SpectroMotion: Dynamic 3D Reconstruction of Specular Scenes

Cheng-De Fan[1] Chen-Wei Chang[1] Yi-Ruei Liu[1,2] Jie-Ying Lee[1]
Jiun-Long Huang[1] Yu-Chee Tseng[1] Yu-Lun Liu[1]

[1]National Yang Ming Chiao Tung University [2]University of Illinois Urbana-Champaign

Abstract

We present SpectroMotion, a novel approach that combines 3D Gaussian Splatting (3DGS) with physically-based rendering (PBR) and deformation fields to reconstruct dynamic specular scenes. Previous methods extending 3DGS to model dynamic scenes have struggled to represent specular surfaces accurately. Our method addresses this limitation by introducing a residual correction technique for accurate surface normal computation during deformation, complemented by a deformable environment map that adapts to time-varying lighting conditions. We implement a coarse-to-fine training strategy significantly enhancing scene geometry and specular color prediction. It is the only existing 3DGS method capable of synthesizing photorealistic real-world dynamic specular scenes, outperforming state-of-the-art methods in rendering complex, dynamic, and specular scenes. Please see our project page at cd-fan0627.github.io/spectromotion.

1. Introduction

3D Gaussian Splatting (3DGS) [16] has emerged as a groundbreaking technique in 3D scene reconstruction, offering fast training and real-time rendering capabilities. By representing 3D scene using a collection of 3D Gaussians and employing a point-based rendering approach, 3DGS has significantly improved efficiency in novel view synthesis. However, extending 3DGS to model dynamic scenes, especially those containing specular surfaces accurately, has remained a significant challenge.

Existing extensions of 3DGS have made progress in either dynamic scene reconstruction or specular object rendering, but none have successfully combined both aspects. Methods tackling dynamic scenes often struggle with accurately representing specular surfaces, while those focusing on specular rendering are limited to static scenes. This capability gap has hindered the application of 3DGS to real-world scenarios where both motion and specular reflections are present.

We present SpectroMotion, a novel approach that addresses these limitations by combining 3D Gaussian Splatting with physically based rendering (PBR) and deformation fields. Our method introduces three key innovations: a residual correction technique for accurate surface normal computation during deformation, a deformable environment map that adapts to time-varying lighting conditions, and a coarse-to-fine training strategy that significantly enhances scene geometry and specular color prediction.

Our evaluations demonstrate that SpectroMotion outperforms prior methods in view synthesis of scenes containing dynamic specular objects, as illustrated in Fig. 1. It is the only 3DGS method capable of synthesizing photorealistic real-world dynamic specular scenes, surpassing state-of-the-

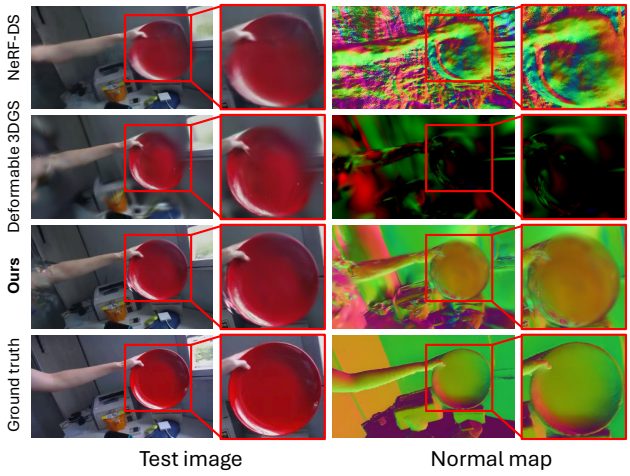

Figure 1. **Our method, SpectroMotion, recovers and renders dynamic scenes with higher-quality reflections compared to prior work.** It introduces physical normal estimation, deformable environment maps, and a coarse-to-fine training strategy to achieve superior results in rendering dynamic scenes with reflections. Here, we present a rendered test image, corresponding normal maps, and a ground-truth image, where the ground-truth normal map (used as a reference) is generated using a pre-trained normal estimator [6]. For Deformable 3DGS, we use the shortest axes of the deformed 3D Gaussians as the normals. We have highlighted the specular regions to demonstrate the effectiveness of our approach.

art techniques in rendering complex, dynamic, and specular content. This advancement represents a significant leap in 3D scene reconstruction, particularly for challenging scenarios involving moving specular objects.

In summary, we make the following contributions:

- We propose SpectroMotion, a physically-based rendering (PBR) approach combining deformation fields and 3D Gaussian Splatting for real-world dynamic specular scenes.
- We introduce a residual correction method for accurate surface normals during deformation, coupled with a deformable environment map to handle time-varying lighting conditions in dynamic scenes.
- We develop a coarse-to-fine training strategy enhancing scene geometry and specular color prediction, outperforming state-of-the-art methods.

2. Related Work

2.1. Dynamic Scene Reconstruction

Recent works have leveraged NeRF representations to jointly solve for canonical space and deformation fields in dynamic scenes using RGB supervision [4, 9, 19, 26, 27, 31, 32, 34, 39, 45, 47]. Further advancements in dynamic neural rendering include object segmentation [36], incorporation of depth information [1], utilization of 2D CNNs for scene priors [24, 33], and multi-view video compression [18]. However, these NeRF-based methods are computationally intensive, limiting their practical applications. To address this, 3D Gaussian Splatting [16] has emerged as a promising alternative, offering real-time rendering capabilities while maintaining high visual quality. Building upon this efficient representation, recent research has adapted 3D Gaussians for dynamic scenes [11, 22, 29, 38, 44, 46, 49]. Nevertheless, these approaches do not explicitly account for changes in surface normal during the dynamic process. Our work extends this line of research by combining specular object rendering based on normal estimation with a deformation field, enabling each 3D Gaussian to model dynamic specular scenes effectively.

2.2. Reflective Object Rendering

While significant progress has been made in rendering reflective objects, challenges from complex light interactions persist. Recent years have seen numerous studies addressing these issues, primarily by decomposing appearance into lighting and material properties [2, 3, 17, 30, 37, 41, 54–56]. Building on this foundation, some research has focused on improving the capture and reproduction of specular reflections [28, 40, 42, 51]. In contrast, others have leveraged signed distance functions (SDFs) to enhance normal estimation [8, 20, 21, 25, 53]. The emergence of 3D Gaussian Splatting (3DGS) has sparked a new wave of techniques [7, 12, 23, 35, 50, 57] that integrate Gaussian splatting with physically-based rendering. Nevertheless, accurately modeling dynamic environments and time-varying specular reflections remains a significant challenge. To address this limitation, our work introduces a novel approach incorporating a deformable environment map and additional explicit Gaussian attributes specifically designed to capture specular color changes over time.

3. Method

Overview of the approach. The overview of our method is illustrated in Fig. 2. Given an input monocular video sequence of frames and corresponding camera poses, we design a three-stage approach to reconstruct the dynamic specular scene, as detailed in Sec. 3.2. Accurate specular reflection requires precise normal estimation, so Sec. 3.3 elaborates on how we estimate normals in dynamic scenes. Finally, we introduce the losses used throughout the training process in Sec. 3.4.

