

DR-GS: Physically-Based Deformable and Relightable 2D Gaussians

Jiaxin Li^{1,2,3}, Tong Wu⁵, Yi Wei⁵, Tailin Wu³, and Li Zhang^{1,4*}

¹ Shanghai Innovation Institute, China

² Zhejiang University, China

³ Department of Artificial Intelligence, Westlake University, China

⁴ School of Data Science, Fudan University, China

⁵ Central Media Technology Institute, Huawei, China

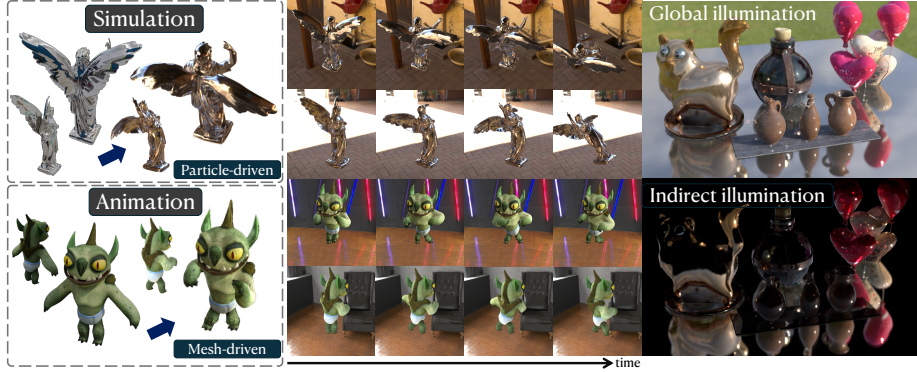


Fig. 1: DR-GS enables physically plausible rendering under deformations and lighting changes via decoupled geometry, lighting, and materials. The project page is available at <https://jiaxinlia.github.io/DR-GS/>.

Abstract. Gaussian splatting (GS) has garnered significant attention in VR/AR and digital content creation due to its explicit parameterization and efficient rendering capabilities. However, existing GS-based methods for deformable objects face two key limitations: (i) illumination is erroneously baked into textures, causing physically inconsistent responses under dynamic deformations and lighting changes; (ii) snapshot-based reconstruction restricts post-reconstruction material editing. To address these challenges, we propose **D**eformable and **R**elightable **G**S (**DR-GS**), a unified Gaussian framework that integrates physically-based inverse rendering, relighting, and deformation-aware manipulation. Through explicitly disentangling geometry, illumination, and material representations, DR-GS overcomes the limitations of static snapshots, resolving unrealistic appearance under varying conditions while enabling post-reconstruction parameter editing. Extensive experiments show that DR-GS achieves leading visual quality across static reconstruction, dynamic

* Corresponding author. E-mail: lizhangfd@fudan.edu.cn

deformation, and relighting, reliably preserving reflections and specular highlights on glossy surfaces. It further establishes a fully decoupled geometry-illumination-material pipeline, enabling high-quality 3D asset creation and comprehensive post-editing.

Keywords: 2D Gaussian Splatting · Inverse Rendering · Physics-Based Rendering

1 Introduction

Gaussian splatting (GS) represents 3D scenes with learnable anisotropic Gaussians, providing an explicit scene representation and real-time rendering. Compared with NeRF [36], GS achieves substantially higher rendering throughput and supports interactive applications that require both realism and low latency, such as VR/AR [14, 21, 28, 33], avatars [17, 26, 40], and embodied systems [20, 32, 45, 57, 62].

Existing deformable GS methods largely follow a reconstruct-then-drive pipeline [9, 10, 13, 19, 30, 53]: a static reconstruction is first obtained from multi-view images, then Gaussians are driven by particle-based or mesh-based deformation. This design introduces two fundamental limitations. First, appearance is typically baked into per-Gaussian colors, yielding physically inconsistent responses under deformation and relighting, especially for glossy surfaces. Second, the static reconstruction stage restricts post-hoc editing of materials. While inverse rendering can disentangle geometry, materials, and illumination for photorealistic relighting, existing solutions remain largely confined to static scenes.

We present **DR-GS**, a unified deformable Gaussian framework that jointly supports inverse rendering, relighting, physics-based simulation, and 3D animation within a shared representation. DR-GS models materials from diffuse to specular for reconstruction, and introduces a hybrid driving mechanism that supports both particle systems and mesh deformations. Following the separation principle in GSP [7], we decouple simulation from rendering and update Gaussians through generalized moving least squares (GMLS) interpolation [35].

To render dynamic scenes under changing illumination, DR-GS estimates the full rendering equation with low-sample Monte Carlo integration and multiple importance sampling (MIS), combining cosine-weighted, GGX [16], and environment-light sampling [39], followed by cross-bilateral denoising [42]. For efficiency, we store decoupled material parameters on TSDF-extracted mesh vertices and perform mesh-based ray tracing, replacing 2D Gaussian ray tracing [11] during continuous deformation. This design preserves high-quality shadows and inter-reflections while significantly improving runtime.

To summarize, our contributions include: **(i) Unified deformable GS:** DR-GS integrates physically-based inverse rendering, relighting, and deformation-driven control in a single Gaussian framework; **(ii) Efficient dynamic rendering:** We combine MIS-based Monte Carlo rendering with denoising and mesh-based ray tracing to accelerate relighting under deformation efficiently; **(iii) Editable asset pipeline:** A decoupled geometry-illumination-material

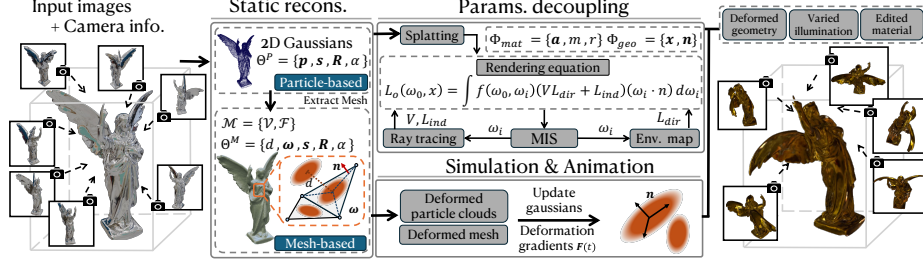


Fig. 2: Overview of DR-GS. Our framework consists of three stages: (i) Static reconstruction: building initial 2D Gaussians and mesh from calibrated images, with optional mesh-based reinitialization; (ii) Parameters decoupling: generating material (*albedo, metallic, roughness*) and geometry (*position, normal*) maps via splatting while solving the rendering equation through MIS and ray tracing; (iii) Dynamic driving: updating Gaussians via deformation gradients between canonical and deformed particle clouds/meshes from physical simulation/animation. Overall, our decoupled geometry-illumination-material framework supports physically plausible rendering under deformations, illumination changes, and material edits with interactive manipulation.

pipeline enables high-fidelity creation and post-editing of 3D assets for content production and simulation.

