# BiGS: Bidirectional Primitives for Relightable 3D Gaussian Splatting

Liu Zhenyuan[1,2], Yu Guo[3], Xinyuan Li[1], Bernd Bickel[2], Ran Zhang[1]

[1]Tencent PCG, New York, USA
[2]ETH Zurich, Zurich, Switzerland
[3]George Mason University, Fairfax, USA

zhenyliu@ethz.ch, tflsguoyu@gmail.com, xinyuanli@global.tencent.com, bickelb@ethz.ch, ranorizhang@global.tencent.com

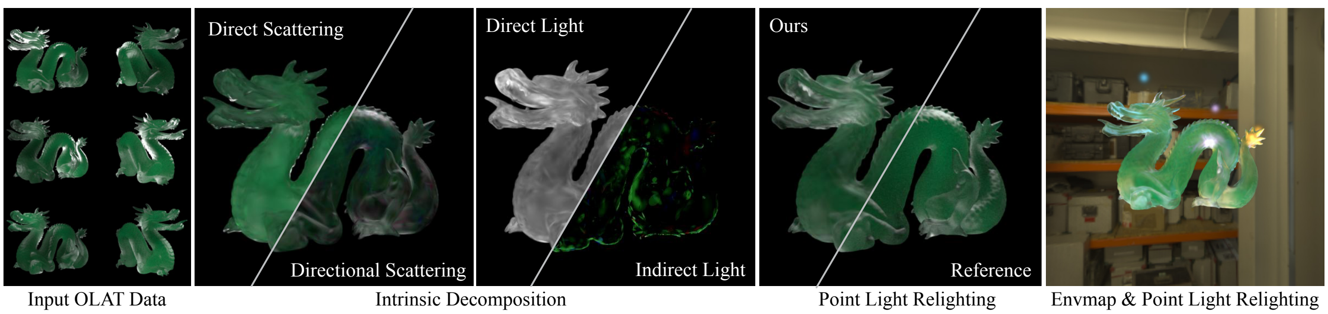


Figure 1. BiGS reconstructs view- and light-dependent color functions for Gaussian Splats using OLAT data such as the shown translucent DRAGON of 31,252 bidirectional Gaussian primitives. Our model decomposes the appearance of each primitive into different intrinsic components and is able to achieve plausible relighting and novel view synthesis with various environment maps and point light sources.

## Abstract

*We present BiGS, an image-based novel view synthesis technique designed to model and render 3D objects with surface and volumetric materials under dynamic illumination, achieving real-time relighting of 3D objects with the rasterization algorithm of Gaussian Splatting. Our method represents light- and view-dependent scattering via bidirectional spherical harmonics that does not use a specific surface normal-related reflectance function, making it more compatible with volumetric representations like Gaussian splatting, where the normals are undefined. We demonstrate our method by reconstructing and rendering objects with complex materials on synthetic and captured One-Light-At-a-Time (OLAT) datasets, showcasing various photorealistic appearances our method captures and the real-time performance.*

## 1. Introduction

Capturing and rendering 3D content from images has been a long-standing research topic in computer graphics and computer vision, with a wide range of applications in virtual production, video games, architecture, and mixed reality.

Recently, 3D Gaussian Splatting (3DGS) [10] has emerged as a novel 3D representation that excels in these areas, delivering both a high level of photo-realism and real-time performance. However, the spherical harmonics based appearance model used in 3DGS is only capable of synthesizing novel views under the static illumination in the capture data, unable to adapt to new lighting conditions. This limits the usage of 3DGS in interactive applications like video games and mixed reality, where dynamic lighting plays an essential role.

Recent research [4, 9] has attempted to address this limitation by incorporating surface-based shading models into the framework of Gaussian Splatting, describing how light interacts with Gaussian primitives. These models can work well for objects with surface-based materials, such as smooth or rough solid objects. However, they tend to struggle with fuzzy objects that lack a clear surface definition, like fur and hair, or those with volumetric appearances that are not solely determined by their surface. For example, subsurface scattering in translucent objects presents significant challenges for surface-based models.

In this paper, we introduce a lighting-dependent appearance model for Gaussian primitives based on bidirectional spherical harmonics. As shown in Fig. 1, our approach enables the rendering of 3D objects with various materials under dynamic illumination, including both near-field and environmental lighting. Our unified representation does not make assumption of the materials of the objects, therefore enabling the modeling of both surface-based and volumetric appearance. Our contributions are:

- A novel formulation of relightable Gaussian primitives accounting for both surface and volume appearances.
- Bidirectional spherical harmonics for representing the scattering function for Gaussian primitives.
- An optimization method for obtaining relightable Gaussian primitives by performing intrinsic light decomposition on the OLAT data.

## 2. Related Work

***Relightable Gaussian Splatting*** Recently, Gaussian Splatting has become one of the most popular techniques in novel view synthesis. It represents scenes using 3D Gaussian volumetric primitives and reaches high-level photorealism and real-time rendering performance via fast rasterization. One major limitation of Gaussian Splatting is that the illumination in the training data is baked into the model, making it challenging to render the scene under novel lighting conditions. Saito et al. [15] decomposes the color of the Gaussians into various components using multi-layer perceptrons (MLPs) for modeling human faces. Gao et al. [4] proposes to derive normal maps from depth fields, and then extract shading properties using physics-based rendering. GaussianShader [9] estimates the normals of the Guassian primitives using their shapes and proposes a shading model for reflective surfaces. GS-Phong [6] also assumes fully opaque Gaussians and applies the Blinn-Phong shading model to compute diffuse and specular colors. These methods all adopt surface-based appearance models, and our method sticks with a general lighting formulation that can represent both surface and volumetric effects, hence without the need to drive all the Gaussian primitives to be completely opaque or rely on a specific shading model, yielding greater expressivity. While there are also some works [3, 35] using Gaussian as primitives for volumetric ray tracing, reaching outstanding photorealism at a greater computational cost, our method is rasterization-based, without compromising real-time rendering speed.