3.1. Preliminary

3D Gaussian Splatting. Each 3D Gaussian is defined by a center position $\boldsymbol{x} \in \mathbb{R}^3$ and a covariance matrix $\boldsymbol{\Sigma}$. 3D Gaussian Splatting [16] optimizes the covariance matrix using scaling factors $\boldsymbol{s} \in \mathbb{R}^3$ and rotation unit quaternion $\boldsymbol{r} \in \mathbb{R}^4$. For novel-view rendering, 3D Gaussians are projected onto 2D camera planes using differentiable splatting [52]:

$$\boldsymbol{\Sigma}'=\mathbf{JW\Sigma W}^T\mathbf{J}^T. \tag{1}$$

Pixel colors are computed using point-based volumetric rendering:

$$C = \sum_{i \in N} T_i \alpha_i c_i, \quad \alpha_i = \sigma_i e^{-\frac{1}{2}(\boldsymbol{x})^T \boldsymbol{\Sigma}'(\boldsymbol{x})}, \tag{2}$$

where $T_i = \prod_{j=1}^{i-1}(1-\alpha_j)$ is the transmittance, σ_i is the opacity, and $\mathbf{c}_i$ is the color of each 3D Gaussian.

3.2. Specular Rendering

Since accurate reflections depend heavily on precise geometry, we implement a three-stage coarse-to-fine training strategy: static, dynamic, and specular stages. This approach ensures both stable scene geometry and accurate specular rendering.

3.2.1. Coarse-to-Fine Training Strategy

Static stage. In the static stage, we employ vanilla 3DGS [16] for static scene reconstruction to stabilize the geometry of the static scene. Specifically, we optimize the position $\boldsymbol{x}$, scaling $\boldsymbol{s}$, rotation $\boldsymbol{r}$, opacity α, and coefficients of spherical harmonics (SH) of the 3D Gaussians by minimizing the photometric loss $\mathcal{L}_{\text{color}}$ identical to 3DGS [16].

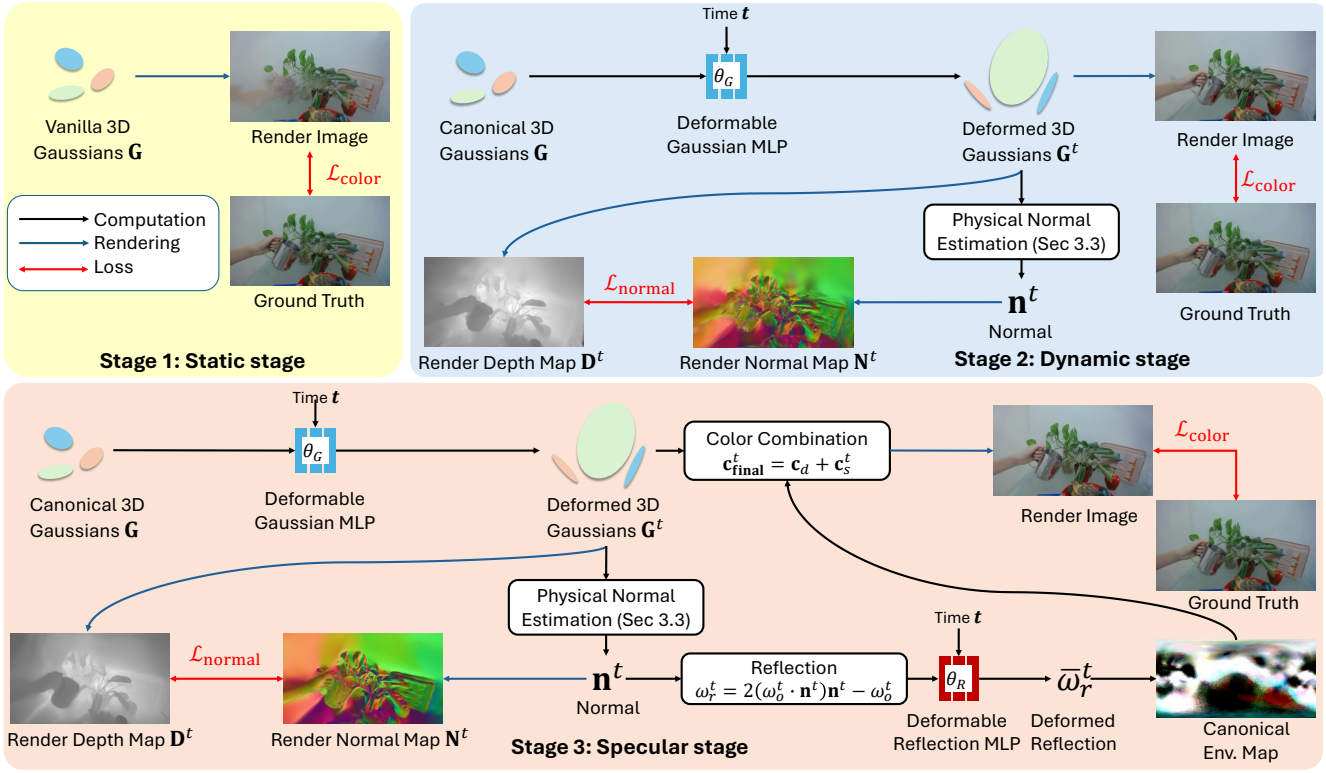

Figure 2. **Method Overview.** Our method stabilizes the scene geometry through three stages. In the static stage, we stabilize the geometry of the static scene by minimizing photometric loss $\mathcal{L}_{\text{color}}$ between vanilla 3DGS renders and ground truth images. The dynamic stage combines canonical 3D Gaussians $\mathbf{G}$ with a deformable Gaussian MLP to model dynamic scenes while simultaneously minimizing normal loss $\mathcal{L}_{\text{normal}}$ between rendered normal map $\mathbf{N}^t$ and gradient normal map from depth map $\mathbf{D}^t$, thus further enhancing the overall scene geometry. Finally, the specular stage introduces a deformable reflection MLP to handle changing environment lighting, deforming reflection directions ω^t_r to query a canonical environment map for specular color $\mathbf{c}^t_s$. It is then combined with diffuse color $\mathbf{c_d}$ (using zero-order spherical harmonics) and learnable specular tint $\mathbf{s_{tint}}$ per 3D Gaussian to obtain the final color $\mathbf{c^t_{final}}$. This approach enables the modeling of dynamic specular scenes and high-quality novel view rendering.

Dynamic stage. Following the static stage, we address dynamic objects using Deformable 3DGS [49]. For each 3D Gaussian in canonical 3D Gaussians $\mathbf{G}$, we input its position $\boldsymbol{x}$ and time t into a deformable Gaussian MLP with parameters θ_G to predict position, rotation, and scaling residuals: $(\Delta\boldsymbol{x}^t, \Delta\boldsymbol{r}^t, \Delta\boldsymbol{s}^t) = F_{\theta_G}(\gamma(x), \gamma(t))$, where γ denotes positional encoding. Attributes of the corresponding 3D Gaussian in deformed 3D Gaussians $\mathbf{G}^t$ at time t is obtained by $(\boldsymbol{x}^t, \boldsymbol{r}^t, \boldsymbol{s}^t) = (\Delta\boldsymbol{x}^t, \Delta\boldsymbol{r}^t, \Delta\boldsymbol{s}^t) + (\boldsymbol{x}, \boldsymbol{r}, \boldsymbol{s})$.