2 Related works

Dynamic Gaussian splatting. Prior work mainly falls into two paradigms. Learning-based methods take videos as input and encode temporal variation directly into Gaussians [5, 17, 26, 27, 34, 40, 52, 54, 55]. Deformation-driven methods follow a reconstruct-then-drive pipeline: they first reconstruct a static GS from multi-view images and then drive deformation via physics simulation or mesh manipulation. Our framework adopts the latter. Within this line, physics-inspired approaches couple GS with material-aware simulation [3, 19, 30, 33, 41, 60, 61], e.g., MPM in PhysGaussian [53] and PBD in GSP [7], while geometry-driven methods rely on mesh editing for artist-controlled deformation [9, 10, 13, 21, 49]. Although effective for rendering static appearance, existing methods largely bake lighting into colors and thus cannot faithfully reproduce appearance changes under deformation or varying illumination, producing noticeable artifacts, especially on glossy surfaces. DR-GS addresses this limitation with a decoupled representation that stores editable material parameters separately and renders them with physically-based rendering, improving realism in dynamic scenarios. **Physically-based inverse rendering** aims to recover geometry, materials, and illumination from images, but is inherently ill-posed due to ambiguities between observations and physical parameters. Most methods rely on differentiable rendering with physics-based light transport. NeRF-based approaches [1, 2, 12, 31, 46, 58, 63] can model global illumination via ray marching over neural fields, but remain computationally expensive. In contrast, 3DGS [24] provides an efficient

explicit representation and has been extended with material modeling for inverse rendering [4, 8, 15, 25, 29, 44, 47, 56]. Ref-Gaussian [56] uses deferred BRDF shading and mesh ray tracing for specular visibility, but the split-sum approximation can compromise material and lighting accuracy. IRGS [11] evaluates the full rendering equation with 2D Gaussian ray tracing to capture inter-reflections, but its heavy stratified sampling is inefficient and often struggles on glossy surfaces. To support deformable objects under changing illumination, DR-GS builds on these advances while explicitly maintaining the full physically-based rendering equation and improving efficiency in dynamic settings.

3 Method

This section introduces DR-GS, a decoupled deformable and relightable GS framework for physically-plausible rendering of deformable objects under varying illumination. Our three-stage pipeline first establishes the foundations of 2D Gaussian splatting and Gaussian ray tracing (Sec. 3.1). The static reconstruction stage builds initial 2D Gaussians and mesh representations from multi-view imagery (Sec. 3.2). We then solve the rendering equation to decouple material attributes from illumination while simultaneously optimizing geometric parameters (Sec. 3.3). Finally, we develop a physically-based deformation-driven method for 2D Gaussians utilizing physical simulation and animation (Sec. 3.4). The full pipeline is illustrated in Fig. 2.

3.1 Preliminary

2D Gaussian splatting [18] addresses key limitations of 3DGS [24], including its lack of explicit surface normals and multi-view inconsistencies, through geometrically constrained explicit surface representation. It compresses 3D Gaussians into 2D surfels parameterized by a center $\mathbf{p}$, opacity α , view-dependent color $\mathbf{c}$, axial scaling $\mathbf{s} = (s_u, s_v)$, and a rotation matrix $\mathbf{R}$ built from orthonormal tangents $\mathbf{t}_u$ and $\mathbf{t}_v$. The surface normal is explicitly derived as $\mathbf{n} = \mathbf{t}_u \times \mathbf{t}_v$, ensuring view-consistent geometry. A central innovation is ray-splat intersection, which maps screen-space pixels to UV coordinates via the Gaussian kernel $G(\mathbf{u}) = \exp(-(u^2 + v^2)/2)$, augmented with perspective-correct splatting to reduce multi-view artifacts. Rendering uses depth-ordered alpha blending:

$$\mathbf{c}(\mathbf{r}) = \sum_{i=1}^N \mathbf{c}_i \alpha_i \hat{G}_i(\mathbf{u}) \prod_{j=1}^{i-1} (1 - \alpha_j \hat{G}_j(\mathbf{u})), \quad (1)$$

enabling end-to-end optimization of learnable parameters $\Theta_i = \{\mathbf{p}_i, \mathbf{s}_i, \mathbf{R}_i, \alpha_i, \mathbf{c}_i\}$.

Gaussian ray tracing integrates ray tracing techniques with Gaussian primitives to overcome limitations of rasterization-based Gaussian splatting in simulating effects like shadows and inter-reflections. The pioneering 3DGRT [37] introduces particle-based ray tracing for 3D Gaussians, using a k -buffer hit-based marching algorithm with OptiX [38] hardware acceleration to improve

both speed and accuracy. Following this advancement, 2DGRT [11] resolves ray-splat inconsistencies through explicit surface representation and geometric constraints. This approach achieves physically accurate ray tracing, particularly for complex light paths involving multi-bounce indirect illumination.

3.2 Static Reconstruction

Initialization. Our framework begins by pretraining with Ref-Gaussian [56] to acquire initial 2D Gaussians $\Theta^P = \{\mathbf{p}, \mathbf{s}, \mathbf{R}, \alpha\}$ and subsequently extracts triangular mesh $\mathcal{M} = \{\mathcal{V}, \mathcal{F}\}$ via TSDF. DR-GS accommodates dual deformation approaches: particle-based deformation with optional mesh filling and mesh-based deformation necessitating reinitialization of Θ^P with $\mathcal{M}$.

Mesh-based reinitialization. We initialize a set of 2D Gaussians $\Theta^M = \{d, \boldsymbol{\omega}, \mathbf{s}, \mathbf{R}, \alpha\}$ on each triangular face $f \in \mathcal{F}$, where $\mathbf{R}$ aligns with the face normal $\mathbf{n}$, and $\mathbf{s}$ determined by local geometric properties. Same as Gaussian-Mesh [9], each Gaussian center $\mathbf{p}$ is parameterized by interpolation weights $\boldsymbol{\omega} = \{\omega_a, \omega_b, \omega_c\}$ and normal offset d , initialized as barycentric coordinates and zero, respectively. The relationship can be expressed as $\mathbf{p} = (\omega_a \mathbf{v}_a + \omega_b \mathbf{v}_b + \omega_c \mathbf{v}_c) + dr\mathbf{n}$, where r denotes the circumradius of the associated triangle with vertices $\{\mathbf{v}_a, \mathbf{v}_b, \mathbf{v}_c\}$. Pretrained Gaussians Θ^P are projected onto mesh surfaces using spatial acceleration structure, retaining only interior projections. Finally, we compute averaged $\boldsymbol{\omega}$ and α per face and transfer them to corresponding mesh-based Gaussians Θ^M , achieving geometry-aware reparameterization. See Alg. 1 for details.

Algorithm 1 Mesh-based 2D Gaussian Initialization

Input: Mesh $\mathcal{M} = (\mathcal{V}, \mathcal{F})$, pretrained Gaussians $\Theta^P = \{(\mathbf{p}_i, \alpha_i)\}_{i=1}^{N_p}$.

Output: Face-attached Gaussians $\Theta^M = \{(d_j, \boldsymbol{\omega}_j, \mathbf{s}_j, \mathbf{R}_j, \alpha_j)\}_{j=1}^{N_f}$.

- 1: **Per-face init.**
 - 2: **for** $j = 1, \dots, N_f$ **do**
 - 3: $\mathbf{n}_j \propto (\mathbf{v}_{j2} - \mathbf{v}_{j1}) \times (\mathbf{v}_{j3} - \mathbf{v}_{j1})$, $\mathbf{R}_j \leftarrow \text{Align}(\mathbf{n}_j)$
 - 4: $\mathbf{c}_j \leftarrow (\mathbf{v}_{j1} + \mathbf{v}_{j2} + \mathbf{v}_{j3})/3$, $d_j \leftarrow \min_{k \neq j} \|\mathbf{c}_j - \mathbf{c}_k\|$
 - 5: $d_j \leftarrow 0$, $\boldsymbol{\omega}_j \leftarrow (1/3, 1/3, 1/3)$, $\mathbf{s}_j \leftarrow \log(d_j + \epsilon)\mathbf{1}_2$
 - 6: **end for**
 - 7: **Project to faces.** Build BVH on $\mathcal{M}$.
 - 8: **for** $i = 1, \dots, N_p$ **do**
 - 9: $(d_i, f_i, \boldsymbol{\omega}_i) \leftarrow \Pi_{\mathcal{M}}(\mathbf{p}_i)$
 - 10: **if** $f_i \geq 0 \wedge \min(\boldsymbol{\omega}_i) > \tau$ **then**
 - 11: append i to $\mathcal{I}_{f_i}$
 - 12: **end if**
 - 13: **end for**
 - 14: **Aggregate & transfer.**
 - 15: **for** $j = 1, \dots, N_f$ with $\mathcal{I}_j \neq \emptyset$ **do**
 - 16: $\boldsymbol{\omega}_j \leftarrow \frac{1}{|\mathcal{I}_j|} \sum_{i \in \mathcal{I}_j} \boldsymbol{\omega}_i$, $\alpha_j \leftarrow \frac{1}{|\mathcal{I}_j|} \sum_{i \in \mathcal{I}_j} \alpha_i$
 - 17: (opt.) $\forall i \in \mathcal{I}_j : (\boldsymbol{\omega}_i, \alpha_i) \leftarrow (\boldsymbol{\omega}_j, \alpha_j)$
 - 18: **end for**
-