***Relighting of Neural Implicit Representation***. Neural radiance field (NeRF) [1, 13] is another popular representation for novel view synthesis. It uses MLPs to represent spatially varying color and density fields, and apply ray marching to render. NeRF faces similar limitations as Gaussian Splatting: The illumination, geometry, and material of the objects in the training data are baked into the radiance fields. To separate the illumination and material, many researchers try to incorporate various material and lighting representations into NeRF by conditioning the color field on intermediate quantities such as visibility and lighting [20, 25] with a material/lighting model, leading to more efficient rendering and relighting of objects of complex appearance such as reflective surfaces [18, 22, 32, 34], self-emissive [8] and transparent objects [23]. Surface representations such as neural signed distance function [24, 27] are also introduced to disentangle geometry from appearance, enabling challenging material modeling and high-quality surface reconstruction simultaneously [2, 5, 12, 28]. Using light transport hints generated from signed distance functions, Zeng et al. [31] also demonstrates relighting NeRF with high-frequency shadows and highlights. Other than surface materials, a range of scattering models for volumes are also incorporated to relight translucent objects [30, 36]. Zhang et al. [33] uses the SGGX phase function [7] to achieve a parameterized subsurface scattering appearance. LitNeRF [16] decomposes the lighting into reflectance and intrinsic components and demonstrates physically plausible relighting performance for human faces. Our method shares the same spirit with LitNeRF in terms of the lighting model, but we only use spherical harmonics for representing different components in our model instead of MLP, and we also introduce additional regularization terms to disambiguate multiple solutions in the decomposition.

## 3. Background: 3D Gaussian Splatting

3D Gaussian Splatting (3DGS) [10] represents a scene using numerous volumetric primitives in the form of 3D Gaussian kernels. Each Gaussian kernel is characterized by a set of parameters, including its position $\mu$ (center of the primitive), covariance matrix $\Sigma$ (parameterized by scale and rotation), opacity $o$ at the center, and spherical harmonics coefficients that define the color function $L(\omega_\mathrm{o})$, which depends on the view direction $\omega_\mathrm{o}$.

To render a scene represented by $n$ 3D Gaussian primitives, all primitives are first projected onto the 2D camera plane, resulting in 2D Gaussian kernels with mean $\mu_i'$ and covariance $\Sigma_i'$. The color of a pixel centered at $p$ is then determined by $\alpha$-blending the projected Gaussians from the nearest to the farthest relative to the camera, following the equation:

$$I(\omega_\mathrm{o}) = \sum_{i=1}^{n} T_i \alpha_i L_i(\omega_\mathrm{o}),\ T_i = \prod_{j=1}^{i-1}(1-\alpha_j). \quad (1)$$

Here, $\alpha_i = o_i G_i(p)$ accounts for the falloff induced by the 2D Gaussian, where $G_i(p) = \exp\left(-\frac{1}{2}(p-\mu_i')^t \Sigma_i'^{-1}(p-\mu_i')\right)$. The term $T_i$ denotes the transmittance of from the $i$-th Gaussian primitive

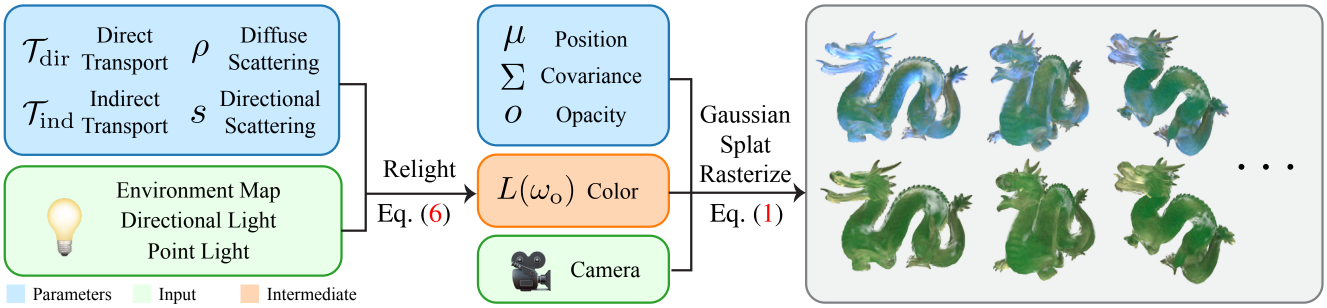


Figure 2. **Pipeline overview**: our method introduces per-Gaussian optimizable lighting parameters: $\mathcal{T}_{\rm dir}$, $\mathcal{T}_{\rm ind}$, $\rho$, and $s$, each represented using spherical harmonics. Given novel lighting conditions, we relight each Gaussian by generating the view-dependent color of each Gaussian $L(\omega_{\rm o})$ represented by spherical harmonics that are compatible with the Gaussian rasterization pipeline, and therefore can render under novel light and view conditions.

to the camera, while the color function $L_i(\omega_{\rm o})$ encodes view-dependent color. This color function is represented by spherical harmonics. However, in Eq. (1), the color function is solely dependent on the view direction, making it only suitable for reconstructing and rendering objects under static illumination.

To achieve the goal of reproducing appearance under dynamic illumination, We extend the color function in Eq. (1) to model lighting-dependent Gaussian primitives. As illustrated in Fig. 2 and detailed in the following sections, the extended color function $L(\omega_{\rm o})$ incorporates additional parameters, including direct and indirect light transport, as well as diffuse and directional scattering components. This allows us to take the input of diverse lighting conditions, including environment maps, directional lights, and point lights, to render Gaussian splats under dynamic illumination.

# 4. Relightable Gaussian Primitives

## 4.1. Intrinsic Light Decomposition

The key to relightable Gaussian primitives is to express the view-dependent color $L(\omega_{\rm o})$ as a function of both view direction $\omega_{\rm o}$ and the lighting conditions, represented by the direction from the light source to the primitive $\omega_{\rm i}$ and the light source's intensity in $\omega_{\rm i}$, denoted by $L_{\rm e}(\omega_{\rm i})$. This approach allows us to update $L(\omega_{\rm o})$ when the lighting changes and to render images under novel lighting conditions.

Inspired by LitNeRF [16], we adopt the intrinsic decomposition model that partitions the out radiance $L(\omega_{\rm o})$ of each Gaussian primitive into the sum of two components, assuming no self-emission: $L(\omega_{\rm o}) = L_{\rm dir}(\omega_{\rm o}) + L_{\rm ind}$, where $L_{\rm dir}$ stands for the direct illumination – the contribution from the light that travels from the emitter to the primitive directly and $L_{\rm ind}$ for the indirect illumination that encompasses the light bounces and scatters in the scene and arrives at the primitive. We follow the common assumption that $L_{\rm ind}$ does not depend on view direction [16].

We model the direct illumination $L_{\rm dir}$ by the ratio of light traveling from the emitter and scattering into view direction, $L_{\rm dir}(\omega_{\rm o}) = \int_{\mathcal{S}^2} \mathcal{T}_{\rm dir}(\omega_{\rm i}) L_{\rm e}(\omega_{\rm i}) f(\omega_{\rm i}, \omega_{\rm o}) \; {\rm d}\omega_{\rm i}$, where $\mathcal{T}_{\rm dir}(\omega_{\rm i}) : \mathcal{S}^2 \rightarrow \mathbb{R}$ is the ratio of the light arriving from the emitter to the primitive in $\omega_{\rm i}$, accounting for attenuation from occlusion or absorption along the light direction; $L_{\rm e}(\omega_{\rm i})$ is the light intensity at the emitter. $f(\omega_{\rm i}, \omega_{\rm o}) : \mathcal{S}^2 \times \mathcal{S}^2 \rightarrow \mathbb{R}^3$ is the scattering function, indicating the amount of scattering when light travels in from $\omega_{\rm i}$ and out into $\omega_{\rm o}$. Deviating from LitNeRF [16], we use 3 channels for $f$ to capture the multichromatic highlight that can be observed in metallic or iridescent reflection, as shown in Fig. 6.

