PhySG: Inverse Rendering with Spherical Gaussians for Physics-based Material Editing and Relighting

Kai Zhang* Fujun Luan* Qianqian Wang Kavita Bala Noah Snavely

Cornell University

Abstract

We present PhySG, an end-to-end inverse rendering pipeline that includes a fully differentiable renderer, and can reconstruct geometry, materials, and illumination from scratch from a set of images. Our framework represents specular BRDFs and environmental illumination using mixtures of spherical Gaussians, and represents geometry as a signed distance function parameterized as a Multi-Layer Perceptron. The use of spherical Gaussians allows us to efficiently solve for approximate light transport, and our method works on scenes with challenging non-Lambertian reflectance captured under natural, static illumination. We demonstrate, with both synthetic and real data, that our reconstructions not only enable rendering of novel viewpoints, but also physics-based appearance editing of materials and illumination.

1. Introduction

Vision as inverse graphics has long been an intriguing concept. Solving inverse rendering problems, i.e., recovering shape, material and lighting from images, has thus been a long-standing goal. Recently, neural rendering methods [46, 54, 29, 31, 25, 55, 21, 32, 28, 44, 43, 35, 4, 47], have drawn significant attention due to their remarkable success in a range of problems, including shape reconstruction, novel view synthesis, non-physically-based relighting, and surface reflectance map estimation. These neural rendering methods adopt scene representations that are either physical, neural, or a mixture of both, along with a neural-network-based renderer. Methods that reconstruct textures or radiance fields [25, 54, 31] work well for the task of interpolating novel views, but do not factorize appearance into lighting and materials, precluding physically-based appearance manipulation like material editing or relighting.

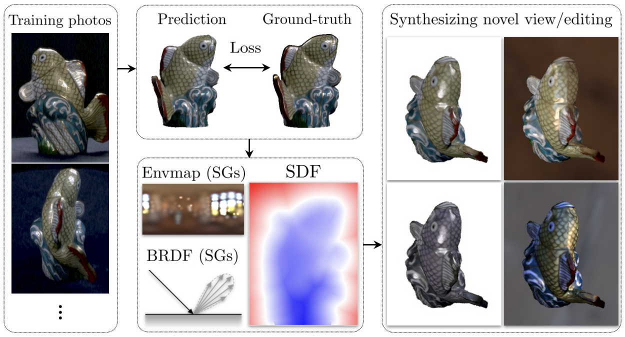

Figure 1: PhySG performs physics-based inverse rendering by taking as input multi-view images of a static glossy object under static natural illumination and jointly optimizes for geometry (represented by an SDF), material BRDF and environment maps (both represented by a mixture of spherical Gaussians), which can then be used for novel view synthesis, relighting and material editing.

Prior multi-view inverse rendering methods assume RGBD input [35, 4] or varying illumination across input images achieved either by co-locating an active flashlight with moving cameras [7, 8, 40, 30] or capturing objects on a turntable with a fixed camera [51, 10]. Learning-based single-view methods that recover shape, illumination, and material properties have also been proposed [20, 5].

In this work, we tackle the multi-view inverse rendering problem under the challenging setting of normal RGB input images sharing the same static illumination, without assuming scanned geometry. To this end, we propose **PhySG**, an end-to-end physically-based differentiable rendering pipeline to jointly estimate lighting, material, geometry and surface normals from posed multi-view images of specular objects. In our pipeline, we represent shape using signed distance functions (SDFs), building on their successful use in recent work [54, 15, 34, 24, 58]. Additionally, a key component of our framework is our use of spherical Gaussians to approximate lighting and specular BRDFs allowing for efficient approximate evaluation of light transport [53]. From 2D images alone, our method jointly reconstructs shape, illumination, and materials and allows for

*Authors contributed equally to this work.
†Project page: https://kai-46.github.io/PhySG-website/.

subsequent physics-based appearance manipulations such as material editing and relighting.