3.3 Parameters Decoupling

Rasterization. Following the same framework as IRGS [11], DR-GS employs a physically-based deferred rendering pipeline: Gaussians are first rasterized to generate per-pixel maps, after which the rendering equation is applied. Each Gaussian is augmented with a set of material parameters Φ_{mat} , which includes albedo $\mathbf{a} \in [0, 1]^3$, roughness $r \in [0, 1]$, and metallic $m \in [0, 1]$. Per-pixel attributes are aggregated via Gaussian rasterization:

$$\sum_{i=1}^N \gamma_i \{\mathbf{c}_i, d_i, \mathbf{n}_i, \mathbf{a}_i, r_i, m_i\}, \quad \text{where} \quad \gamma_i = \frac{T_i \alpha_i}{\sum_{k=1}^N T_k \alpha_k}, \quad T_i = \prod_{j=1}^{i-1} (1 - \alpha_j). \quad (2)$$

Here, $\mathbf{c}_i$ denotes the outgoing radiance, d_i represents depth, and $\mathbf{n}_i$ is the normal vector. Leveraging the resulting depth map, the 3D surface point $\mathbf{x}$ corresponding to each pixel can be computed.

Physically-based rendering. To achieve photorealistic rendering, we employ the complete rendering equation [23], which is the fundamental formulation simulating light transport in physically-based rendering (PBR):

$$L_o(\boldsymbol{\omega}_o, \mathbf{x}) = \int_{\Omega} f(\boldsymbol{\omega}_o, \boldsymbol{\omega}_i, \mathbf{x}) L_i(\boldsymbol{\omega}_i, \mathbf{x}) (\boldsymbol{\omega}_i \cdot \mathbf{n}) d\boldsymbol{\omega}_i, \quad (3)$$

where L_o and L_i denote outgoing and incident radiance at point $\mathbf{x}$ in direction $\boldsymbol{\omega}_o$ and $\boldsymbol{\omega}_i$, respectively. The integral is computed over the hemisphere Ω around the surface normal $\mathbf{n}$. The bidirectional reflectance distribution function (BRDF) f captures the material’s light scattering behavior.

We further decompose incident light at the surface point $\mathbf{x}$ into direct and indirect components:

$$L_i(\boldsymbol{\omega}_i, \mathbf{x}) = V(\boldsymbol{\omega}_i, \mathbf{x}) L_{\text{dir}}(\boldsymbol{\omega}_i) + L_{\text{ind}}(\boldsymbol{\omega}_i, \mathbf{x}), \quad (4)$$

where L_{dir} represents distant illumination from an environment map, while visibility V and indirect light L_{ind} are calculated using 2DGRT [11]. Note that L_{ind} is handled differently during reconstruction training and dynamic driving stages: during training, it is obtained by alpha-blending outgoing radiance $\mathbf{c}_i$ from Gaussians; the relight stage methodology is detailed in Sec. 3.4.

Given the incident radiance, we estimate the Eq. 3 via importance sampling [6]:

$$c_{\text{pbr}} = \frac{1}{N_r} \sum_{i=1}^{N_r} \frac{f(\boldsymbol{\omega}_o, \boldsymbol{\omega}_i, \mathbf{x}) L_i(\boldsymbol{\omega}_i, \mathbf{x}) (\boldsymbol{\omega}_i \cdot \mathbf{n})}{q(\boldsymbol{\omega}_i)}, \quad (5)$$

where N_r directions $\boldsymbol{\omega}_i$ are drawn from proposal distribution q with PDF $q(\boldsymbol{\omega}_i)$.

Training strategy. Accurate modeling of light-surface interactions through geometry-aware ray tracing requires robust scene geometry. Precise disentanglement of geometry, illumination, and material is essential for flexible editing of virtual assets. We jointly optimize pretrained Gaussians Θ and material parameters Φ_{mat} while estimating illumination. To reduce computational cost, we

selectively evaluate the Eq. 3 on a subset of pixels per view during training. The loss function is defined as:

$$\mathcal{L} = \mathcal{L}_c + \lambda_1^{\text{pbr}} \mathcal{L}_1^{\text{pbr}} + \lambda_{\text{light}} \mathcal{L}_{\text{light}}, \quad (6)$$

where $\mathcal{L}_c$ denotes the RGB reconstruction loss from 3DGS [24] for rendered radiance $\mathcal{C}$, $\mathcal{L}_1^{\text{pbr}}$ represents the L_1 loss between physically-based rendered results and ground truth, and $\mathcal{L}_{\text{light}}$ regularizes incident illumination to natural white balance.

3.4 Dynamic Driving

Particle-driven deformation. Particle representations provide superior flexibility for modeling complex geometries. Conventional mesh extraction often yields non-manifold, self-intersecting, or non-watertight meshes [43], complicating physical simulations. High-resolution meshing incurs significant computational cost, while low-resolution alternatives lose geometric fidelity. We therefore employ particles for spatial discretization and leverage the MPM for simulation.

Specifically, we generate a particle cloud $\mathcal{P}$ by populating the interior of an extracted mesh $\mathcal{M}$ and incorporating its vertices. This hybrid representation supports subsequent mesh-based ray tracing and deformation-aware processing. Physical simulation is performed, yielding a deformed particle set. We then compute the deformation gradient $\mathbf{F}_i$ and updated center position $\bar{\mathbf{p}}_i$ for each 2D Gaussian via GMLS interpolation (detailed in Alg. 2). The deformation gradient $\mathbf{F}_i$ is decomposed via polar decomposition into a rotation matrix $\bar{\mathbf{R}}_i$ and a scaling-shearing matrix $\bar{\mathbf{S}}_i$. We efficiently apply $\bar{\mathbf{R}}_i$ and $\bar{\mathbf{S}}_i$ to each Gaussian as follows:

$$\mathbf{p}'_i = \bar{\mathbf{p}}_i, \quad \mathbf{R}'_i = \bar{\mathbf{R}}_i \mathbf{R}_i, \quad \mathbf{S}'_i = \text{diag}(\bar{\mathbf{A}}_i) \cdot \mathbf{S}_i, \quad (7)$$

where

$$\bar{\mathbf{A}}_i = \begin{bmatrix} |\lambda_2(\bar{\mathbf{S}}_i)| & 0 \\ 0 & |\lambda_1(\bar{\mathbf{S}}_i)| \end{bmatrix}.$$

Mesh-driven deformation. Mesh-based representations enable efficient editing, sculpting, animation, and relighting operations. For each deformed triangle $f' = (\mathbf{v}'_a, \mathbf{v}'_b, \mathbf{v}'_c)$ in mesh $\mathcal{M}'$ and its corresponding face $f = (\mathbf{v}_a, \mathbf{v}_b, \mathbf{v}_c)$ in the canonical mesh $\mathcal{M}$, we compute a rotation matrix $\bar{\mathbf{R}}_i$, a shearing matrix $\bar{\mathbf{S}}_i$, and face-based displacement, which are directly applied to the bound Gaussians.

$$\begin{aligned} \Delta \mathbf{p} &= w_a(\mathbf{v}'_a - \mathbf{v}_a) + w_b(\mathbf{v}'_b - \mathbf{v}_b) + w_c(\mathbf{v}'_c - \mathbf{v}_c), \\ \bar{\mathbf{R}}_i &= w_a \bar{\mathbf{R}}_{v'_a} + w_b \bar{\mathbf{R}}_{v'_b} + w_c \bar{\mathbf{R}}_{v'_c}, \\ \bar{\mathbf{S}}_i &= w_a \bar{\mathbf{S}}_{v'_a} + w_b \bar{\mathbf{S}}_{v'_b} + w_c \bar{\mathbf{S}}_{v'_c}. \end{aligned} \quad (8)$$

The Gaussian center is updated as $\mathbf{p}'_i = \mathbf{p}_i + \Delta \mathbf{p}$, while the rotation and scale updates remain consistent with Eq. 7 and Eq. 8.