For scattering functions to be physically meaningful, they have to satisfy properties such as being *reciprocal*, $f(\omega_{\rm i}, \omega_{\rm o}) = f(\omega_{\rm o}, \omega_{\rm i}), \;\; \forall \omega_{\rm i}, \omega_{\rm o}$. Another property of scattering functions originates from the conservation of energy—the amount of light leaving a primitive does not exceed the total amount of incoming light, giving $\int_{\mathcal{S}^2} f(\omega_{\rm i}, \omega_{\rm o}) \; {\rm d}\omega_{\rm o} \leq 1, \;\; \forall \omega_{\rm i}$. In BiGS, we incorporate these constraints to reduce the number of degrees of freedom and to achieve a more stable decomposition and optimization, as will be discussed Sec. 5.1.

We further decompose $f$ into two parts: a diffuse component $\rho \in \mathbb{R}^3$, which captures lighting- and view-independent features such as albedo and ambient occlusion, and a directional scattering component $s(\omega_{\rm i}, \omega_{\rm o})$, which models the lighting- and view-dependent effects. With this decomposition, we can write the direct illumination $L_{\rm dir}(\cdot)$ as the following:

$$L_{\rm dir}(\omega_{\rm o}) = \int_{\mathcal{S}^2} \mathcal{T}_{\rm dir}(\omega_{\rm i}) L_{\rm e}(\omega_{\rm i}) (\rho + s(\omega_{\rm i}, \omega_{\rm o})) \; {\rm d}\omega_{\rm i}. \quad (2)$$

As for indirect illumination $L_{\rm ind}$, we model it as a residue term that captures the direct illumination struggle to represent. We express it as

$$L_{\rm ind} = \int_{\mathcal{S}^2} \mathcal{T}_{\rm ind}(\omega_{\rm i}) L_{\rm e}(\omega_{\rm i}) \; {\rm d}\omega_{\rm i}, \quad (3)$$

where $\mathcal{T}_{\text{ind}} : \mathcal{S}^2 \to \mathbb{R}^3$ is the indirect transport operator that models the light travels from the emitter to the primitive after multiple bounces.

Different components in the intrinsic decomposition might lead to multiple solutions that produce similar final rendering, and we discuss how we regularize the components and disambiguate the solutions in Sec. 5.1.

### 4.2. Bidirectional Spherical Harmonics

We show how we use spherical harmonics to represent $\mathcal{T}_{\text{dir}}(\cdot)$ and $\mathcal{T}_{\text{ind}}(\cdot)$, and spherical harmonics extended to inputs of 2 directions, dubbed *bidirectional spherical harmonics* to represent $s(\cdot,\cdot)$. Then we show how to use our representations to compute Gaussian primitives' colors that can be used for rasterization from arbitrary viewpoints.

Spherical harmonics are a special set of orthonormal basis functions over a 3D unit sphere $\mathcal{S}^2$, having a long history of applications in real-time rendering [11, 17]. Using the first $n$ spherical harmonics basis functions $\{y_i(\cdot)\}_{i=1}^n$, a function defined over $\mathcal{S}^2$ can be approximated by the sum of the bases and the $n$ corresponding coefficients $\{c_i\}_{i=1}^n$ Using $\mathcal{T}_{\text{dir}}(\cdot)$ as an example: given a direction $\omega$, it can be expressed as the $\mathcal{T}_{\text{dir}}(\omega) = \sum_{i=1}^n c_i y_i(\omega)$. Likewise, we can represent $\mathcal{T}_{\text{ind}}(\cdot)$ in the same fashion.

For each primitive, we use the first 25 spherical harmonics basis functions to model $\mathcal{T}_{\text{dir}}(\cdot)$ and each channel of $\mathcal{T}_{\text{ind}}(\cdot)$. This amounts to 25 coefficients for $\mathcal{T}_{\text{dir}}(\cdot)$ and 75 for $\mathcal{T}_{\text{ind}}(\cdot)$ (25 per RGB channel). These coefficients are the optimizable parameters in the training pipeline.

Now we extend the representation for functions defined over $\mathcal{S}^2 \times \mathcal{S}^2$, for instance, $s(\cdot,\cdot)$. We can do this by composing two groups of spherical harmonics to form a new basis function for $\mathcal{S}^2 \times \mathcal{S}^2$. Using the first $n$ spherical harmonics basis, $s$ can be written as

$$s(\omega_{\text{i}}, \omega_{\text{o}}) = \sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij} y_i(\omega_{\text{i}}) y_j(\omega_{\text{o}}), \tag{4}$$

and we call $y_i(\omega_{\text{i}})y_j(\omega_{\text{o}})$ *bidirectional spherical harmonics*. This leads to $n^2$ coefficients $\{c_{ij}\}_{i,j=1}^n$. But $n^2$ would have been more than we actually need. Eq. (4) does not enforce the reciprocity of $s$, and is therefore representing a larger function space. We incorporate this important physical property into $s$ by letting $c_{ij} = c_{ji}$ for all $i, j$ pairs. This reduces the space of function that can be represented from $\mathcal{S}^2 \times \mathcal{S}^2$ to those that are reciprocal; see Sec. 8 in supplementary material for a proof. And it also makes our model more compact by reducing the number of parameters from $n^2$ to $n(n+1)/2$. Using 25 bases, $s$ costs $325 \times 3 = 975$ parameters per primitive.

One could evaluate the amount of scattering $s(\omega_{\text{i}}, \omega_{\text{o}})$ given $\omega_{\text{i}}$ and $\omega_{\text{o}}$, and this results in evaluating $s$ every time viewpoint ($\omega_{\text{o}}$) changes. The spherical harmonics representation also allows us to evaluate $s$ over one input variable $\omega_{\text{i}}$ for relighting and later evaluate over $\omega_{\text{o}}$ for novel view synthesis. In other words, when we need to rasterize the primitives from a new view with no lighting change, we save the computation for evaluating over $\omega_{\text{i}}$.