This approach separates motion and geometric structural learning, allowing accurate simulation of dynamic behaviors while maintaining a stable geometric reference. To further enhance scene geometry, we estimate normals of deformed 3D Gaussians and optimize them using:

$$\mathcal{L}_{\text{normal}} = 1 - \mathbf{N}^t \cdot \hat{\mathbf{N}}^t, \tag{3}$$

where $\mathbf{N}^t$ is the rendered normal map and $\hat{\mathbf{N}}^t$ is the normal map derived from the rendered depth map $\mathbf{D}^t$. This process improves local associations among 3D Gaussians and optimizes both depth and normal information across the entire scene.

Specular stage. We adopt an image-based lighting (IBL) model, where the environment light is given by a learnable cubemap. Following the rendering equation [14], split-sum approximation [15, 30], and Cook-Torrance reflectance model [5], the outgoing radiance of the specular component L_s is expressed as:

$$L_s = \int_\Omega \frac{DGF}{4(\omega^t_o \cdot \mathbf{n}^t)(\omega_i \cdot \mathbf{n}^t)}(\omega_i \cdot \mathbf{n}^t)d\omega_i \times \int_\Omega L_i(\omega_i)D(\omega_i, \omega^t_o)(\omega_i \cdot \mathbf{n}^t)d\omega_i, \tag{4}$$

where Ω is the hemisphere around the surface normal $\mathbf{n}^t$ (describe in Sec. 3.3.) D, G, and F represent the GGX normal distribution function [43], geometric attenuation, and Fresnel term, respectively. ω^t_o is the view direction, and $L_i(\omega_i)$ is the incident radiance. In the first term, we follow the GaussianShader [13] directly computed by $\mathbf{s}_{\text{tint}} * F_1 + F_2$,

where F_1 and F_2 are two pre-computed scalars depending on roughness ρ, view direction ω_o^t and normal $\mathbf{n}^t$. Roughness $\rho \in [0,1]$ and specular tint $\mathbf{s_{tint}} \in [0,1]^3$ are learnable parameters for each 3D Gaussian. The second term is pre-integrated in a filtered learnable cubemap, where each mip-level corresponds to a specific roughness value. The cubemap can be queried using the reflection direction to obtain the value of the second term. After the static and dynamic stages, the geometry is well-defined. This allows us to calculate reflection directions $\omega_r^t = 2(\omega_o^t \cdot \mathbf{n}^t)\mathbf{n}^t - \omega_o^t$ accurately.

L_s represents only the specular color of the static environment light. To handle time-varying lighting in dynamic scenes, we introduce a deformable environment map, detailed in the following section.

3.2.2. Deformable Environment Map for Dynamic Lighting.

The concept of a deformable environment map involves treating the vanilla environment cubemap as a canonical environment map and combining it with a deformation field. This approach enables us to model time-varying lighting conditions effectively. We first apply positional encoding to the reflection direction ω_r^t and time t to implement this. These encoded values are then input into a deformable reflection MLP with parameters θ_R. This process allows us to obtain the deformed reflection residual $\Delta\bar{\omega}_r^t = F_{\theta_R}(\gamma(\omega_r^t), \gamma(t))$ for each specified time t.

Subsequently, we add the deformed reflection residual $\Delta\bar{\omega}_r^t$ to the reflection direction ω_r^t, yielding the deformed reflection direction $\bar{\omega}_r^t = \Delta\bar{\omega}_r^t + \omega_r^t$.

We can then use this deformed reflection direction $\bar{\omega}_r^t$ to query the canonical environment map. The queried value is then multiplied by the first term of Equation 4, allowing us to obtain time-varying specular color $\mathbf{c}_s^t$. This approach effectively captures the dynamic nature of lighting in the scene while maintaining a stable canonical reference.

3.2.3. Color Decomposition and Staged Training Strategy.

We decompose the final color $\mathbf{c}_{\mathbf{final}}^t$ into diffuse and specular components to better distinguish between high and low-frequency information: $\mathbf{c}_{\mathbf{final}}^t = \mathbf{c}_d + \mathbf{c}_s^t$, where $\mathbf{c_d}$ is the diffuse color (using zero-order spherical harmonics as view-independent color), and $\mathbf{c_s}^t$ is the view-dependent color component. To manage the transition from spherical harmonics to $\mathbf{c}_{\mathbf{final}}^t$ and mitigate potential geometry disruptions, in the early specular stage, we fix the deformable Gaussian MLP and most 3D Gaussian attributes, optimizing only zero-order SH, specular tint, and roughness. We temporarily suspend densification during this phase. As $\mathbf{c}_{\mathbf{final}}^t$ becomes more complete, we gradually resume optimization of all parameters and reinstate the densification process.

We further split the specular stage into two parts, applying a coarse-to-fine strategy to the environment map. In the first part, we focus on optimizing the canonical environment map for time-invariant lighting. This establishes a stable foundation for the overall lighting structure. In the second part, we proceed to optimize the deformable reflection MLP for dynamic elements. This approach ensures a more robust learning process, allowing us to capture the static lighting conditions before introducing the complexities of dynamic components.

3.3. Physical Normal Estimation

Challenges in normal estimation for 3D Gaussians. Normal estimation is essential for modeling specular objects in 3D Gaussians, where GaussianShader [12] initially used the shortest axis combined with a residual normal for approximation. While this works for static scenes, it becomes problematic with deformed Gaussians because the residual should vary at each time step. A straightforward approach of rotating the residual normal based on quaternion differences between canonical and deformed states proves insufficient, as it does not account for shape changes during deformation. When deformation alters the relative axis lengths, the shortest axis assumption breaks down. This highlights the need for a more comprehensive approach that considers both rotational and shape deformation effects to achieve accurate normal estimation for dynamic specular objects.