Rendering acceleration and optimization. During the dynamic driving phase, the original radiance values $\mathbf{c}_i$ are no longer valid due to changes in the

Algorithm 2 GMLS Interpolation

Input: Reference positions $\mathbf{X}_i \in \mathbb{R}^3$, deformed positions $\mathbf{x}_i \in \mathbb{R}^3$, deformation gradients $\mathbf{F}_i \in \mathbb{R}^{3 \times 3}$, target points $\mathbf{y}_j \in \mathbb{R}^3$, binding matrix $\mathcal{B}_{jk}$ (maps target j to source k), distance matrix $\mathcal{D}_{jk}$ **Output:** Interpolated positions $\mathbf{y}_j^{\text{interp}} \in \mathbb{R}^3$, interpolated deformation gradients $\mathbf{F}_j^{\text{interp}} \in \mathbb{R}^{3 \times 3}$

```

1: Initialize:  $\mathbf{y}_j^{\text{interp}} \leftarrow \mathbf{0}$ ,  $\mathbf{F}_j^{\text{interp}} \leftarrow \mathbf{I}$ 
2: for each batch  $\mathcal{J} \subseteq \{1, \dots, N_g\}$  do
3:    $\mathcal{N}_j \leftarrow \{k | \mathcal{B}_{jk} = 1\}, \forall j \in \mathcal{J}$ 
4:    $\mathbf{W}_j \leftarrow \exp(-\mathcal{D}_{jk}^2 / \sigma^2), k \in \mathcal{N}_j$ 
5:    $\Phi_k \leftarrow [1, \mathbf{X}_k^T]^T$ 
6:    $\mathbf{A}_j \leftarrow \sum_{k \in \mathcal{N}_j} \mathbf{W}_j^k \Phi_k \Phi_k^T + \epsilon \mathbf{I}$ 
7:    $\mathbf{b}_j \leftarrow \sum_{k \in \mathcal{N}_j} \mathbf{W}_j^k \Phi_k \mathbf{x}_k^T$ 
8:    $\alpha_j \leftarrow \mathbf{A}_j^{-1} \mathbf{b}_j$ 
9:    $\mathbf{y}_j^{\text{interp}} \leftarrow \Phi(\mathbf{0})^T \alpha_j$ 
10:   $\mathbf{F}_j^{\text{interp}} \leftarrow \sum_{k \in \mathcal{N}_j} \mathbf{W}_j^k \Phi_k \mathbf{F}_k \mathbf{A}_j^{-1} \Phi(\mathbf{0})$ 
11: end for

```

illumination conditions from those in the static reconstruction stage. We need to aggregate material attributes via Gaussian ray tracing and estimate incident radiance. To address the computational bottleneck of ray tracing, we replace 2DGS-based tracing with mesh-based ray tracing in this stage. Material properties learned through physical-based parameter decoupling (detailed in Sec. 3.3) are stored on a triangular mesh extracted via TSDF, enabling accelerated attribute lookup at the first ray intersection.

To further improve ray sampling efficiency, we employ multiple importance sampling (MIS) [48], combining cosine-weighted, GGX, and environmental distributions to model diffuse, specular, and environmental lighting, respectively. The Monte Carlo estimator with the balance heuristic is defined as:

$$\sum_{i=1}^n \frac{1}{n_i} \sum_{j=1}^{n_i} w_i(X_{i,j}) \frac{g(X_{i,j})}{p_i(X_{i,j})}, \quad w_i(x) = \frac{n_i p_i(x)}{\sum_k n_k p_k(x)}. \quad (9)$$

To mitigate inherent Monte Carlo noise, we incorporate a cross-bilateral filter based on spatiotemporal variance-guided filtering (SVGF) [42], which preserves geometric edges through depth- and normal-aware weighting.

4 Experiment

4.1 Experiment Setup

Datasets and metrics. To evaluate the proposed method, we conducted experiments on the widely-used GlossySynthetic dataset [31], TensoIR dataset [22] and character models from Sketchfab and Mixamo (including Vegeta, Mutant, NotEnrique). For quantitative assessment of static reconstruction and relighting

quality, we employed three metrics: PSNR, SSIM [50], and LPIPS [59]. Since ground truth images are unavailable after deformation, quantitative evaluation is infeasible. We therefore conduct a user study with three perceptual criteria: (i) Physical plausibility (PP), assessing whether light transport appears physically realistic; (ii) Temporal coherence (TC), assessing flickering and temporal instability in dynamic sequences; and (iii) Deformation consistency (DC), assessing whether local appearance details remain consistent before and after deformation. We invite 30 participants to rate each criterion on a 0-5 scale and report the mean scores. The corresponding videos are provided in the project page.

Baselines. We extensively compare with dynamic Gaussian splatting baselines (*PhysGaussian* [53], *GSP* [7], *SuGaR* [13], *Mani-GS* [10], *GaussianMesh* [9]) and physics-based relightable reconstruction methods (*GSP* [7], *IRGS* [11], *Ref-Gaussian* [56], *GS-ROR²* [64]). Notably, GSP serves as our primary baseline for simultaneously supporting deformation and relighting. For fair comparison, all methods are implemented using identical input data.

Implementation details. We first strictly adhere to the original configuration of Ref-Gaussian [56] for pre-training, followed by an extended fine-tuning phase of 20,000 iterations. For MIS, 512 rays are sampled during reconstruction (256 cosine-weighted, 128 GGX, and 128 light samples), while the number is reduced to 32 rays (16 cosine-weighted, 8 GGX, and 8 light samples) for dynamic driving and relighting stage. Environment maps are set to a resolution of 128×256 .

4.2 Evaluations and Comparisons

Qualitative comparison. Rendering results under geometric deformation are shown in Fig. 3. PhysGaussian bakes environment reflections into static textures, causing severe artifacts on *Angel* under motion, while GSP’s inaccurate surface reconstruction fails on glossy objects. In contrast, DR-GS preserves crisp, consistent metallic appearance even under large deformations.

As further shown in Fig. 4 and Fig. 6, GSP suffers from surface irregularities and holes that are amplified by deformation, leading to catastrophic relighting. DR-GS instead reconstructs smooth surfaces and remains physically plausible under joint deformation and illumination changes.

Under novel illumination (Fig. 7), Ref-Gaussian produces mottled relighting due to insufficient material–lighting disentanglement, whereas DR-GS stays spatially coherent. IRGS yields implausibly dim glossy relighting (Fig. 5) due to missing importance sampling and metallic modeling.

Quantitative comparison. Table 1 and Table 2 summarize quantitative results on the GlossySynthetic dataset, where darker red indicates better performance. Table 1 compares per-scene static reconstruction quality across dynamic methods, while Table 2 evaluates relighting quality and additionally reports efficiency in terms of training time and rendering speed. Overall, our method achieves the best or competitive reconstruction and relighting performance across most scenes, while maintaining favorable computational efficiency.

Due to the lack of ground-truth after dynamic deformation, we evaluate visual quality via a user study (Table 3). Our method achieves the highest ratings across all metrics, and remains superior under relighting. GSP degrades noticeably after relighting, while PhysGaussian does not support relighting and is therefore marked as N/A.