Specifically, given light entering the Gaussian from direction $\omega_{\text{i}}$, we can express the out-scatter color as a function over all $\omega_{\text{o}}$, denoted by $s_{\omega_{\text{i}}}(\omega_{\text{o}}) : \mathcal{S}^2 \to \mathbb{R}^3$. Again, $s_{\omega_{\text{i}}}$ can be represented by spherical harmonics:

$$s_{\omega_{\text{i}}}(\cdot) = \sum_{j=0}^{n} c_j y_j(\cdot), \tag{5}$$

where coefficients $c_j = \sum_{i=1}^n c_{ij} y_i(\omega_{\text{i}})$ come from summing up bases for $\omega_{\text{i}}$.

### 4.3. Rendering Under Novel Lighting Conditions

Given $t$ directional lights, each characterized by their direction $\omega_{\text{i}0}, \omega_{\text{i}1}, \cdots, \omega_{\text{i}t}$, and the corresponding intensity $L_e(\omega_{\text{i}t})$, our goal is to compute $L(\omega_{\text{o}})$ for each primitive. To do this, we evaluate Eq. (2) and Eq. (3) for direct and indirect illumination respectively, in which the integral over $\mathcal{S}^2$ becomes the summation of contribution from each light. The final color reads:

$$L(\omega_{\text{o}}) = \sum_{\omega_{\text{i}t}} \mathcal{T}_{\text{dir}}(\omega_{\text{i}t})(\rho + s_{\omega_{\text{i}t}}(\omega_{\text{o}})) + \mathcal{T}_{\text{ind}}(\omega_{\text{i}t}))L_e(\omega_{\text{i}t}). \tag{6}$$

Then, we can rasterize the primitives from arbitrary view directions to obtain relighted renderings.

We can also extend it to point lights and environment map light sources. To relight with a point light source centered at $\mathbf{x}_t$ and center light intensity $\hat{L}_t$, substitute into Eq. (6) $\omega_{\text{i}t} = (\mathbf{x}_t - \mu)/||\mathbf{x}_t - \mu||$ and, $L_e(\omega_{\text{i}t}) = \hat{L}_t/||\mathbf{x}_t - \mu||^2$ where $\mu$ is the Gaussian center position and $||\cdot||$ denotes Euclidean norm.

To relight with an environment map, we employ a low-frequency approximation by sampling the environment map at a set of predefined lighting positions $\mathbf{x}_t$. The corresponding light direction is given by $\omega_{\text{i}t} = (\mathbf{x}_t - \mu)/||\mathbf{x}_t - \mu||$, where the light intensity $L_{\text{e}}(\omega_{\text{i}t})$ is determined by the pixel value of the environment map, weighted by the solid angle associated with $\omega_{\text{i}t}$.

## 5. Training BiGS on OLAT data

To construct relightable Gaussian primitives, we employ an inverse optimization method. Given a set of images in the format of One-Light-At-a-Time (OLAT) data as shown in Fig. 1 (Input OLAT Data), we search for the optimal configuration of each Gaussian primitive to best match the provided input images.

### 5.1. Model Supervision and Disambiguation

Our method introduces the following optimizable parameters for each Gaussian primitive: $\mathcal{T}_{\text{dir}}$, $\rho$, $s$, and $\mathcal{T}_{\text{ind}}$. These parameters are optimized alongside the original Gaussian Splatting parameters, including position $\mu$, covariance $\Sigma$, and opacity $o$ for each primitive.

We supervise our model with two goals in mind: realistic image synthesis under novel lighting conditions, and physically plausible intrinsic light decomposition. These two goals lead to the following loss function terms: image reconstruction loss $\mathcal{L}_{\text{rec}}$ and two regularization terms $\mathcal{L}_s, \mathcal{L}_+$ to promote plausible intrinsic light decomposition.

For the image reconstruction loss, we use the same term as in 3DGS [10], that include the mean absolute error and SSIM loss, evaluated on the rendered $\Gamma(I)$ and the reference image $\Gamma(I')$ after clamping the pixel values to $[0, 1]$ and applying the gamma correction function $\Gamma$ (gamma being 2.2), as we use HDR images for training our model. Section 5.2 provides more details of our HDR OLAT datasets.

Optimizing Eq. (2) without regularization might lead to unphotorealistic ambiguity in $\mathcal{T}_{\text{dir}}$ and $s$ and unstable training, as can be seen in Fig. 5. Therefore, we introduce $\mathcal{L}_{\text{s}}$ and $\mathcal{L}_+$ as regularizers. $\mathcal{L}_{\text{s}}$ is for alleviating the ambiguity issue by using the energy conservation constraint of the phase function as a penalty term,

$$\mathcal{L}_{\text{s}} = \frac{1}{N} \left( \int_{\mathcal{S}^2} s_{\omega_{\text{i}}}(\omega_{\text{o}})\, \mathrm{d}\omega_{\text{o}} - 1 \right)_+^2, \tag{7}$$

where $(\cdot)_+ = \max\{\cdot, 0\}$ refers to the ReLU function, and $N$ is the number of Gaussians.

During training, $\mathcal{L}_{\text{s}}$ is evaluated twice with different $\omega_{\text{i}}, \omega_{\text{o}}$: the first time by using the training OLAT light direction $\omega_{\text{i}}$ and camera direction $\omega_{\text{o}}$ associated with the training image; the second time by using randomly sampled $\omega_{\text{i}}, \omega_{\text{o}}$ to evaluate the loss with, then add up with the first evaluation. The random sampling evaluation helps generalize the constraint to novel lights and views unseen during training.

The second regularization is the non-negativity of light intensity. We explicitly constrain $\rho$ between 0 and 1 by using a Sigmoid function. For other lighting components, namely $s, \mathcal{T}_{\text{dir}}$ and $\mathcal{T}_{\text{ind}}$, we clamp the negative part after computing their values by $\max\{\cdot, 0\}$ before rendering the image. However, the clamping operation causes the gradients of the negative values to be zero, and therefore they do not get updated during the training loop. We use another loss term $\mathcal{L}_+$ to encourage the values to be non-negative:

$$\mathcal{L}_+ = \frac{1}{N} \left[ \mathcal{T}_{\text{dir}}(\omega_{\text{i}})_-^2 + \mathcal{T}_{\text{ind}}(\omega_{\text{i}})_-^2 + s(\omega_{\text{i}}, \omega_{\text{o}})_-^2 \right], \tag{8}$$

where $(\cdot)_- = \min\{\cdot, 0\}$. This term is evaluated in the same manner as $\mathcal{L}_s$: on both training $\omega_{\text{i}}, \omega_{\text{o}}$ and a pair of randomly sampled $\omega_{\text{i}}, \omega_{\text{o}}$ per iteration.