Improved rotation calculation for deformed 3D Gaussians. To overcome the limitations of naive methods and accurately model the normal of deformed 3D Gaussians, we propose using both the shortest and longest axes of canonical and deformed Gaussians to compute the rotation matrix. This approach accounts for both rotation and shape changes during deformation. We first align the deformed Gaussian's axes with those of the canonical Gaussian using the following method:

$$\mathbf{v}_s^t = \begin{cases} \mathbf{v}_s^t & \text{if } \mathbf{v}_s \cdot \mathbf{v}_s^t > 0, \\ -\mathbf{v}_s^t & \text{otherwise.} \end{cases}, \quad \mathbf{v}_l^t = \begin{cases} \mathbf{v}_l^t & \text{if } \mathbf{v}_l \cdot \mathbf{v}_l^t > 0, \\ -\mathbf{v}_l^t & \text{otherwise.} \end{cases}, \tag{5}$$

where $\mathbf{v}_s$ and $\mathbf{v}_l$ represent the shortest and longest axes of canonical 3D Gaussians, while $\mathbf{v}_s^t$ and $\mathbf{v}_l^t$ denote the same for deformed 3D Gaussians. We then construct orthogonal matrices using these aligned axes and their cross-products:

$$\mathbf{U} = \begin{bmatrix} \mathbf{v}_s & \mathbf{v}_l & \mathbf{v}_s \times \mathbf{v}_l \end{bmatrix}, \quad \mathbf{V}^t = \begin{bmatrix} \mathbf{v}_s^t & \mathbf{v}_l^t & \mathbf{v}_s^t \times \mathbf{v}_l^t \end{bmatrix}. \tag{6}$$

Finally, we derive the rotation matrix $\mathbf{R}^t = \mathbf{V}^t\mathbf{U}^\top$.

Adjusting normal residuals and ensuring accuracy. To account for shape changes during deformation, we scale the normal residual based on the ratio of oblateness $\frac{\beta}{\beta^t}$ between canonical and deformed 3D Gaussians.

$$\beta = \frac{|\mathbf{v}_l| - |\mathbf{v}_s|}{|\mathbf{v}_l|}, \quad \beta^t = \frac{|\mathbf{v}_l^t| - |\mathbf{v}_s^t|}{|\mathbf{v}_l^t|}, \tag{7}$$

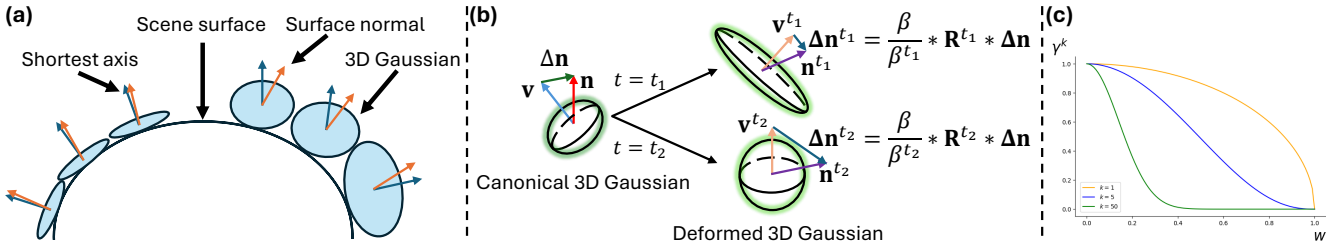

Figure 3. **Normal estimation.** **(a)** shows that flatter 3D Gaussians align better with scene surfaces, their shortest axis closely matching the surface normal. In contrast, less flat 3D Gaussians fit less accurately, with their shortest axis diverging from the surface normal. **(b)** shows that when the deformed 3D Gaussian becomes flatter ($t = t_1$), normal residual $\Delta\mathbf{n}$ is rotated by $\mathbf{R}_1^t$ and scaled down by $\frac{\beta}{\beta_1^t}$, as flatter Gaussians require smaller normal residuals. Conversely, when the deformation results in a less flat shape ($t = t_2$), $\Delta\mathbf{n}$ is rotated by $\mathbf{R}_2^t$ and amplified by $\frac{\beta}{\beta_2^t}$, requiring a larger correction to align the shortest axis with the surface normal. **(c)** shows how γ^k changes with w (where $w = \frac{|\mathbf{v}_s^t|}{|\mathbf{v}_l^t|}$) for $k = 1$, $k = 5$, and $k = 50$. Larger w indicates less flat Gaussians, while smaller w represents flatter Gaussians. As k increases, γ^k decreases more steeply as w rises. For $k = 5$, we observe a balanced behavior: γ^k approaches 1 for low w and 0 for high w, providing a nuanced penalty adjustment across different Gaussian shapes.

where β and β^t represent the oblateness of canonical and deformed 3D Gaussians, respectively. This is because flatter 3D Gaussians tend to align more closely with the surface, meaning their shortest axis becomes more aligned with the surface normal, as shown in Fig. 3 **(a)**. In such cases, less compensation from the normal residual is needed. Conversely, less flat Gaussians require more compensation, as illustrated in Fig. 3 **(b)**. We then obtain deformed normal residuals:

$$\Delta\mathbf{n}^t = \frac{\beta}{\beta^t}\mathbf{R}^t\Delta\mathbf{n}. \tag{8}$$

The final normal $\mathbf{n}^t$ is computed by adding this residual to the shortest axis and ensuring outward orientation:

$$\mathbf{n}^t = \Delta\mathbf{n}^t + \mathbf{v}_s^t, \quad \mathbf{n}^t = \begin{cases} \mathbf{n}^t & \text{if } \mathbf{n}^t \cdot \omega_o^t > 0, \\ -\mathbf{n}^t & \text{otherwise.} \end{cases} \tag{9}$$

This approach adjusts for Gaussian flatness and ensures accurate normal estimation.

3.4. Loss Functions

Normal regularization. To allow the normal residual to correct the normal while not excessively influencing the optimization of the shortest axis towards the surface normal, we introduce a penalty term for the normal residual:

$$\mathcal{L}_{\text{reg}} = \gamma^k \|\Delta\mathbf{n}\|_2^2 \quad \text{where} \quad \gamma = \sqrt{1 - \frac{|\mathbf{v}_s^t|^2}{|\mathbf{v}_l^t|^2}}. \tag{10}$$

In our experiments, we set $k = 5$. When $k = 5$, less flatter 3D Gaussians have γ^k approaching 0. Their shortest axis aligns poorly with the surface normal, requiring more normal residual correction and smaller penalties. Conversely, flatter Gaussians have γ^k near 1. Their shortest axis aligns better with the surface normal, needing less normal residual correction and allowing larger penalties, as shown in Fig. 3 **(c)**.

Total training loss. To refine all parameters in the dynamic and specular stages, we employ the total training loss:

$$\mathcal{L} = \mathcal{L}_{\text{color}} + \lambda_{\text{normal}}\mathcal{L}_{\text{normal}} + \mathcal{L}_{\text{reg}}, \tag{11}$$

where $\mathcal{L}_{\text{color}}$ and $\mathcal{L}_{\text{normal}}$ are obtained as described in Section 3.2.1. In our experiments, we set $\lambda_{\text{normal}} = 0.01$. Due to space constraints, complete implementation details are provided in the supplementary materials.

4. Experiments

4.1. Evaluation Results

We evaluate our method on two real-world datasets: NeRF-DS dataset [48] and HyperNeRF dataset [32]. While GaussianShader [12] and GS-IR [23] are originally designed for static scenes and are included here only as reference baselines, we train our method and all baseline approaches for 40,000 iterations to ensure fair comparison.

NeRF-DS dataset. The NeRF-DS dataset [48] is a monocular video dataset comprising seven real-world scenes from daily life featuring various types of moving or deforming specular objects. We compare our method with the most relevant state-of-the-art approaches. As shown in Tab. 1 and Fig. 4, the quantitative results demonstrate that our method decisively outperforms baselines in reconstructing and rendering real-world highly reflective dynamic specular scenes.