In summary, our contributions are as follows:

- PhySG, an end-to-end inverse rendering approach to this problem of jointly estimating lighting, material properties, and geometry from multi-view images of glossy objects under static illumination. Our pipeline utilizes spherical Gaussians to approximately and efficiently evaluate the rendering equation in closed form.
- Compared to prior neural rendering approaches, we show that PhySG not only generalizes to novel viewpoints, but also enables physically-intuitive material editing and relighting.

2. Background

Our approach lies at the intersection of multiple fields. We briefly review the related prior works below.

Neural rendering. The success of neural rendering [46, 29, 54, 31, 44, 43, 47] has generated significant excitement. In particular, NeRF [29] enables photo-realistic novel view synthesis by representing scenes as radiance fields via multi-layer-perceptrons (MLPs) and fitting these to a collection of input views. While NeRF represents scenes as volumetric opacity fields, other recent methods like DVR [31] and IDR [54] are surface-based. In these three works, appearance is represented by a single MLP that takes a 3D point (and a view direction), and outputs a color. Hence, their appearance model is essentially a surface light field [50] that treats objects as light sources. Such an approach works well for novel view synthesis, but does not disentangle material and lighting, and hence is not suitable for physics-based relighting and material editing. Other approaches learn an appearance space [28, 21, 25] from Internet photos of landmarks captured under diverse lighting, but are not physics-based and cannot generalize to arbitrary new lighting.

In contrast to such prior work that represents appearance as a single neural network, we model appearance via the physical rendering equation. Our approach can solve challenging inverse rendering problems involving specular or glossy objects under static lighting, and enable physically meaningful editing of lighting and materials.

Material and environment estimation. To estimate material properties, most prior works require scenes to be captured under varying illumination [6, 7, 8, 40, 30, 51, 10, 13]. They either place the object of interest on a mechanical turntable and capture it with a fixed camera [51, 10], or move a camera with co-located flashlight to capture a static object from multiple viewpoints [6, 7, 8, 40, 30]. The varying illumination yields rich cues for inferring material properties and geometry [3]. For environment estimation from multi-view images, prior works [35, 23] factorize scene appearance into diffuse image and surface reflectance map given high-quality geometry from RGBD sensors. The surface reflectance map entangles the material and lighting, because it represents the distant environmental illumination convolved with an object's specular BRDF, hence preventing relighting. In contrast to a global environment map, Azinovic *et al.* [4] model lighting as surface emissions, and use a Monte Carlo differentiable renderer to jointly estimate material properties and surface emissions from multi-view images conditioned on scanned geometry and object segmentation masks. Other work seeks to predict illumination, materials and shape from a single image via learning-based priors [5, 20, 39]. Ramamoorthi and Hanrahan [38] estimates BRDF and lighting via deconvolution given known geometry. In our work, we aim to jointly estimate the material and environment, together with geometry and surface normals, solely from multi-view 2D images under the challenging setting of unknown static natural illumination.

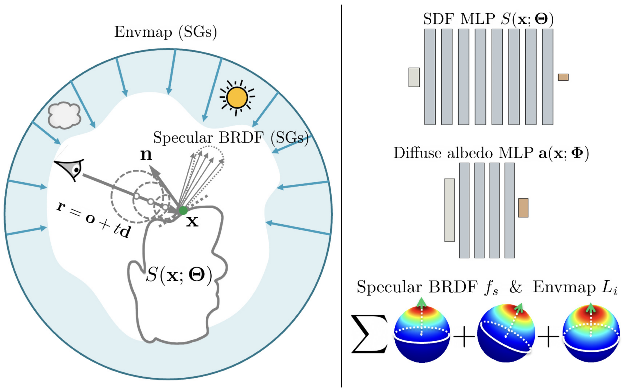
Envmap (SGs)
Specular BRDF (SGs)
$\mathbf{n}$
$\mathbf{r} = \mathbf{o} + t\mathbf{d}$
$\mathbf{x}$
$S(\mathbf{x};\mathbf{\Theta})$
SDF MLP $S(\mathbf{x};\mathbf{\Theta})$
Diffuse albedo MLP $\mathbf{a}(\mathbf{x};\mathbf{\Phi})$
Specular BRDF f_s & Envmap L_i
$\sum$ + +

