofeng Zhang, Hujun Bao, and Xiaowei Zhou. Neural 3d scene reconstruction with the manhattan-world assumption. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5511–5520, 2022. 2

[18] Jon Hasselgren, Nikolai Hofmann, and Jacob Munkberg. Shape, Light, and Material Decomposition from Images using Monte Carlo Rendering and Denoising. *arXiv:2206.03380*, 2022. 6, 7

[19] Jon Hasselgren, Nikolai Hofmann, and Jacob Munkberg. Shape, light, and material decomposition from images using monte carlo rendering and denoising. *Advances in Neural Information Processing Systems*, 35:22856–22869, 2022. 3

[20] James T Kajiya. The rendering equation. In *Proceedings of the 13th annual conference on Computer graphics and interactive techniques*, pages 143–150, 1986. 4

[21] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics (ToG)*, 42(4):1–14, 2023. 1, 2, 3, 4, 5, 6, 7, 8

[22] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 6

[23] Arno Knapitsch, Jaesik Park, Qian-Yi Zhou, and Vladlen Koltun. Tanks and temples: Benchmarking large-scale scene reconstruction. *ACM Transactions on Graphics (ToG)*, 36 (4):1–13, 2017. 5, 7, 8, 1

[24] Chloe LeGendre, Wan-Chun Ma, Graham Fyffe, John Flynn, Laurent Charbonnel, Jay Busch, and Paul Debevec. Deeplight: Learning illumination for unconstrained mobile mixed reality. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5918–5928, 2019. 3

[25] Marc Levoy and Pat Hanrahan. Light field rendering. In *Seminal Graphics Papers: Pushing the Boundaries, Volume 2*, pages 441–452. 2023. 3

[26] Ruofan Liang, Jiahao Zhang, Haoda Li, Chen Yang, Yushi Guan, and Nandita Vijaykumar. Spidr: Sdf-based neural point fields for illumination and deformation. *arXiv preprint arXiv:2210.08398*, 2022. 3

[27] Ruofan Liang, Huiting Chen, Chunlin Li, Fan Chen, Selvakumar Panneer, and Nandita Vijaykumar. Envidr: Implicit differentiable renderer with neural environment lighting. *arXiv preprint arXiv:2303.13022*, 2023. 1, 2, 3, 6, 7

[28] Lingjie Liu, Jiatao Gu, Kyaw Zaw Lin, Tat-Seng Chua, and Christian Theobalt. Neural sparse voxel fields. *Advances in Neural Information Processing Systems*, 33:15651–15663, 2020. 2

[29] 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. *arXiv preprint arXiv:2305.17398*, 2023. 3, 5, 6, 7, 1

[30] Stephen Lombardi, Tomas Simon, Jason Saragih, Gabriel Schwartz, Andreas Lehrmann, and Yaser Sheikh. Neural volumes: Learning dynamic renderable volumes from images. *arXiv preprint arXiv:1906.07751*, 2019. 5

[31] Wojciech Matusik. *A data-driven reflectance model*. PhD thesis, Massachusetts Institute of Technology, 2003. 3

[32] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. *Communications of the ACM*, 65(1):99–106, 2021. 1, 2, 5, 6, 8

[33] 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

[34] 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*, pages 8280–8290, 2022. 4, 6, 7

[35] Merlin Nimier-David, Delio Vicini, Tizian Zeltner, and Wenzel Jakob. Mitsuba 2: A retargetable forward and inverse renderer. *ACM Transactions on Graphics (TOG)*, 38(6):1–17, 2019. 3

[36] Michael Oechsle, Songyou Peng, and Andreas Geiger. Unisurf: Unifying neural implicit surfaces and radiance fields for multi-view reconstruction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5589–5599, 2021. 2

[37] Jeong Joon Park, Aleksander Holynski, and Steven M Seitz. Seeing the world in a bag of chips. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1417–1427, 2020. 3

[38] Thomas Richter-Trummer, Denis Kalkofen, Jinwoo Park, and Dieter Schmalstieg. Instant mixed reality lighting from casual scanning. In *2016 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, pages 27–36. IEEE, 2016. 3

[39] Darius Rückert, Linus Franke, and Marc Stamminger. Adop: Approximate differentiable one-pixel point rendering. *ACM Transactions on Graphics (ToG)*, 41(4):1–14, 2022. 2

[40] Noah Snavely, Steven M Seitz, and Richard Szeliski. Photo tourism: exploring photo collections in 3d. In *ACM siggraph 2006 papers*, pages 835–846. 2006. 3

[41] 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*, pages 7495–7504, 2021. 2, 3

[42] Cheng Sun, Min Sun, and Hwann-Tzong Chen. Direct voxel grid optimization: Super-fast convergence for radiance fields reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5459–5469, 2022. 2

[43] Matthew Tancik, Vincent Casser, Xinchen Yan, Sabeek Pradhan, Ben Mildenhall, Pratul P Srinivasan, Jonathan T Barron, and Henrik Kretzschmar. Block-nerf: Scalable large scene neural view synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8248–8258, 2022. 2

[44] Haithem Turki, Deva Ramanan, and Mahadev Satyanarayanan. Mega-nerf: Scalable construction of large-scale nerfs for virtual fly-throughs. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12922–12931, 2022. 2

[45] 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. 1, 2, 3, 4, 5, 6, 7, 8

[46] Bruce Walter, Stephen R Marschner, Hongsong Li, and Kenneth E Torrance. Microfacet models for refraction through rough surfaces. In *Proceedings of the 18th Eurographics conference on Rendering Techniques*, pages 195–206, 2007. 4

[47] 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

[48] 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. 6

[49] Daniel N Wood, Daniel I Azuma, Ken Aldinger, Brian Curless, Tom Duchamp, David H Salesin, and Werner Stuetzle. Surface light fields for 3d photography. In *Seminal Graphics Papers: Pushing the Boundaries, Volume 2*, pages 487–496. 2023. 3

[50] Qiangeng Xu, Zexiang Xu, Julien Philip, Sai Bi, Zhixin Shu, Kalyan Sunkavalli, and Ulrich Neumann. Point-nerf: Point-based neural radiance fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5438–5448, 2022. 2, 5

[51] 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

[52] Lior Yariv, Jiatao Gu, Yoni Kasten, and Yaron Lipman. Volume rendering of neural implicit surfaces. *Advances in Neu-*

ral Information Processing Systems, 34:4805–4815, 2021. 2, 5, 6

[53] Zehao Yu, Songyou Peng, Michael Niemeyer, Torsten Sattler, and Andreas Geiger. Monosdf: Exploring monocular geometric cues for neural implicit surface reconstruction. *Advances in neural information processing systems*, 35:25018–25032, 2022. 2

[54] Kai Zhang, Fujun Luan, Qianqian Wang, Kavita Bala, and Noah Snavely. Physg: Inverse rendering with spherical gaussians for physics-based material editing and relighting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5453–5462, 2021. 3

[55] Qiang Zhang, Seung-Hwan Baek, Szymon Rusinkiewicz, and Felix Heide. Differentiable point-based radiance fields for efficient view synthesis. In *SIGGRAPH Asia 2022 Conference Papers*, pages 1–12, 2022. 2

[56] 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

[57] 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 (ToG)*, 40(6):1–18, 2021. 2, 3

[58] Yuanqing Zhang, Jiaming Sun, Xingyi He, Huan Fu, Rongfei Jia, and Xiaowei Zhou. Modeling indirect illumination for inverse rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18643–18652, 2022. 3