The rendering speed is correlated with the quantity of 3D Gaussians. When the number of 3D Gaussians is below 178k, our method can achieve real-time rendering over 30 FPS on an NVIDIA RTX 4090.

HyperNeRF dataset. The HyperNeRF dataset contains real-world dynamic scenes and does not include specular objects.

Table 1. **Quantitative comparison on the NeRF-DS [48] dataset.** We report the average PSNR, SSIM, and LPIPS (VGG) of several previous models on test images. The best, the second best, and third best results are denoted by red, orange, yellow.

Method	As			Basin			Bell			Cup		
	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
Deformable 3DGS [49]	26.04	0.8805	0.1850	19.53	0.7855	0.1924	23.96	0.7945	0.2767	24.49	0.8822	0.1658
4DGS [46]	24.85	0.8632	0.2038	19.26	0.7670	0.2196	22.86	0.8015	0.2061	23.82	0.8695	0.1792
GaussianShader [12]	21.89	0.7739	0.3620	17.79	0.6670	0.4187	20.69	0.8169	0.3024	20.40	0.7437	0.3385
GS-IR [23]	21.58	0.8033	0.3033	18.06	0.7248	0.3135	20.66	0.7829	0.2603	20.34	0.8193	0.2719
NeRF-DS [48]	25.34	0.8803	0.2150	20.23	0.8053	0.2508	22.57	0.7811	0.2921	24.51	0.8802	0.1707
HyperNeRF [32]	17.59	0.8518	0.2390	22.58	0.8156	0.2497	19.80	0.7650	0.2999	15.45	0.8295	0.2302
Ours	26.80	0.8843	0.1761	19.75	0.7915	0.1896	25.46	0.8490	0.1600	24.65	0.8871	0.1588

Method	Plate			Press			Sieve			**Mean**		
	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
Deformable 3DGS [49]	19.07	0.7352	0.3599	25.52	0.8594	0.1964	25.37	0.8616	0.1643	23.43	0.8284	0.2201
4DGS [46]	18.77	0.7709	0.2721	24.82	0.8355	0.2255	25.16	0.8566	0.1745	22.79	0.8235	0.2115
GaussianShader [12]	14.55	0.6423	0.4955	19.97	0.7244	0.4507	22.58	0.7862	0.3057	19.70	0.7363	0.3819
GS-IR [23]	15.98	0.6969	0.4200	22.28	0.8088	0.3067	22.84	0.8212	0.2236	20.25	0.7796	0.2999
NeRF-DS [48]	19.70	0.7813	0.2974	25.35	0.8703	0.2552	24.99	0.8705	0.2001	23.24	0.8384	0.2402
HyperNeRF [32]	21.22	0.7829	0.3166	16.54	0.8200	0.2810	19.92	0.8521	0.2142	19.01	0.8167	0.2615
Ours	20.84	0.8172	0.2198	26.49	0.8657	0.1889	25.22	0.8705	0.1513	24.17	0.8522	0.1778

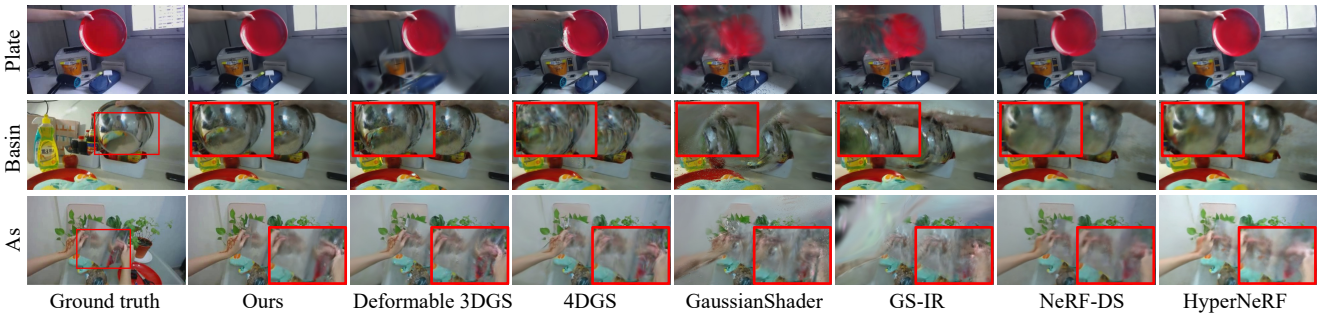

Figure 4. **Qualitative comparison on the NeRF-DS [48] dataset.**

Table 2. **Quantitative comparison on the HyperNeRF [32] dataset.** We report the average PSNR, SSIM, and LPIPS (VGG) of several previous models. The best, the second best, and third best results are denoted by red, orange, yellow.

Method	PSNR ↑	SSIM ↑	LPIPS ↓
Deformable 3DGS [49]	22.78	0.6201	0.3380
4DGS [46]	24.89	0.6781	0.3422
GaussianShader [12]	18.55	0.5452	0.4795
GS-IR [23]	19.87	0.5729	0.4498
NeRF-DS [48]	23.65	0.6405	0.3972
HyperNeRF [32]	23.11	0.6387	0.3968
Ours	22.22	0.6088	0.3295

As shown in Tab. 2 and Fig. 5, the results demonstrate that we are on par with state-of-the-art techniques for rendering novel views, and our method's performance is not limited to shiny scenes.

In Fig. 6, we compare our method's normal maps with those from Deformable 3DGS [49] and NeRF-DS [48]. For Deformable 3DGS [49], we obtain the normals by using the shortest axes of the deformed 3D Gaussians. As demonstrated, our method produces significantly better quality normal maps compared to Deformable 3DGS [49] and NeRF-DS [48].

4.2. Ablation Study

For a fair comparison, we train our method and all ablation experiments for 40,000 iterations.

Different coarse to fine training strategy stages. As shown in Tab. 3 and Fig. 7, each stage contributes effectively to the model's performance. The Dynamic stage enhances dynamic object stability compared to the Static stage alone, while the Specular stage improves reflection clarity beyond the combined Static and Dynamic stages.