Another strategy during optimization is the late activation of $\mathcal{T}_{\text{ind}}$. $\mathcal{T}_{\text{ind}}$ in Eq. (6) is defined as a residual term that only represents the effects unable to be captured by other components, but the optimization could cause $\mathcal{T}_{\text{ind}}$ to dominate, representing most of the light transport effect. We mitigate this problem by only adding $\mathcal{T}_{\text{ind}}$ into optimization at the late stage of the optimization (final 30k out of 100k iterations), during which the values of other components have already stabilized.

## 5.2. Data and Implementation Details

***Synthetic data***. Our synthetic dataset is generated using 40 light sources and 48 cameras, amounting to 1920 images. These lights and cameras are evenly placed on a hemisphere with the subject at the center. The images in our dataset are comprised of three parts: 1) 40 different OLAT lighting conditions: only one of the light sources is turned on, and each camera generates an image under this OLAT condition, serving as the training set; 2) all light turned on: all 40 lights are turned on simultaneously. This partition provides a neutral lighting condition that is used to train the original Gaussian Splatting model to obtain a set of Gaussian primitives that serves as the initial values of our optimization loop; 3) 58 novel OLAT lighting conditions with one camera per OLAT on which we evaluate our algorithm.

***Capture data***. We also evaluate our method using OLAT data that we capture with a light stage of 216 global-shutter industrial cameras and 145 LED lights. 88 front-viewing LEDs are used to capture 88 OLAT conditions, resulting in a total of 19,008 images and 216 additional images with all lights on to extract the foreground mask and obtain the initialization of the training process. Example images can be found in Fig. 10 in the supplementary material. The positions of the cameras and lights are calibrated such that they are in the same coordinate system.

***Training.*** For simplicity of the training process, we use the all-light-turned-on part of our data to train a Gaussian Splatting model for initializing our training pipeline. We take the Gaussian Splat's opacity, covariance, and positions as the initial values of the same quantities in our model, and the colors as the initial values of $\rho$. During this training phase, the number of Gaussians stays constant, no culling, merging, or splitting. This implies the training quality of our relightable pipeline is limited by the original Gaussian Splatting we start the training with. We will leave the exploration of a more systematic training scheme as future work. For training the Gaussian Splat model, we use SPLATFACTO provided in nerfstudio [19].

# 6. Results

We present experiments on relighting and novel view synthesis on synthetic and capture data. All experiments are conducted on one NVIDIA A100 GPU. All models

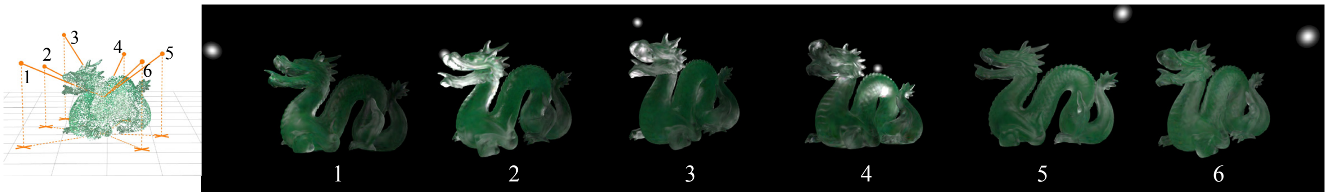


Figure 3. **Point light relighting**: The leftmost shows the six positions of the point light that illuminates the translucent DRAGON. Renderings for each light position are shown. DRAGON becomes brighter as the light gets nearer; please see the supplementary video.

are trained with 100k iterations on the OLAT data, taking 1.5 hours. We implement a CUDA kernel for evaluating the spherical harmonics, and the relighting computation, i.e., Eq. (6) is implementing in PyTorch. Rasterization is performed using the library GSPLAT [29].

***Model size and runtime.*** Each Gaussian primitive has 1,089 optimizable parameters, amounting to $4.254$ KB memory consumption using 32-bit floating point numbers. This usually leads to a nearly 200 MB memory cost for a model of around 40,000 primitives. The runtime of our pipeline scales linearly with the number of Gaussian primitives. With our hardware, computing the colors of each primitive per Eq. (6) takes on average 5–6ms for a model of around 40,000 primitives, dwarfed by the rasterization step which takes around 19.5ms. Please refer to Tab. 1 in the supplementary material for detailed statistics.

## 6.1. Relighting

Our method applies to different surface and volumetric materials. As shown in Fig. 8, our method gives plausible relighting and intrinsic lighting decomposition across various types of materials on different objects using a point light. The objects we tested include glossy surface-based appearance such as METALBUNNY and IRIDESCENCEBALL; fuzzy material such as FURBALL and HAIRBALL; translucent volume such as DRAGON. The method performs especially well on modeling volumetric and fuzzy appearance. Besides synthetic data, our model also performs well on our capture OLAT dataset as in Fig. 9. We refer the readers to the video in the supplementary material for the results of intrinsic decomposition lit with different point light sources.

Our method supports relighting an object using point light, directional light, and environment maps. In Fig. 3, the translucent DRAGON model is lit by a point light, rendered from varying viewpoints. The point light is rotating around with varying distances fromom the object. The brightness of the model increases as the point light approaches it. In Fig. 4, the KNOB with both reflective surface and jade-like material is re-lit with a directional light shone vertically from the top, the reflection of the surface can be observed from the rendering. The FURBALL is lit by the environment map from two angles and produces consistent novel view synthesis results. Figure 9 shows environment map relighting results on the capture dataset. Please see the video in the supplementary material for more results.

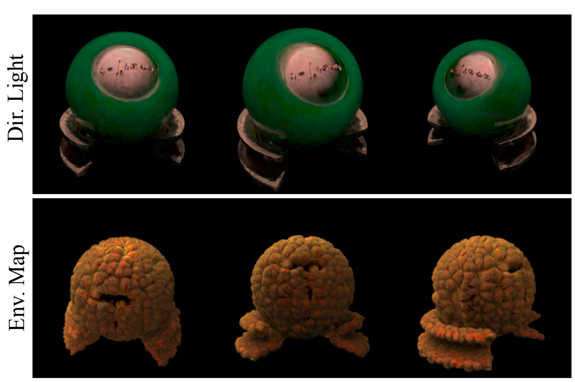


Figure 4. **Directional light and environment map relighting**. The KNOB model is lit with a vertical directional light. The FURBALL model is lit using an environment map. Both are viewed from three different angles.

## 6.2. Ablation Study

**The $\mathcal{L}_s$ loss term** is introduced to constrain the directional scattering $s$, and its effect can be seen in Fig. 5: without $\mathcal{L}_s$, the value of $s$ is unchecked and could be very large. This means the amount of directionally scattered light can be many times that of the incoming light, leading to a nonphysical decomposition, and also noise in the directional scattering component and the final rendering.