Figure 2: Overview of our PhySG inverse rendering pipeline. To render the color for a camera ray $\mathbf{r} = \mathbf{o} + t\mathbf{d}$, we first use sphere tracing to find the ray's intersection $\mathbf{x}$ with the geometry in the form of a signed distance function (SDF) represented as an MLP $S(\mathbf{x};\mathbf{\Theta})$. The surface normal $\mathbf{n} = \nabla_{\mathbf{x}} S$ at location $\mathbf{x}$ is then computed as the SDF gradient. We also represent the spatially-varying diffuse albedo $\mathbf{a}(\mathbf{x};\mathbf{\Phi})$ with an MLP. Given the surface normal, albedo, and viewing direction at $\mathbf{x}$, we render a color using the spherical Gaussian (SG) renderer, where we represent both the environment map and a specular BRDF using SGs. The rendered image can then be compared to the ground-truth via image reconstruction loss to jointly optimize the unknowns: geometry and surface normal, spatially-varying diffuse albedo, specular BRDF and environment map.

Joint shape and appearance refinement. Given the initial geometry and appearance from RGBD sensors, Maier et al. [24] and Zollhofer et al. [58] jointly refine geometry and appearance by assuming Lambertian BRDF and incorporating shading cues. They pre-compute a lighting model based on spherical harmonics [37], then fix it while optimizing the shape and diffuse albedo. They adopt voxelized SDFs as their geometric representation. Assuming known illumination, Oxholm and Nishino [33] also exploit reflectance cues to refine geometry computed via visual hulls. In contrast to these prior works, our method does not require scanned geometry or a known environment map. Instead, we estimate material and lighting parameters, as well as geometry and surface normals in an end-to-end fashion.

The rendering equation. Kajiya et al. [16] proposed the rendering equation based on the physical law of energy conservation. For a surface point $\mathbf{x}$ with surface normal $\mathbf{n}$, suppose $L_i(\boldsymbol{\omega}_i;\mathbf{x})$ is the incident light intensity at location $\mathbf{x}$ along the direction $\boldsymbol{\omega}_i$, and BRDF $f_r(\boldsymbol{\omega}_o,\boldsymbol{\omega}_i;\mathbf{x})$ is the reflectance coefficient of the material at location $\mathbf{x}$ for incident light direction $\boldsymbol{\omega}_i$ and viewing direction $\boldsymbol{\omega}_o$, then the observed light intensity $L_o(\boldsymbol{\omega}_o;\mathbf{x})$ is an integral over the hemisphere $\Omega = \{\boldsymbol{\omega}_i : \boldsymbol{\omega}_i \cdot \mathbf{n} > 0\}$ [*]:

$$L_o(\boldsymbol{\omega}_o;\mathbf{x}) = \int_{\Omega} L_i(\boldsymbol{\omega}_i)\, f_r(\boldsymbol{\omega}_o,\boldsymbol{\omega}_i;\mathbf{x})\, (\boldsymbol{\omega}_i \cdot \mathbf{n}) \mathrm{d}\boldsymbol{\omega}_i. \quad (1)$$

The BRDF $f_r(\boldsymbol{\omega}_o,\boldsymbol{\omega}_i;\mathbf{x})$ is a function of viewing direction $\boldsymbol{\omega}_o$, and models view-dependent effects such as specularity.

3. Method

In this section, we describe our PhySG pipeline and its three major components: (1) geometry modeling, (2) appearance modeling, and (3) forward rendering. These components are designed to be differentiable, so that the whole pipeline can be optimized end-to-end from multiple images captured under static illumination.