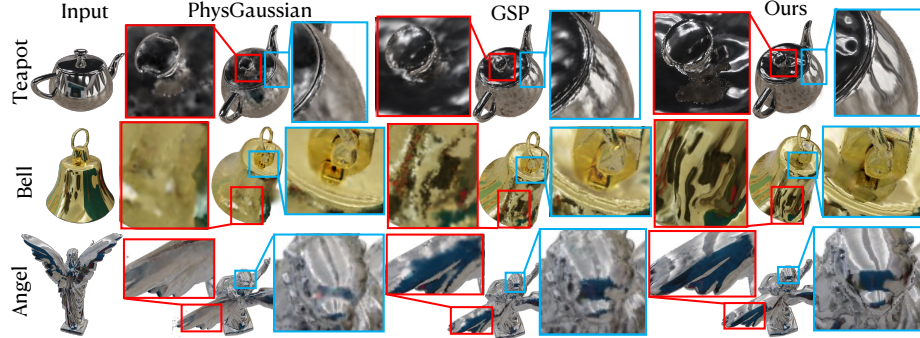


Fig. 3: Qualitative comparisons on deformed reflective scenes. Compared with PhysGaussian [53] and GSP [7], DR-GS yields the clearest, most physically plausible glossy reflections under large deformations.

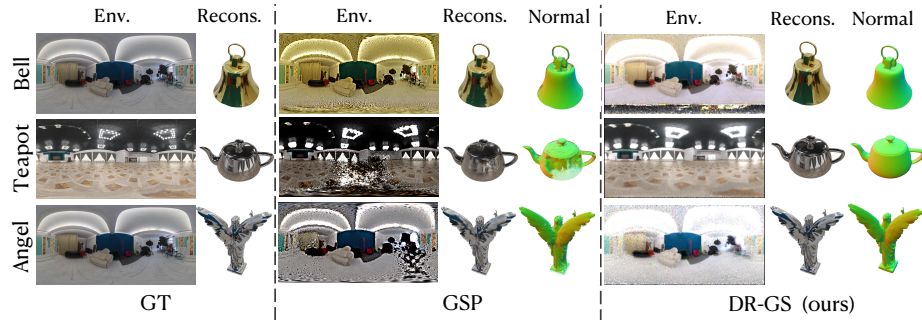


Fig. 4: Qualitative comparison of illumination decomposition and reconstruction. Compared with GSP, DR-GS better decouples environment lighting and yields clearer, more photorealistic novel-view renderings.

4.3 Ablations

Parameter editing. DR-GS’s fully decoupled parameters enable flexible post-reconstruction editing of digital assets. Fig. 8 and Fig. 9 showcase edits for

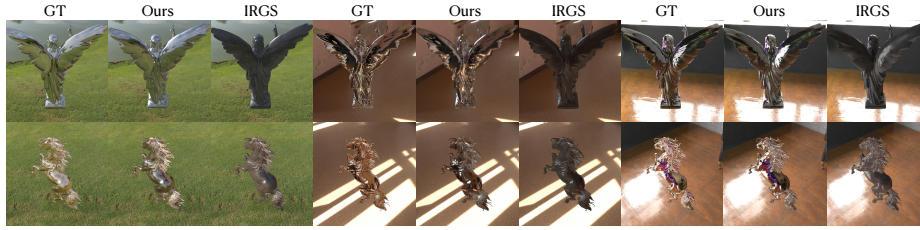


Fig. 5: Relighting results on glossy objects. DR-GS more faithfully matches the GT specular highlights and reflection details while preserving material consistency, whereas IRGS often appears darker with weakened reflections and lost fine details.

simulation and rendering: the former demonstrates gravity-driven soft-body deformation, where parameter changes markedly affect the Angel’s wings and the Horse’s legs (red boxes), while the latter shows that joint PBR edits produce diverse and photorealistic appearances.

Gaussian-based inter-reflection. Reflective surfaces can generate indirect illumination through their own geometric structures, leading to inter-reflection effects. As shown in Fig. 11, DR-GS incorporates ray-traced visibility to synthesize plausible indirect illumination under novel views, achieving physically faithful modeling of inter-reflection.

Table 1: Per-scene static reconstruction quality comparison on GlossySynthetic [31] dataset across dynamic methods. The intensity of the red color signifies a better result. (P) and (M) denote particle-driven and mesh-driven variants, respectively.

Scenes	angel			bell			cat			horse		
Metrics	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
PhysGaussian(P)	25.38	0.801	0.092	24.31	0.904	0.122	29.48	0.947	0.085	24.49	0.790	0.087
GSP(P)	25.70	0.916	0.083	29.26	0.940	0.090	30.82	0.955	0.068	25.90	0.926	0.064
SuGaR(M)	12.81	0.775	0.202	12.40	0.740	0.240	12.51	0.730	0.203	14.31	0.748	0.228
Mani-GS(M)	26.39	0.911	0.075	24.58	0.901	0.114	29.90	0.950	0.072	24.76	0.897	0.075
GaussianMesh(M)	25.84	0.901	0.079	24.02	0.890	0.118	28.39	0.936	0.083	23.70	0.886	0.081
ours(P)	29.64	0.943	0.052	31.94	0.962	0.056	31.25	0.965	0.052	25.54	0.929	0.055
ours(M)	25.04	0.887	0.094	28.12	0.917	0.103	27.07	0.931	0.091	21.25	0.851	0.111
Scenes	luyu			potion			tbell			teapot		
Metrics	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
PhysGaussian(P)	26.22	0.904	0.086	28.71	0.923	0.132	22.76	0.893	0.153	21.14	0.877	0.122
GSP(P)	27.19	0.916	0.073	28.41	0.932	0.113	24.52	0.910	0.131	22.67	0.898	0.108
SuGaR(M)	15.15	0.754	0.216	14.68	0.613	0.254	12.77	0.449	0.434	13.62	0.744	0.194
Mani-GS(M)	26.42	0.909	0.069	27.83	0.922	0.110	21.38	0.885	0.145	21.18	0.878	0.106
GaussianMesh(M)	26.02	0.905	0.071	27.53	0.918	0.112	20.55	0.871	0.156	20.64	0.871	0.109
ours(P)	28.57	0.940	0.053	30.55	0.941	0.095	27.54	0.942	0.094	26.72	0.945	0.062
ours(M)	24.10	0.873	0.098	26.75	0.887	0.145	23.67	0.879	0.165	22.93	0.882	0.107

Robustness to extracted mesh. Fig. 12 demonstrates that introducing learnable Gaussian attributes, i.e. normal offset d and opacity α , effectively enhances adaptability and robustness to inaccuracies in the extracted mesh. Without learnable d , the result exhibits noticeable spiky artifacts and structural blur; without learnable α , scattered stain-like noise and localized crack-like distortions appear. In contrast, our full model achieves high-fidelity, clean, and detail-preserving rendering.

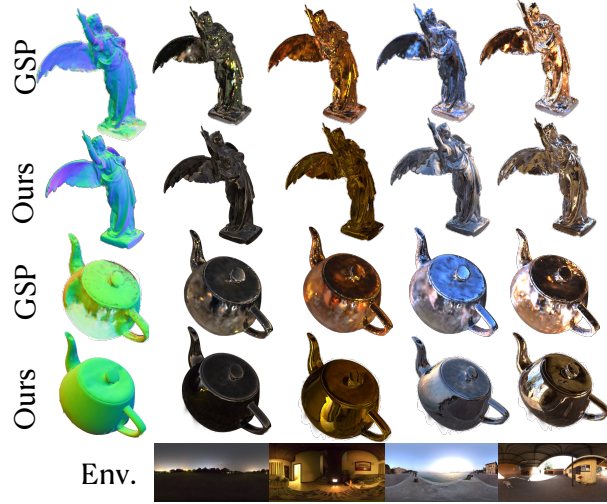


Fig. 6: Deformation results under relighting. Under combined deformation and lighting changes, DR-GS yields smoother normals and physically plausible reflections.