Figure 5. **Qualitative comparison on the HyperNeRF [32] dataset.**

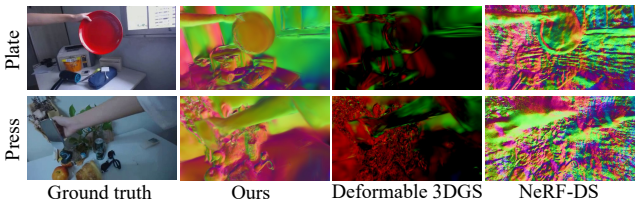

Figure 6. **Qualitative comparison of normal maps between our method, Deformable 3DGS, and NeRF-DS.**

Table 3. **Ablation studies on different coarse to fine training strategy stages.**

Stage	PSNR ↑	SSIM ↑	LPIPS ↓
Static	20.26	0.7785	0.2953
St. + Dynamic	24.02	0.8508	0.1831
St. + Dy. + Specular	**24.17**	**0.8522**	**0.1778**

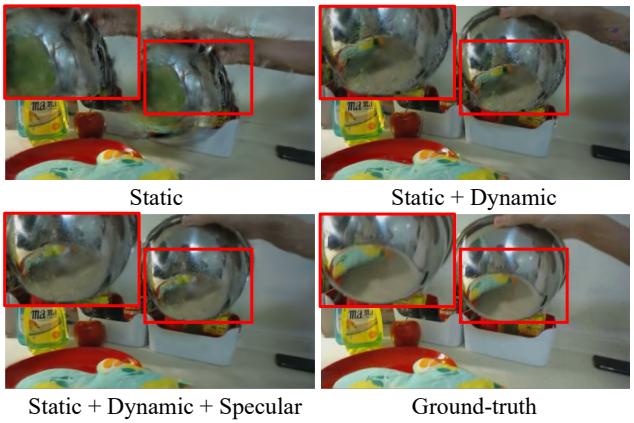

Figure 7. **Qualitative comparison of each training stage in our coarse-to-fine approach.**

Ablation study on coarse-to-fine, and loss function. The model's performance was evaluated without key components: the coarse-to-fine training strategy, normal loss $\mathcal{L}_{\text{normal}}$, normal regularization $\mathcal{L}_{\text{reg}}$, and γ^k. Fig. 8 and Tab. 4 illustrate the effects of these omissions. Without the coarse-to-fine approach, the model, trained directly at the specular stage, produces incomplete scene geometry, affecting environment map queries for specular color. Omitting normal loss $\mathcal{L}_{\text{normal}}$ removes direct supervision on normals and leads to inaccurate reflection directions and less precise specular colors. Removing normal regularization $\mathcal{L}_{\text{reg}}$ allows the normal residual to dominate normal optimization, resulting in insufficient optimization of the 3D Gaussians' shortest axis towards the correct normal, which in turn reduces the rendering quality. The normal residual decreases for non-flattened and flat Gaussians without γ^k in normal regularization. This particularly affects less flat 3D Gaussians whose shortest axis significantly deviates from the surface normal. The insufficient normal residual correction causes these 3D Gaussians' shortest axes to deviate greatly from their original direction in an attempt to align with the surface normal, ultimately reducing rendering quality.

Table 4. **Ablation studies on different coarse to fine training strategy stages.**

C2F	$\mathcal{L}_{\text{normal}}$	$\mathcal{L}_{\text{reg}}$	γ^k	PSNR↑	SSIM↑	LPIPS↓
	✓	✓	✓	23.16	0.8294	0.2156
✓				23.40	0.8277	0.2278
✓	✓			24.15	0.8510	0.1845
✓	✓	✓		24.09	0.8515	0.1818
✓	✓	✓	✓	**24.17**	**0.8522**	**0.1778**

Table 5. **Ablation studies on SH, Static and Deformable environment map.**

	PSNR ↑	SSIM ↑	LPIPS ↓
SH	23.63	0.8453	0.1844
Static Env. map	24.02	0.8508	0.1831
Deformable Env. map	**24.17**	**0.8522**	**0.1778**

Ablation study on SH, Static environment map, and Deformable environment map. Fig. 9 and Tab. 5 demonstrate the superiority of the deformable environment map over the static environment map, which in turn outperforms Spherical Harmonics (SH). SH struggles to accurately model high-

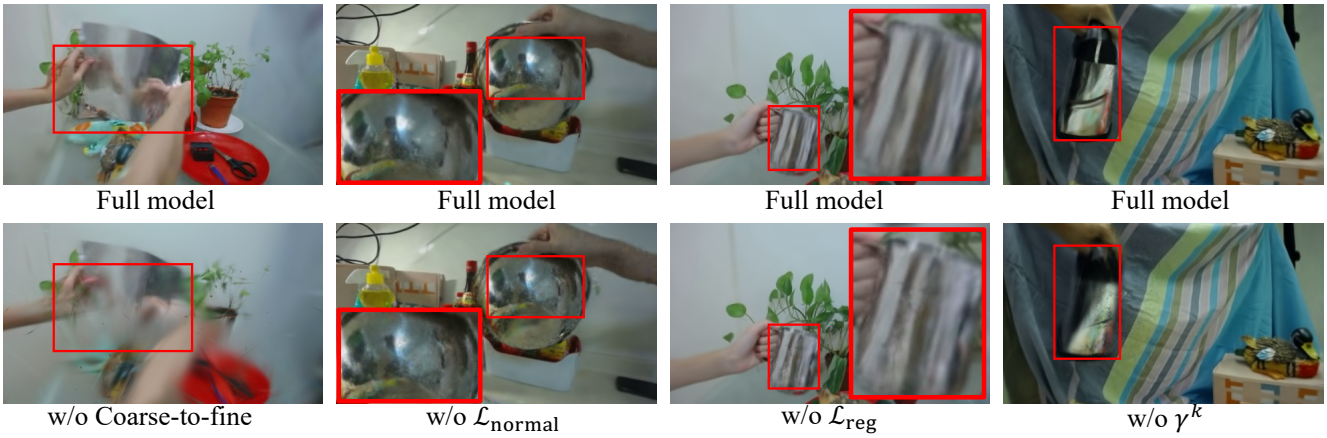

Figure 8. **Qualitative comparison of ablation study without different components.**

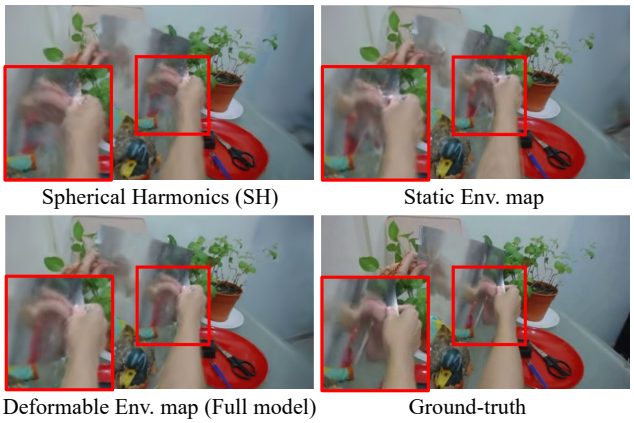

Figure 9. **Qualitative comparison of ablation study on SH, Static environment map, and Deformable environment map.**

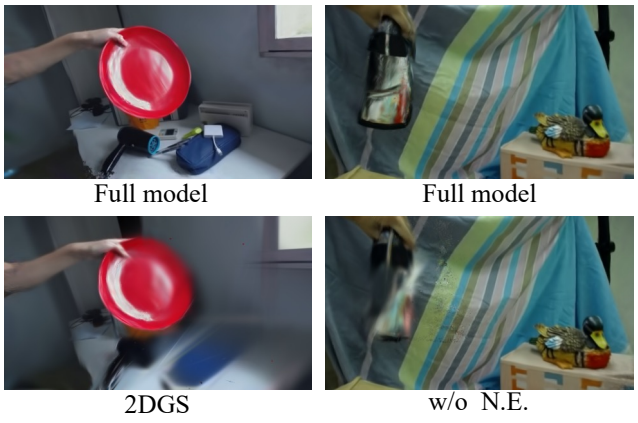

Figure 10. **Qualitative comparison of ablation study on 2DGS and without Physical Normal Estimation.**

Table 6. **Ablation studies on 2DGS and without Physical Normal Estimation.**

	PSNR ↑	SSIM ↑	LPIPS ↓
2DGS [10]	23.22	0.8219	0.2283
w/o N.E.	23.89	0.8490	0.1837
Full model	**24.17**	**0.8522**	**0.1778**

frequency specular colors. While the static environment map can model high-frequency colors, it is best suited for static lighting conditions. In contrast, the deformable environment map models time-varying lighting, offering superior performance for dynamic scenes.