**The number of channels of** $s$ affects the specular highlight that our model is able to learn. Using a single channel only generates specular highlights of neutral color, whereas multiple-channel $s$ supports highlights of different tones, such as the blue tint in IRIDESCENCEBALL in Fig. 6.

## 6.3. Comparison

We qualitatively compare our method against R3G [4], and GaussianShader [9]. It is worth noting that both have different training setups from ours; we leverage OLAT datasets and they require only one environment map lighting during training. We train both of their methods using a neutral environment map, and relight with another environment map GEARSHOP. Our model restores more volumetric light transport effects, such as the subsurface scattering in DRAGON and preserves the details of the fur in FURBALL that can be seen in Fig. 7.

## 7. Conclusion

We present a method that enables image synthesis under novel view and lighting conditions of objects represented by Gaussian Splats. Our method decomposes the appearances of the Gaussian primitives into intrinsic components represented by spherical harmonics. Our model is able to generate spherical harmonics coefficients for each Gaussian primitive that are compatible with the Gaussian rasterization pipeline, enabling real-time relighting and novel view synthesis performance. We test our method over OLAT datasets of a variety of materials, showcasing the versatile modeling capability for various appearances.

***Limitations and future work.*** Our method has some limitations that can be improved in the future. Even though we propose a few strategies to reduce the ambiguity of different lighting components and increase the stability of the training, there are still cases where the intrinsic decomposition does not yield satisfactory results, such as the directional scattering component of SPOT in Fig. 8. To represent different lighting components in our method, we use spherical harmonics, which are an inherently low-frequency representation. This makes our method struggle to accurately capture some high-frequency light transport effects, such as hard shadow. We believe there are multiple promising directions to improve: inspiration can be taken from prior works from the real-time rendering community, for example, choices of basis functions for representing all-frequency light transport effects [14, 21, 26]; alternatively, one might condition lighting components on the positions and the opacities of other Gaussian primitives in the scene. There are also many possibilities to parameterize the scattering functions or incorporate material models to enable more robust modeling of certain light transport effects. Another limitation of our method is that it requires as input dense OLAT datasets, which are usually obtained by dedicated capture devices such as light stages that are rather expensive to set up.

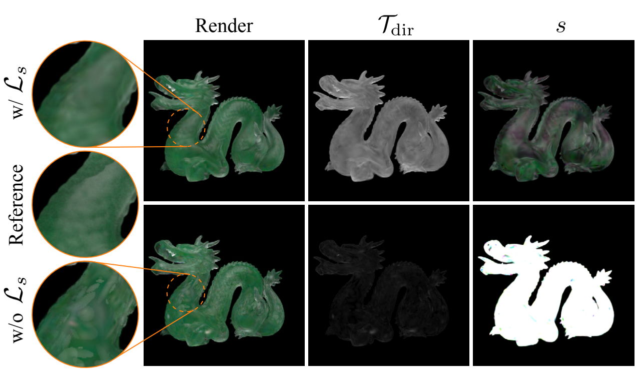


Figure 5. $\mathcal{L}_s$ **preventing excessive value of** $s$: without constraints, the optimization loop might make $s$ nonphysically large and cause noisy blobs in the rendering. Adding $\mathcal{L}_s$ alleviates the problem by penalizing overly large $s$.

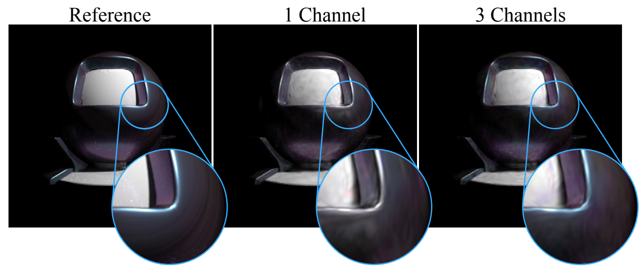


Figure 6. **Number of channels in directional scattering**: using only 1 channel for directional scattering is not sufficient for the blue-tinted reflection from the iridescent material.

## Acknowledgments

We thank the reviewers for their suggestions to improve the manuscript; Keenan Crane for the model SPOT; Benedikt Bitterli for the model FURBALL; Stanford Computer Graphics Laboratory for the models DRAGON and BUNNY; Wenzel Jakob for the model used in IRIDESCENCE BALL; Samuli Laine and Tero Karras for the model HAIRBALL; Jeremy Chew and Rafael Bischof for providing feedback on the manuscript. Special thanks to Changxi Zheng for supporting the internship program at Tencent Pixel Lab.

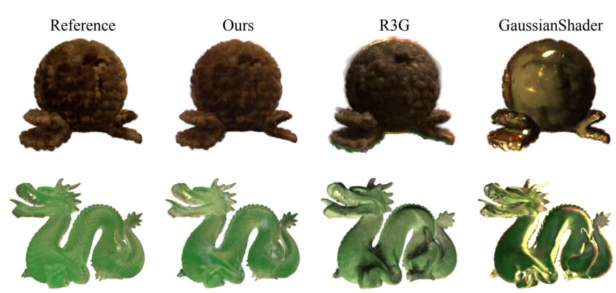


Figure 7. **Comparison with R3G [4] and GaussianShader [9]**: our model preserves the details of the fur in FURBALL and captures volumetric subsurface scattering effect in DRAGON.