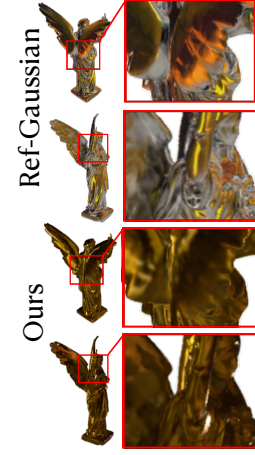


Fig. 7: Relighting under dim illumination. Consistent metallic sheen suggests DR-GS learns a more accurate base color.

Table 2: Quantitative relighting comparison on the GlossySynthetic [31] dataset. The intensity of the red color signifies a better result.

Scenes	angel		bell		cat		horse		Training time
Metrics	PSNR $\uparrow$	SSIM $\uparrow$	PSNR $\uparrow$	SSIM $\uparrow$	PSNR $\uparrow$	SSIM $\uparrow$	PSNR $\uparrow$	SSIM $\uparrow$	Hour
GS-ROR ²	20.81	0.878	24.49	0.927	26.28	0.942	23.31	0.938	1.5
Ref-Gaussian	21.39	0.900	22.90	0.919	20.54	0.912	24.97	0.944	0.58
IRGS	20.58	0.860	20.98	0.878	22.43	0.888	22.10	0.921	1.0
ours	24.15	0.905	25.81	0.931	27.34	0.940	24.19	0.939	0.68
Scenes	luyu		potion		tbell		teapot		Render time
Metrics	PSNR $\uparrow$	SSIM $\uparrow$	PSNR $\uparrow$	SSIM $\uparrow$	PSNR $\uparrow$	SSIM $\uparrow$	PSNR $\uparrow$	SSIM $\uparrow$	FPS
GS-ROR ²	22.61	0.899	25.67	0.918	22.80	0.918	21.17	0.893	122
Ref-Gaussian	19.74	0.875	20.06	0.868	20.74	0.904	21.78	0.924	208
IRGS	22.73	0.852	22.92	0.866	19.97	0.853	19.27	0.870	0.5
ours	25.16	0.889	27.35	0.923	22.12	0.884	23.08	0.925	24

Table 3: User study.

Metrics	PhysGaussian	PhysGaussian(relight)	GSP	GSP (relight)	ours	ours (relight)
PP $\uparrow$	3.26	N/A	4.38	3.82	4.56	4.62
TC $\uparrow$	3.87	N/A	4.12	3.56	4.42	4.27
DC $\uparrow$	2.89	N/A	3.76	3.65	4.19	4.21

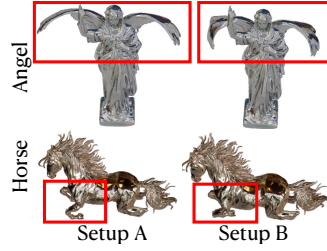


Fig. 8: Simulated deformation comparison. The higher Young’s modulus in Setup A diminished its elastic deformability compared to Setup B.

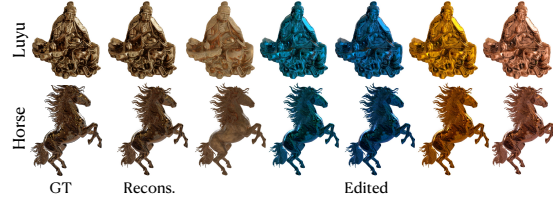


Fig. 9: PBR results with material parameter editing. We present a comparison between DR-GS static reconstruction results and ground truth, along with rendered results after editing the decoupled material parameters.



Fig. 10: More results. It demonstrates reconstruction, geometric deformation, and PBR under various illumination conditions using estimated material parameters.

Shadow and reflection. Fig. 14 shows that DR-GS correctly models occlusion relationships induced by geometric visibility. The resulting shadows and reflections vary coherently with occlusions, yielding physically plausible occlusion-aware shading and reflections.

Efficiency analysis. Fig. 13 shows that the denoiser, MIS, and mesh acceleration substantially reduce rendering time and improve/stabilize PSNR under the same sampling budget, leading to a better quality-efficiency trade-off.

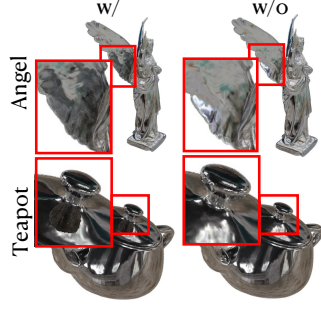


Fig. 11: Ablation on modeling of inter-reflection.

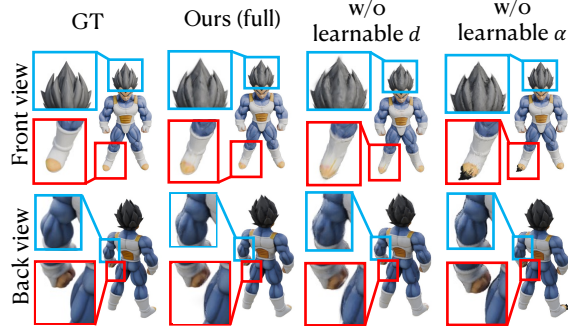


Fig. 12: Ablation on learnable parameters.

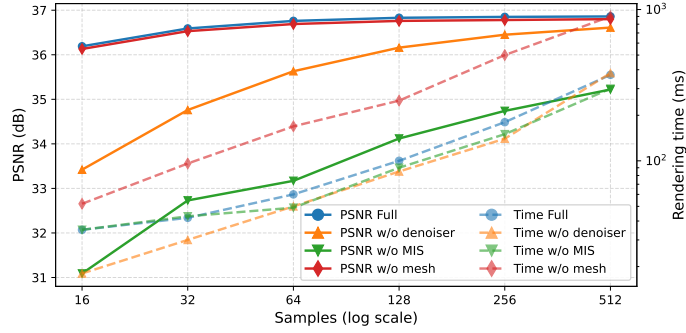


Fig. 13: Effect of denoiser, MIS, and mesh acceleration on quality and run-time. It plots PSNR (left axis) and rendering time (right axis, log scale) versus the number of samples for the full method and ablated variants on the Armadillo scene.

More results. To assess the generalization of DR-GS, Fig. 10 showcases diverse geometric deformations and relighting outcomes under various illumination conditions. These include both bright and dim settings across indoor and outdoor environments, utilizing particle-driven and mesh-driven methods. Further results are available in the supplementary video.



Fig. 14: Composed scenes. It demonstrates DR-GS’s ability to model occlusion-induced shadows and reflections arising from geometric visibility effects.

Complex non-manifold geometry.

Unlike mesh-based methods, GS-based approaches better handle non-manifold topologies. On the TensoIR [22] (ficus) and Shelly [51] (khady, kitten, pug, wolly) datasets, covering foliage, hair, fur, and porous structures, DR-GS achieves high-fidelity NVS (Fig. 15) where mesh-based methods often fail.



Fig. 15: Results of non-manifold geometry. It demonstrates DR-GS’s ability to deal with intricate structures.

5 Conclusion

This paper presents DR-GS, a unified Gaussian framework for physically-based inverse rendering, relighting, and deformation-aware manipulation. By decoupling geometry, illumination, and materials, DR-GS addresses the unrealistic appearance of snapshots-based methods under geometric and lighting variations, while enabling flexible parameter control. We incorporate MIS and low-sample Monte Carlo estimation to achieve efficient dynamic rendering with high physical fidelity, particularly on glossy surfaces. Supporting both particle- and mesh-based deformation, DR-GS facilitates physical simulation and 3D animation, offering a new pathway for virtual content creation.