Ablation study on 2DGS [10] and without Physical Normal Estimation. In Fig. 10 and Tab. 6, "2DGS" represents replacing our 3D Gaussian with 2D Gaussian representation. Since 2DGS inherently includes normals, we omit physical normal estimation. "w/o N.E." means skipping physical normal estimation and using the shortest axis of 3D Gaussians as the normal. This causes the normal loss $\mathcal{L}_{\text{normal}}$ to directly supervise the shortest axes, making some axes deviate significantly to align with surface normals, resulting in degraded rendering quality.

5. Conclusion

SpectroMotion enhances 3D Gaussian Splatting for dynamic specular scenes by combining specular rendering with deformation fields. Using normal residual correction, coarse-to-fine training, and a deformable environment map, it achieves superior accuracy and visual quality in novel view synthesis, outperforming existing methods while maintaining geometric consistency.

Limitations. Though we stabilize the entire scene's geometry using a coarse-to-fine training strategy, some failure cases still occur. Please refer to the supplementary materials for visual results of failure cases.

Acknowledgements. This research was funded by the National Science and Technology Council, Taiwan, under Grants NSTC 112-2222-E-A49-004-MY2 and 113-2628-E-A49-023-. The authors are grateful to Google, NVIDIA, and MediaTek Inc. for their generous donations. Yu-Lun Liu acknowledges the Yushan Young Fellow Program by the MOE in Taiwan.

References

[1] Benjamin Attal, Eliot Laidlaw, Aaron Gokaslan, Changil Kim, Christian Richardt, James Tompkin, and Matthew O'Toole. Törf: Time-of-flight radiance fields for dynamic scene view synthesis, 2021. 2

[2] 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, 2020. 2

[3] Mark Boss, Varun Jampani, Raphael Braun, Ce Liu, Jonathan T. Barron, and Hendrik P. A. Lensch. Neural-pil: Neural pre-integrated lighting for reflectance decomposition, 2021. 2

[4] Ting-Hsuan Chen, Jie Wen Chan, Hau-Shiang Shiu, Shih-Han Yen, Changhan Yeh, and Yu-Lun Liu. Narcan: Natural refined canonical image with integration of diffusion prior for video editing. In *NeurIPS*, 2024. 2

[5] Robert L Cook and Kenneth E. Torrance. A reflectance model for computer graphics. *ACM Transactions on Graphics (ToG)*, 1(1):7–24, 1982. 3

[6] Ainaz Eftekhar, Alexander Sax, Jitendra Malik, and Amir Zamir. Omnidata: A scalable pipeline for making multi-task mid-level vision datasets from 3d scans. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10786–10796, 2021. 1

[7] Jian Gao, Chun Gu, Youtian Lin, Hao Zhu, Xun Cao, Li Zhang, and Yao Yao. Relightable 3d gaussian: Real-time point cloud relighting with brdf decomposition and ray tracing. *arXiv preprint arXiv:2311.16043*, 2023. 2

[8] 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, 2023. 2

[9] Xiang Guo, Jiadai Sun, Yuchao Dai, Guanying Chen, Xiaoqing Ye, Xiao Tan, Errui Ding, Yumeng Zhang, and Jingdong Wang. Forward flow for novel view synthesis of dynamic scenes, 2023. 2

[10] 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. 8

[11] Yi-Hua Huang, Yang-Tian Sun, Ziyi Yang, Xiaoyang Lyu, Yan-Pei Cao, and Xiaojuan Qi. Sc-gs: Sparse-controlled gaussian splatting for editable dynamic scenes, 2024. 2

[12] 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. *arXiv preprint arXiv:2311.17977*, 2023. 2, 4, 5, 6

[13] 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*, pages 5322–5332, 2024. 3

[14] James T Kajiya. The rendering equation. In *Proceedings of the 13th annual conference on Computer graphics and interactive techniques*, pages 143–150, 1986. 3

[15] Brian Karis and Epic Games. Real shading in unreal engine 4. *Proc. Physically Based Shading Theory Practice*, 4(3):1, 2013. 3

[16] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering, 2023. 1, 2

[17] Junxuan Li and Hongdong Li. Neural reflectance for shape recovery with shadow handling, 2022. 2

[18] Tianye Li, Mira Slavcheva, Michael Zollhoefer, Simon Green, Christoph Lassner, Changil Kim, Tanner Schmidt, Steven Lovegrove, Michael Goesele, Richard Newcombe, and Zhaoyang Lv. Neural 3d video synthesis from multi-view video, 2022. 2

[19] Zhengqi Li, Simon Niklaus, Noah Snavely, and Oliver Wang. Neural scene flow fields for space-time view synthesis of dynamic scenes, 2021. 2

[20] Ruofan Liang, Huiting Chen, Chunlin Li, Fan Chen, Selvakumar Panneer, and Nandita Vijaykumar. Envidr: Implicit differentiable renderer with neural environment lighting, 2023. 2

[21] Ruofan Liang, Jiahao Zhang, Haoda Li, Chen Yang, Yushi Guan, and Nandita Vijaykumar. Spidr: Sdf-based neural point fields for illumination and deformation, 2023. 2

[22] Yiqing Liang, Numair Khan, Zhengqin Li, Thu Nguyen-Phuoc, Douglas Lanman, James Tompkin, and Lei Xiao. Gaufre: Gaussian deformation fields for real-time dynamic novel view synthesis, 2023. 2

[23] Zhihao Liang, Qi Zhang, Ying Feng, Ying Shan, and Kui Jia. Gs-ir: 3d gaussian splatting for inverse rendering. *arXiv preprint arXiv:2311.16473*, 2023. 2, 5, 6

[24] Haotong Lin, Sida Peng, Zhen Xu, Yunzhi Yan, Qing Shuai, Hujun Bao, and Xiaowei Zhou. Efficient neural radiance fields for interactive free-viewpoint video, 2022. 2

[25] 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, 2023. 2

[26] Yu-Lun Liu, Chen Gao, Andreas Meuleman, Hung-Yu Tseng, Ayush Saraf, Changil Kim, Yung-Yu Chuang, Johannes Kopf, and Jia-Bin Huang. Robust dynamic radiance fields. In *CVPR*, 2023. 2