Geometry modeling. Motivated by the success of signed distance functions (SDFs) for representing shape [54, 15, 34, 24, 58], we adopt SDFs as our geometric representation. SDFs support ray casting via sphere tracing, are differentiable, and automatically satisfy the constraint between shape and surface normal—the surface normal is exactly the gradient of the SDF. We represent SDFs with MLPs (rather than voxel grids) for their memory efficiency and infinite resolution [34]. Concretely, let $S(\mathbf{x};\boldsymbol{\Theta})$ be our SDF,[†] where $\mathbf{x}$ is a 3D point and $\boldsymbol{\Theta}$ are the MLP weights. Our MLP consists of 8 nonlinear layers of width 512, with a skip connection at 4^{th} layer. To allow the MLP to model high-frequency geometric detail, we use positional encoding with 6 frequency components to encode the location of a 3D point [45, 29].[‡] An alternate to SDFs is to use occupancy fields [27, 31], but ray tracing through occupancy fields is much slower, requiring root-finding to locate the surface. While occupancy fields require over 100 MLP evaluations per cast ray [31], for SDFs, the MLP only needs to be evaluated ∼10 times via sphere tracing.

To render the pixel color for a camera ray, we first find the ray's point of intersection with the SDF by starting from the ray's intersection with the object bounding box and marching along the ray via sphere tracing as in [54], where the size of each step is the signed distance at the current location. The intersection point's location $\mathbf{x}$ and surface normal $\mathbf{n} = \nabla_{\mathbf{x}} S$ are then used by our appearance component to render the pixel's color. Hence, to optimize the geometry, gradients must back-propagate through both $\mathbf{x}$ and $\mathbf{n}$ to the SDF parameters $\boldsymbol{\Theta}$. Back-propagating through the surface normal $\mathbf{n}$ is straightforward via auto-differentiation [36]. To back-propagate through the surface location $\mathbf{x}$, we use the implicit differentiation method presented in [31, 54]. Note however that the sphere tracing algorithm itself need not be differentiable, hence it is very memory-efficient.

Appearance modeling. To model a single-material specular object in a way consistent with the rendering equation (see Eq. 1), we use two optimizable components: (1) an environment map, and (2) BRDF consisting of spatially varying diffuse albedo and a shared monochrome isotropic specular component. Note however that we do not model self-occlusion or indirect illumination. The hemispherical integral in the rendering equation generally does not have a closed-form expression, necessitating expensive Monte-Carlo methods for numeric evaluation. However, in our setting of glossy material and distant direct illumination, we can utilize spherical Gaussians (SGs) [53] to efficiently approximate the rendering equation in closed form.

An n-dimensional spherical Gaussian (SG) is a spherical function that takes the form [48]:

$$G(\boldsymbol{\nu};\boldsymbol{\xi},\lambda,\boldsymbol{\mu}) = \boldsymbol{\mu}\, e^{\lambda(\boldsymbol{\nu}\cdot\boldsymbol{\xi}-1)}, \quad (2)$$

where $\boldsymbol{\nu} \in \mathbb{S}^2$ is the function input, $\boldsymbol{\xi} \in \mathbb{S}^2$ is the lobe axis, $\lambda \in \mathbb{R}_+$ is the lobe sharpness, and $\boldsymbol{\mu} \in \mathbb{R}^n_+$ is the lobe amplitude. Our environment map $L_i(\boldsymbol{\omega}_i;\mathbf{x}) = L(\boldsymbol{\omega}_i)$[§] is then represented with a mixture of $M = 128$ SGs:

$$L_i(\boldsymbol{\omega}_i) = \sum_{k=1}^{M} G(\boldsymbol{\omega}_i;\boldsymbol{\xi}_k,\lambda_k,\boldsymbol{\mu}_k). \quad (3)$$

We represent the spatially-varying diffuse albedo with an MLP mapping a surface point $\mathbf{x}$ to a color vector $\mathbf{a}$, i.e., $\mathbf{a}(\mathbf{x};\boldsymbol{\Phi})$. Positional encoding is also applied to fit high-frequency texture details [45, 25]. Specifically, we use an MLP with 4 nonlinear layers of width 512, and encode location $\mathbf{x}$ with 10 frequencies. As for the shared specular component, we use the same simplified Disney BRDF model [9, 17] as in prior work [8, 22]:

$$f_s(\boldsymbol{\omega}_o,\boldsymbol{\omega}_i) = \mathcal{M}(\boldsymbol{\omega}_o,\boldsymbol{\omega}_i)\, \mathcal{D}(\mathbf{h}), \quad (4)$$

[*] Viewing direction $\boldsymbol{\omega}_o$, lighting direction $\boldsymbol{\omega}_i$ and surface normal $\mathbf{n}$ are all assumed to point away from the scene.