Limitation. While DR-GS eliminates baked-illumination artifacts in dynamic scenes, it introduces a trade-off: solving the rendering equation inherently reduces both training and rendering efficiency compared to vanilla 3DGS.

Acknowledgements

This work was supported in part by New Generation Artificial Intelligence-National Science and Technology Major Project (2025ZD0123004), Ningbo grant (2025Z038) and National Natural Science Foundation of China (Grant No.62376060).

References

1. Boss, M., Braun, R., Jampani, V., Barron, J.T., Liu, C., Lensch, H.P.: Nerd: Neural reflectance decomposition from image collections. In: ICCV (2021) [3](#)
2. Boss, M., Engelhardt, A., Kar, A., Li, Y., Sun, D., Barron, J.T., Lensch, H.P.A., Jampani, V.: Samurai: Shape and material from unconstrained real-world arbitrary image collections. In: NeurIPS (2022) [3](#)
3. Cao, J., Guan, S., Ge, Y., Li, W., Yang, X., Ma, C.: NeuMA: Neural material adaptor for visual grounding of intrinsic dynamics. In: NeurIPS (2024) [3](#)
4. Chen, H., Lin, Z., Zhang, J.: Gi-gs: Global illumination decomposition on gaussian splatting for inverse rendering. In: ICLR (2025) [4](#)
5. Chen, Y., Gu, C., Jiang, J., Zhu, X., Zhang, L.: Periodic vibration gaussian: Dynamic urban scene reconstruction and real-time rendering. arXiv:2311.18561 (2023) [3](#)
6. Cook, R.L., Torrance, K.E.: A reflectance model for computer graphics. ACM TOG (1982) [6](#)
7. Feng, Y., Feng, X., Shang, Y., Jiang, Y., Yu, C., Zong, Z., Shao, T., Wu, H., Zhou, K., Jiang, C., Yang, Y.: Gaussian splashing: Unified particles for versatile motion synthesis and rendering. In: CVPR (2025) [2](#), [3](#), [9](#), [10](#)
8. Gao, J., Gu, C., Lin, Y., Zhu, H., Cao, X., Zhang, L., Yao, Y.: Relightable 3d gaussian: Real-time point cloud relighting with brdf decomposition and ray tracing. In: ECCV (2024) [4](#)
9. Gao, L., Yang, J., Zhang, B., Sun, J., Yuan, Y., Fu, H., Lai, Y.K.: Real-time large-scale deformation of gaussian splatting. ACM TOG (2024) [2](#), [3](#), [5](#), [9](#)
10. Gao, X., Li, X., Zhuang, Y., Zhang, Q., Hu, W., Zhang, C., Yao, Y., Shan, Y., Quan, L.: Mani-gs: Gaussian splatting manipulation with triangular mesh. In: CVPR (2025) [2](#), [3](#), [9](#)
11. Gu, C., Wei, X., Zeng, Z., Yao, Y., Zhang, L.: Irgs: Inter-reflective gaussian splatting with 2d gaussian ray tracing. In: CVPR (2025) [2](#), [4](#), [5](#), [6](#), [9](#)
12. Gu, C., Wei, X., Zhang, L., Zhu, X.: Tensorflow: Tensorial flow-based sampler for inverse rendering. In: CVPR (2025) [3](#)
13. Guédon, A., Lepetit, V.: Sugar: Surface-aligned gaussian splatting for efficient 3d mesh reconstruction and high-quality mesh rendering. In: CVPR (2024) [2](#), [3](#), [9](#)
14. Guo, M., Chiang, M.J.Y., Santesteban, I., Sarafianos, N., Chen, H.y., Halimi, O., Božič, A., Saito, S., Wu, J., Liu, C.K., Stuyck, T., Larionov, E.: Pgc: Physics-based gaussian cloth from a single pose. In: CVPR (2025) [2](#)
15. Guo, Y., Bai, Y., Hu, L., Guo, Z., Liu, M., Cai, Y., Huang, T., Ma, L.: Prtgs: Pre-computed radiance transfer of gaussian splats for real-time high-quality relighting. In: MM (2024) [4](#)
16. Heitz, E.: Sampling the ggx distribution of visible normals. JCGT (2018) [2](#)
17. Hu, L., Zhang, H., Zhang, Y., Zhou, B., Liu, B., Zhang, S., Nie, L.: Gaussianavatar: Towards realistic human avatar modeling from a single video via animatable 3d gaussians. In: CVPR (2024) [2](#), [3](#)
18. Huang, B., Yu, Z., Chen, A., Geiger, A., Gao, S.: 2d gaussian splatting for geometrically accurate radiance fields. In: SIGGRAPH (2024) [4](#)
19. Huang, T., Zeng, Y., Li, H., Zuo, W., Lau, R.W.: Dreamphysics: Learning physical properties of dynamic 3d gaussians with video diffusion priors. In: AAAI (2025) [2](#), [3](#)
20. Ji, M., Qiu, R.Z., Zou, X., Wang, X.: Graspsplats: Efficient manipulation with 3d feature splatting. In: CoRL (2024) [2](#)

21. Jiang, Y., Yu, C., Xie, T., Li, X., Feng, Y., Wang, H., Li, M., Lau, H., Gao, F., Yang, Y., Jiang, C.: Vr-gs: A physical dynamics-aware interactive gaussian splatting system in virtual reality. In: SIGGRAPH (2024) [2](#), [3](#)
22. Jin, H., Liu, I., Xu, P., Zhang, X., Han, S., Bi, S., Zhou, X., Xu, Z., Su, H.: Tensor: Tensorial inverse rendering. In: CVPR (2023) [8](#), [15](#)
23. Kajiya, J.T.: The rendering equation. In: Proceedings of the 13th Annual Conference on Computer Graphics and Interactive Techniques (1986) [6](#)
24. Kerbl, B., Kopanas, G., Leimkühler, T., Drettakis, G.: 3d gaussian splatting for real-time radiance field rendering. ACM TOG (2023) [3](#), [4](#), [7](#)
25. Lai, S., Huang, L., Guo, J., Cheng, K., Pan, B., Long, X., Lyu, J., Lv, C., Guo, Y.: Glossygs: Inverse rendering of glossy objects with 3d gaussian splatting. TVCG (2025) [4](#)
26. Li, M., Yao, S., Kai, C., Xie, Z., Chen, K., Jiang, Y.G.: Anigaussian: Animatable gaussian avatar with pose-guided deformation. ArXiv (2025) [2](#), [3](#)
27. Li, Z., Chen, Z., Li, Z., Xu, Y.: Spacetime gaussian feature splatting for real-time dynamic view synthesis. In: CVPR (2024) [3](#)
28. Li, Z., Yu, H.X., Liu, W., Yang, Y., Herrmann, C., Wetzstein, G., Wu, J.: Wonderplay: Dynamic 3d scene generation from a single image and actions. In: ICCV (2025) [2](#)
29. Liang, Z., Zhang, Q., Feng, Y., Shan, Y., Jia, K.: Gs-ir: 3d gaussian splatting for inverse rendering. In: CVPR (2024) [4](#)
30. Liu, F., Wang, H., Yao, S., Zhang, S., Zhou, J., Duan, Y.: Physics3d: Learning physical properties of 3d gaussians via video diffusion. arXiv preprint arXiv:2406.04338 (2024) [2](#), [3](#)
31. Liu, Y., Wang, P., Lin, C., Long, X., Wang, J., Liu, L., Komura, T., Wang, W.: Nero: Neural geometry and brdf reconstruction of reflective objects from multiview images. ACM Transactions on Graphics (ToG) (2023) [3](#), [8](#), [11](#), [12](#)
32. Lu, G., Zhang, S., Wang, Z., Liu, C., Lu, J., Tang, Y.: Manigaussian: Dynamic gaussian splatting for multi-task robotic manipulation. In: ECCV (2024) [2](#)
33. Luo, Z., Cui, Z., Luo, S., Chu, M., Li, M.: Vr-doh: Hands-on 3d modeling in virtual reality. ACM TOG (2025) [2](#), [3](#)
34. Ma, Z., Jiang, J., Chen, Y., Zhang, L.: Béziergs: Dynamic urban scene reconstruction with bézier curve gaussian splatting. In: ICCV (2025) [3](#)
35. Martin, S., Kaufmann, P., Botsch, M., Grinspun, E., Gross, M.: Unified simulation of elastic rods, shells, and solids. ACM Transactions on Graphics (TOG) (2010) [2](#)
36. Mildenhall, B., Srinivasan, P.P., Tancik, M., Barron, J.T., Ramamoorthi, R., Ng, R.: Nerf: Representing scenes as neural radiance fields for view synthesis. Communications of the ACM (2021) [2](#)
37. Moenne-Loccoz, N., Mirzaei, A., Perel, O., De Lutio, R., Martinez Esturo, J., State, G., Fidler, S., Sharp, N., Gojcic, Z.: 3d gaussian ray tracing: Fast tracing of particle scenes. ACM Transactions on Graphics (TOG) **43**(6), 1–19 (2024) [4](#)
38. Parker, S.G., Bigler, J., Dietrich, A., Friedrich, H., Hoberock, J., Luebke, D., McAllister, D., McGuire, M., Morley, K., Robison, A., Stich, M.: Optix: a general purpose ray tracing engine. ACM TOG (2010) [4](#)
39. Pharr, M., Jakob, W., Humphreys, G.: Physically Based Rendering: From Theory to Implementation. San Francisco, CA, USA, 3rd edn. (2016) [2](#)
40. Qian, S., Kirschstein, T., Schoneveld, L., Davoli, D., Giebenhain, S., Nießner, M.: Gaussianavatars: Photorealistic head avatars with rigged 3d gaussians. In: CVPR (2024) [2](#), [3](#)
41. Qiu, R.Z., Yang, G., Zeng, W., Wang, X.: Language-driven physics-based scene synthesis and editing via feature splatting. In: ECCV (2024) [3](#)