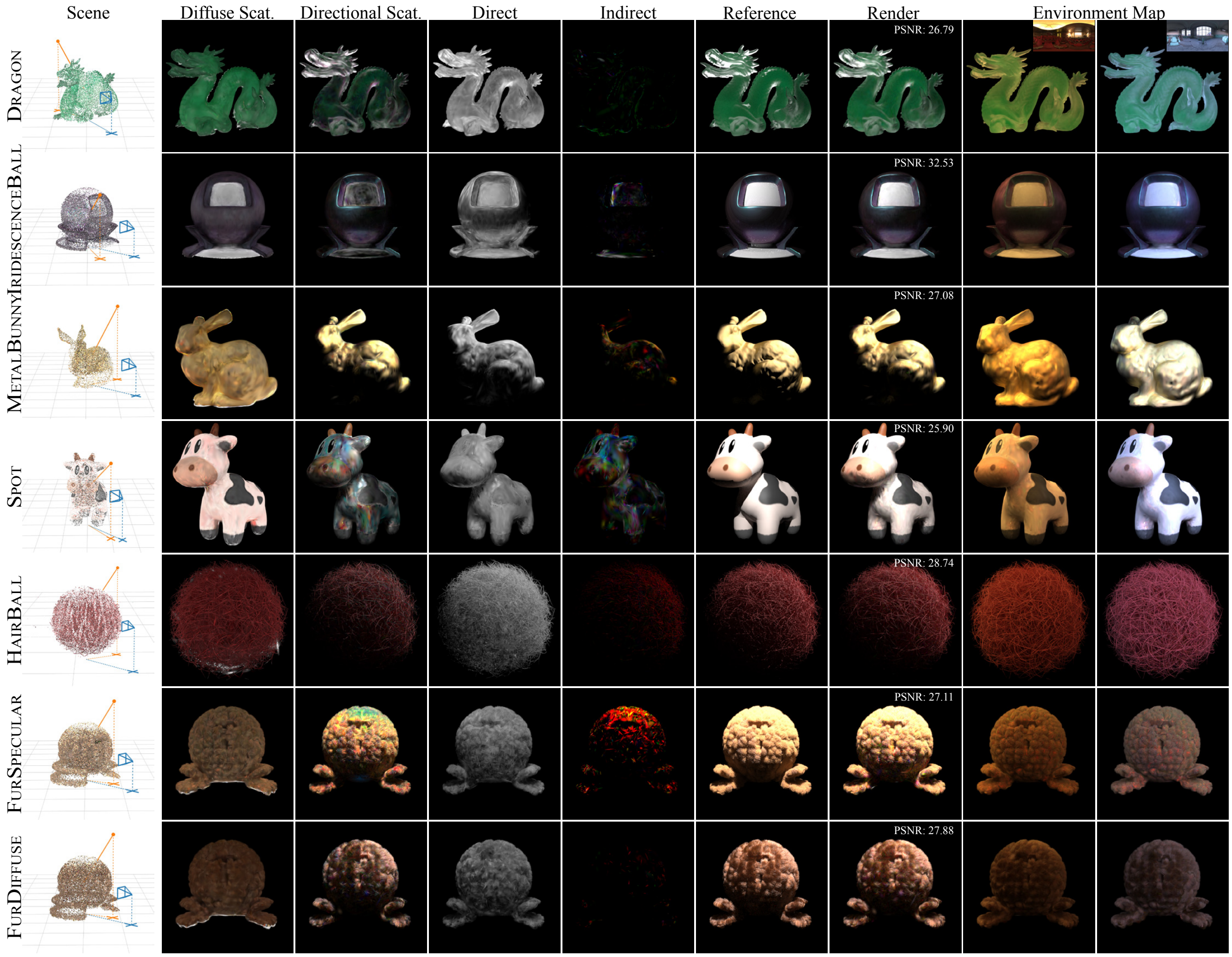


Figure 8. **Intrinsic decomposition and relighting.** We visualize the intrinsic decomposition components given a novel point light, and relighting under environment maps. From left to right: the scene setups including novel point light positions and camera poses (unseen during training); diffuse scattering; directional scattering; direct transport; indirect transport; the reference images; our renders with PSNR between the references and renders; two renderings under two distinct environment maps.

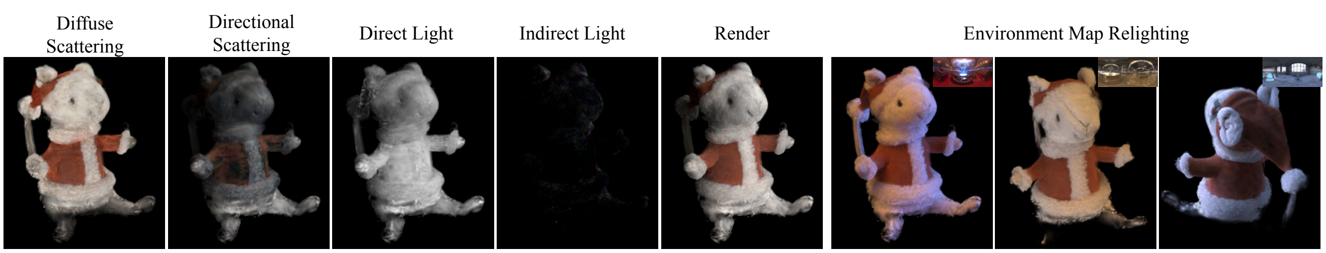


Figure 9. **Capture data result**: We test our method on the PLUSHY OLAT data, and perform intrinsic decomposition under a novel point light source, and relight the PLUSHY with three environment maps, rendering from three distinct viewpoints.

## References


[1] Jonathan T. Barron, Ben Mildenhall, Dor Verbin, Pratul P. Srinivasan, and Peter Hedman. Mip-NeRF 360: Unbounded Anti-Aliased Neural Radiance Fields. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5460–5469, New Orleans, LA, USA, 2022. IEEE. 2

[2] Yuxiang Cai, Jiaxiong Qiu, Zhong Li, and Bo Ren. NeuralTO: Neural Reconstruction and View Synthesis of Translucent Objects. *ACM Trans. Graph.*, 43(4):50:1–50:14, 2024. 2

[3] Jorge Condor, Sebastien Speierer, Lukas Bode, Aljaz Bozic, Simon Green, Piotr Didyk, and Adrian Jarabo. Volumetric Primitives for Modeling and Rendering Scattering and Emissive Media, 2024. arXiv:2405.15425 [cs]. 2

[4] Jian Gao, Chun Gu, Youtian Lin, Hao Zhu, Xun Cao, Li Zhang, and Yao Yao. Relightable 3D Gaussian: Real-time Point Cloud Relighting with BRDF Decomposition and Ray Tracing, 2023. arXiv:2311.16043 [cs]. 1, 2, 6, 7

[5] 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. pages 4251–4260, 2023. 2

[6] Yumeng He, Yunbo Wang, and Xiaokang Yang. GS-Phong: Meta-Learned 3D Gaussians for Relightable Novel View Synthesis, 2024. arXiv:2405.20791 [cs]. 2

[7] Eric Heitz, Jonathan Dupuy, Cyril Crassin, and Carsten Dachsbacher. The SGGX microflake distribution. *ACM Transactions on Graphics*, 34(4):48:1–48:11, 2015. 2

[8] Jinseo Jeong, Junseo Koo, Qimeng Zhang, and Gunhee Kim. ESR-NeRF: Emissive Source Reconstruction Using LDR Multi-view Images, 2024. arXiv:2404.15707 [cs]. 2

[9] 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, 2023. arXiv:2311.17977 [cs]. 1, 2, 6, 7

[10] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkuehler, and George Drettakis. 3D Gaussian Splatting for Real-Time Radiance Field Rendering. *ACM Transactions on Graphics*, 42(4):139:1–139:14, 2023. 1, 2, 5

[11] Jaakko Lehtinen. A framework for precomputed and captured light transport. *ACM Trans. Graph.*, 26(4):13–es, 2007. 4