[27] Caoyuan Ma, Yu-Lun Liu, Zhixiang Wang, Wu Liu, Xinchen Liu, and Zheng Wang. Humannerf-se: A simple yet effective approach to animate humannerf with diverse poses. In *CVPR*, 2024. 2

[28] Li Ma, Vasu Agrawal, Haithem Turki, Changil Kim, Chen Gao, Pedro Sander, Michael Zollhöfer, and Christian Richardt. Specnerf: Gaussian directional encoding for specular reflections. *arXiv preprint arXiv:2312.13102*, 2023. 2

[29] Marko Mihajlovic, Sergey Prokudin, Siyu Tang, Robert Maier, Federica Bogo, Tony Tung, and Edmond Boyer. Splatfields: Neural gaussian splats for sparse 3d and 4d reconstruction. *arXiv preprint arXiv:2409.11211*, 2024. 2

[30] 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. 2, 3

[31] Keunhong Park, Utkarsh Sinha, Jonathan T. Barron, Sofien Bouaziz, Dan B Goldman, Steven M. Seitz, and Ricardo Martin-Brualla. Nerfies: Deformable neural radiance fields, 2021. 2

[32] Keunhong Park, Utkarsh Sinha, Peter Hedman, Jonathan T. Barron, Sofien Bouaziz, Dan B Goldman, Ricardo Martin-Brualla, and Steven M. Seitz. Hypernerf: A higher-dimensional representation for topologically varying neural radiance fields, 2021. 2, 5, 6, 7

[33] Sida Peng, Yunzhi Yan, Qing Shuai, Hujun Bao, and Xiaowei Zhou. Representing volumetric videos as dynamic mlp maps, 2023. 2

[34] Albert Pumarola, Enric Corona, Gerard Pons-Moll, and Francesc Moreno-Noguer. D-nerf: Neural radiance fields for dynamic scenes, 2020. 2

[35] Yahao Shi, Yanmin Wu, Chenming Wu, Xing Liu, Chen Zhao, Haocheng Feng, Jingtuo Liu, Liangjun Zhang, Jian Zhang, Bin Zhou, et al. Gir: 3d gaussian inverse rendering for relightable scene factorization. *arXiv preprint arXiv:2312.05133*, 2023. 2

[36] Liangchen Song, Anpei Chen, Zhong Li, Zhang Chen, Lele Chen, Junsong Yuan, Yi Xu, and Andreas Geiger. Nerfplayer: A streamable dynamic scene representation with decomposed neural radiance fields, 2023. 2

[37] 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, 2020. 2

[38] Colton Stearns, Adam Harley, Mikaela Uy, Florian Dubost, Federico Tombari, Gordon Wetzstein, and Leonidas Guibas. Dynamic gaussian marbles for novel view synthesis of casual monocular videos. *arXiv preprint arXiv:2406.18717*, 2024. 2

[39] Edgar Tretschk, Ayush Tewari, Vladislav Golyanik, Michael Zollhöfer, Christoph Lassner, and Christian Theobalt. Non-rigid neural radiance fields: Reconstruction and novel view synthesis of a dynamic scene from monocular video, 2021. 2

[40] 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. 2

[41] Dor Verbin, Ben Mildenhall, Peter Hedman, Jonathan T Barron, Todd Zickler, and Pratul P Srinivasan. Eclipse: Disambiguating illumination and materials using unintended shadows. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 77–86, 2024. 2

[42] Dor Verbin, Pratul P Srinivasan, Peter Hedman, Ben Mildenhall, Benjamin Attal, Richard Szeliski, and Jonathan T Barron. Nerf-casting: Improved view-dependent appearance with consistent reflections. *arXiv preprint arXiv:2405.14871*, 2024. 2

[43] Bruce Walter, Stephen R Marschner, Hongsong Li, and Kenneth E Torrance. Microfacet models for refraction through rough surfaces. *Rendering techniques*, 2007:18th, 2007. 3

[44] Qianqian Wang, Vickie Ye, Hang Gao, Jake Austin, Zhengqi Li, and Angjoo Kanazawa. Shape of motion: 4d reconstruction from a single video. *arXiv preprint arXiv:2407.13764*, 2024. 2

[45] Chun-Hung Wu, Shih-Hong Chen, Chih-Yao Hu, Hsin-Yu Wu, Kai-Hsin Chen, Yu-You Chen, Chih-Hai Su, Chih-Kuo Lee, and Yu-Lun Liu. Denver: Deformable neural vessel representations for unsupervised video vessel segmentation. In *CVPR*, 2025. 2

[46] 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. *arXiv preprint arXiv:2310.08528*, 2023. 2, 6

[47] Wenqi Xian, Jia-Bin Huang, Johannes Kopf, and Changil Kim. Space-time neural irradiance fields for free-viewpoint video, 2021. 2

[48] Zhiwen Yan, Chen Li, and Gim Hee Lee. Nerf-ds: Neural radiance fields for dynamic specular objects. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8285–8295, 2023. 5, 6

[49] Ziyi Yang, Xinyu Gao, Wen Zhou, Shaohui Jiao, Yuqing Zhang, and Xiaogang Jin. Deformable 3d gaussians for high-fidelity monocular dynamic scene reconstruction. *arXiv preprint arXiv:2309.13101*, 2023. 2, 3, 6

[50] Ziyi Yang, Xinyu Gao, Yangtian Sun, Yihua Huang, Xiaoyang Lyu, Wen Zhou, Shaohui Jiao, Xiaojuan Qi, and Xiaogang Jin. Spec-gaussian: Anisotropic view-dependent appearance for 3d gaussian splatting, 2024. 2

[51] Keyang Ye, Qiming Hou, and Kun Zhou. 3d gaussian splatting with deferred reflection. In *ACM SIGGRAPH 2024 Conference Papers*, pages 1–10, 2024. 2

[52] Wang Yifan, Felice Serena, Shihao Wu, Cengiz Öztireli, and Olga Sorkine-Hornung. Differentiable surface splatting for point-based geometry processing. *ACM Transactions on Graphics (TOG)*, 38(6):1–14, 2019. 2

[53] Jingyang Zhang, Yao Yao, Shiwei Li, Jingbo Liu, Tian Fang, David McKinnon, Yanghai Tsin, and Long Quan. Neilf++: Inter-reflectable light fields for geometry and material estimation, 2023. 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. 2

[55] Xiuming Zhang, Pratul P. Srinivasan, Boyang Deng, Paul Debevec, William T. Freeman, and Jonathan T. Barron. Nerfactor: neural factorization of shape and reflectance under an unknown illumination. *ACM Transactions on Graphics*, 40 (6):1–18, 2021.

[56] Xiaoming Zhao, Pratul P Srinivasan, Dor Verbin, Keunhong Park, Ricardo Martin Brualla, and Philipp Henzler. Illuminerf: 3d relighting without inverse rendering. *arXiv preprint arXiv:2406.06527*, 2024. 2

[57] 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