42. Schied, C., Kaplanyan, A., Wyman, C., Patney, A., Chaitanya, C.R.A., Burgess, J., Liu, S., Dachsbacher, C., Lefohn, A., Salvi, M.: Spatiotemporal variance-guided filtering: real-time reconstruction for path-traced global illumination. In: *Proceedings of High Performance Graphics* (2017) 2, 8
43. Shen, T., Munkberg, J., Hasselgren, J., Yin, K., Wang, Z., Chen, W., Gojcic, Z., Fidler, S., Sharp, N., Gao, J.: Flexible isosurface extraction for gradient-based mesh optimization. *ACM TOG* (2023) 7
44. Shi, Y., Wu, Y., Wu, C., Liu, X., Zhao, C., Feng, H., Zhang, J., Zhou, B., Ding, E., Wang, J.: Gir: 3d gaussian inverse rendering for relightable scene factorization. *T-PAMI* (2025) 4
45. Shorinwa, O., Tucker, J., Smith, A., Swann, A., Chen, T., Firoozi, R., Kennedy, M.D., Schwager, M.: Splat-mover: Multi-stage, open-vocabulary robotic manipulation via editable gaussian splatting. In: *8th Annual Conference on Robot Learning* (2024) 2
46. Srinivasan, P.P., Deng, B., Zhang, X., Tancik, M., Mildenhall, B., Barron, J.T.: Nerv: Neural reflectance and visibility fields for relighting and view synthesis. In: *CVPR* (2021) 3
47. Sun, H., Gao, Y., Xie, J., Yang, J., Wang, B.: Svg-ir: Spatially-varying gaussian splatting for inverse rendering. In: *CVPR* (2025) 4
48. Veach, E., Guibas, L.J.: Optimally combining sampling techniques for monte carlo rendering. In: *Proceedings of the 22nd Annual Conference on Computer Graphics and Interactive Techniques* (1995) 8
49. Waczyńska, J., Borycki, P., Tadeja, S., Tabor, J., Spurek, P.: Games: Mesh-based adapting and modification of gaussian splatting. *arXiv preprint arXiv:2402.01459* (2024) 3
50. Wang, Z., Bovik, A., Sheikh, H., Simoncelli, E.: Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing* (2004) 9
51. Wang, Z., Shen, T., Nimier-David, M., Sharp, N., Gao, J., Keller, A., Fidler, S., Müller, T., Gojcic, Z.: Adaptive shells for efficient neural radiance field rendering. *ACM Trans. Graph.* **42**(6) (2023). <https://doi.org/10.1145/3618390>, <https://doi.org/10.1145/3618390> 15
52. Wu, G., Yi, T., Fang, J., Xie, L., Zhang, X., Wei, W., Liu, W., Tian, Q., Wang, X.: 4d gaussian splatting for real-time dynamic scene rendering. In: *CVPR* (2024) 3
53. Xie, T., Zong, Z., Qiu, Y., Li, X., Feng, Y., Yang, Y., Jiang, C.: Physgaussian: Physics-integrated 3d gaussians for generative dynamics. In: *CVPR* (2024) 2, 3, 9, 10
54. Yang, Z., Yang, H., Pan, Z., Zhang, L.: Real-time photorealistic dynamic scene representation and rendering with 4d gaussian splatting. In: *ICLR* (2024) 3
55. Yang, Z., Gao, X., Zhou, W., Jiao, S., Zhang, Y., Jin, X.: Deformable 3d gaussians for high-fidelity monocular dynamic scene reconstruction. In: *CVPR* (2024) 3
56. Yao, Y., Zeng, Z., Gu, C., Zhu, X., Zhang, L.: Reflective gaussian splatting. In: *ICLR* (2025) 4, 5, 9
57. Yu, J., Fu, L., Huang, H., El-Refai, K., Ambrus, R.A., Cheng, R., Irshad, M.Z., Goldberg, K.: Real2render2real: Scaling robot data without dynamics simulation or robot hardware. *arXiv preprint arXiv:2505.09601* (2025) 2
58. Zhang, J., Yao, Y., Li, S., Liu, J., Fang, T., McKinnon, D., Tsin, Y., Quan, L.: Neilf++: Inter-reflectable light fields for geometry and material estimation. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. pp. 3601–3610 (2023) 3

59. Zhang, R., Isola, P., Efros, A.A., Shechtman, E., Wang, O.: The unreasonable effectiveness of deep features as a perceptual metric. In: CVPR (2018) [9](#)
60. Zhang, T., Yu, H.X., Wu, R., Feng, B.Y., Zheng, C., Snavely, N., Wu, J., Freeman, W.T.: PhysDreamer: Physics-based interaction with 3d objects via video generation. In: ECCV (2024) [3](#)
61. Zhao, H., Wang, H., Zhao, X., Fei, H., Wang, H., Long, C., Zou, H.: Efficient physics simulation for 3d scenes via mllm-guided gaussian splatting. arXiv preprint arXiv:2411.12789 (2024) [3](#)
62. Zheng, Y., Chen, X., Zheng, Y., Gu, S., Yang, R., Jin, B., Li, P., Zhong, C., Wang, Z., Liu, L., et al.: Gaussiangrasper: 3d language gaussian splatting for open-vocabulary robotic grasping. IEEE Robotics and Automation Letters (2024) [2](#)
63. Zhu, T., Chen, Z., Gao, J., Yan, Y., Yang, X.: Multi-times monte carlo rendering for inter-reflection reconstruction. NeurIPS (2024) [3](#)
64. Zhu, Z.L., Wang, B., Yang, J.: Gs-ror²: Bidirectional-guided 3dgs and sdf for reflective object relighting and reconstruction. In: ACM TOG (2025) [9](#)