[12] Yuan Liu, Peng Wang, Cheng Lin, Xiaoxiao Long, Jiepeng Wang, Lingjie Liu, Taku Komura, and Wenping Wang. NeRO: Neural Geometry and BRDF Reconstruction of Reflective Objects from Multiview Images. *ACM Transactions on Graphics*, 42(4):1–22, 2023. 2

[13] Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis. In *ECCV*, 2020. 2

[14] Ren Ng, Ravi Ramamoorthi, and Pat Hanrahan. Triple product wavelet integrals for all-frequency relighting. *ACM Transactions on Graphics*, 23(3):477–487, 2004. 7

[15] Shunsuke Saito, Gabriel Schwartz, Tomas Simon, Junxuan Li, and Giljoo Nam. Relightable Gaussian Codec Avatars, 2023. arXiv:2312.03704 [cs]. 2

[16] Kripasindhu Sarkar, Marcel C. Bühler, Gengyan Li, Daoye Wang, Delio Vicini, Jérémy Riviere, Yinda Zhang, Sergio Orts-Escolano, Paulo Gotardo, Thabo Beeler, and Abhimitra Meka. LitNeRF: Intrinsic Radiance Decomposition for High-Quality View Synthesis and Relighting of Faces. In *SIGGRAPH Asia 2023 Conference Papers*, pages 1–11, Sydney NSW Australia, 2023. ACM. 2, 3

[17] Peter-Pike Sloan, Jan Kautz, and John Snyder. Precomputed radiance transfer for real-time rendering in dynamic, low-frequency lighting environments. *ACM Transactions on Graphics*, 21(3):527–536, 2002. 4

[18] 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. arXiv:2012.03927 [cs]. 2

[19] Matthew Tancik, Ethan Weber, Evonne Ng, Ruilong Li, Brent Yi, Terrance Wang, Alexander Kristoffersen, Jake Austin, Kamyar Salahi, Abhik Ahuja, David Mcallister, Justin Kerr, and Angjoo Kanazawa. Nerfstudio: A Modular Framework for Neural Radiance Field Development. In *ACM SIGGRAPH 2023 Conference Proceedings*, pages 1–12, New York, NY, USA, 2023. Association for Computing Machinery. 5

[20] Marco Toschi, Riccardo De Matteo, Riccardo Spezialetti, Daniele De Gregorio, Luigi Di Stefano, and Samuele Salti. ReLight My NeRF: A Dataset for Novel View Synthesis and Relighting of Real World Objects. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 20762–20772, Vancouver, BC, Canada, 2023. IEEE. 2

[21] Yu-Ting Tsai and Zen-Chung Shih. All-frequency precomputed radiance transfer using spherical radial basis functions and clustered tensor approximation. *ACM Transactions on Graphics*, 25(3):967–976, 2006. 7

[22] 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, 2021. arXiv:2112.03907 [cs]. 2

[23] Dongqing Wang, Tong Zhang, and Sabine Süsstrunk. NEMTO: Neural Environment Matting for Novel View and Relighting Synthesis of Transparent Objects. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 317–327, Paris, France, 2023. IEEE. 2

[24] Peng Wang, Lingjie Liu, Yuan Liu, Christian Theobalt, Taku Komura, and Wenping Wang. NeuS: Learning Neural Implicit Surfaces by Volume Rendering for Multi-view Reconstruction. In *Advances in Neural Information Processing Systems*, pages 27171–27183. Curran Associates, Inc., 2021. 2

[25] Yingyan Xu, Gaspard Zoss, Prashanth Chandran, Markus Gross, Derek Bradley, and Paulo Gotardo. ReNeRF: Relightable Neural Radiance Fields with Nearfield Lighting. 2023. 2

[26] Zilin Xu, Zheng Zeng, Lifan Wu, Lu Wang, and Ling-Qi Yan. Lightweight Neural Basis Functions for All-Frequency Shading. In *SIGGRAPH Asia 2022 Conference Papers*, pages 1–9, Daegu Republic of Korea, 2022. ACM. 7

[27] Lior Yariv, Jiatao Gu, Yoni Kasten, and Yaron Lipman. Volume Rendering of Neural Implicit Surfaces, 2021. arXiv:2106.12052 [cs]. 2

[28] Lior Yariv, Peter Hedman, Christian Reiser, Dor Verbin, Pratul P. Srinivasan, Richard Szeliski, Jonathan T. Barron, and Ben Mildenhall. BakedSDF: Meshing Neural SDFs for Real-Time View Synthesis. In *ACM SIGGRAPH 2023 Con-*

*ference Proceedings*, pages 1–9, New York, NY, USA, 2023. Association for Computing Machinery. 2

[29] Vickie Ye, Ruilong Li, Justin Kerr, Matias Turkulainen, Brent Yi, Zhuoyang Pan, Otto Seiskari, Jianbo Ye, Jeffrey Hu, Matthew Tancik, and Angjoo Kanazawa. gsplat: An open-source library for Gaussian splatting. *arXiv preprint arXiv:2409.06765*, 2024. 6

[30] Hong-Xing Yu, Michelle Guo, Alireza Fathi, Yen-Yu Chang, Eric Ryan Chan, Ruohan Gao, Thomas Funkhouser, and Jiajun Wu. Learning Object-Centric Neural Scattering Functions for Free-Viewpoint Relighting and Scene Composition, 2023. arXiv:2303.06138 [cs]. 2

[31] Chong Zeng, Guojun Chen, Yue Dong, Pieter Peers, Hongzhi Wu, and Xin Tong. Relighting Neural Radiance Fields with Shadow and Highlight Hints. In *ACM SIGGRAPH 2023 Conference Proceedings*, pages 1–11, New York, NY, USA, 2023. Association for Computing Machinery. 2

[32] 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. arXiv:2106.01970 [cs]. 2

[33] Youjia Zhang, Teng Xu, Junqing Yu, Yuteng Ye, Yanqing Jing, Junle Wang, Jingyi Yu, and Wei Yang. NeMF: Inverse Volume Rendering with Neural Microflake Field. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 22862–22872, Paris, France, 2023. IEEE. 2

[34] Quan Zheng, Gurprit Singh, and Hans-peter Seidel. Neural Relightable Participating Media Rendering. In *Advances in Neural Information Processing Systems*, pages 15203–15215. Curran Associates, Inc., 2021. 2

[35] Yang Zhou, Songyin Wu, and Ling-Qi Yan. Unified Gaussian Primitives for Scene Representation and Rendering, 2024. arXiv:2406.09733 [cs]. 2

[36] Shizhan Zhu, Shunsuke Saito, Aljaz Bozic, Carlos Aliaga, Trevor Darrell, and Christoph Lassner. Neural Relighting with Subsurface Scattering by Learning the Radiance Transfer Gradient, 2023. arXiv:2306.09322 [cs]. 2