[†] We assume SDF>0 is an object's exterior, while SDF<0 is its interior.

[‡] Using L frequency components, positional encoding maps vector $\mathbf{p}$ to $(\mathbf{p}, \sin(2^0\mathbf{p}), \cos(2^0\mathbf{p}), \ldots, \sin(2^{L-1}\mathbf{p}), \cos(2^{L-1}\mathbf{p}))$.

[§] We drop the location $\mathbf{x}$ due to the distant illumination assumption.

where $\mathbf{h} = (\boldsymbol{\omega}_o + \boldsymbol{\omega}_i)/\|\boldsymbol{\omega}_o + \boldsymbol{\omega}_i\|_2$, $\mathcal{M}$ accounts for the Fresnel and shadowing effects, and $\mathcal{D}$ is the normalized distribution function. We include details of $\mathcal{M}$ and $\mathcal{D}$ in the supplemental material. We represent $\mathcal{D}$ with a single SG:

$$\mathcal{D}(\mathbf{h}) = G(\mathbf{h}; \boldsymbol{\xi}, \lambda, \boldsymbol{\mu}). \tag{5}$$

Our isotropic specular BRDF assumption results in $\boldsymbol{\xi}$ aligning with surface normal, i.e., $\boldsymbol{\xi} = \mathbf{n}$, while the monochrome assumption makes the three numbers in $\boldsymbol{\mu}$ identical.

To evaluate the rendering equation at a point $\mathbf{x}$ with surface normal $\mathbf{n}$ viewed along direction $\boldsymbol{\omega}_o$, $\mathcal{D}$ must be spherically warped, while $\mathcal{M}$ must be approximated by a constant at this specific location $\mathbf{x}$ [48]:

$$\mathcal{D}_{\mathbf{x}}(\mathbf{h}) = G(\mathbf{h}; \mathbf{n}, \frac{\lambda}{4\mathbf{h} \cdot \boldsymbol{\omega}_o}, \boldsymbol{\mu}), \tag{6}$$

$$\mathcal{M}_{\mathbf{x}}(\boldsymbol{\omega}_o, \boldsymbol{\omega}_i) \approx \mathcal{M}(\boldsymbol{\omega}_o, 2(\boldsymbol{\omega}_o \cdot \mathbf{n})\mathbf{n} - \boldsymbol{\omega}_o). \tag{7}$$

Hence for the point $\mathbf{x}$, we have:

$$f_s(\boldsymbol{\omega}_o, \boldsymbol{\omega}_i; \mathbf{x}) = G(\mathbf{h}; \mathbf{n}, \frac{\lambda}{4\mathbf{h} \cdot \boldsymbol{\omega}_o}, \mathcal{M}_{\mathbf{x}}\boldsymbol{\mu}). \tag{8}$$

Now that both $L_i(\boldsymbol{\omega}_i)$ and $f_r(\boldsymbol{\omega}_o, \boldsymbol{\omega}_i; \mathbf{x}) = \frac{\mathbf{a}}{\pi} + f_s(\boldsymbol{\omega}_o, \boldsymbol{\omega}_i; \mathbf{x})$ in the rendering equation are represented with SGs, we further approximate the remaining term $\boldsymbol{\omega}_i \cdot \mathbf{n}$ with a SG [26]:

$$\boldsymbol{\omega}_i \cdot \mathbf{n} \approx G(\boldsymbol{\omega}_i; 0.0315, \mathbf{n}, 32.7080) - 31.7003. \tag{9}$$

Finally, we integrate the multiplication of these SGs in closed-form [26] to compute the observed color $L_o(\boldsymbol{\omega}_o; \mathbf{x})$.

To summarize, the optimizable parameters in our appearance component are $\{\boldsymbol{\xi}_k, \lambda_k, \boldsymbol{\mu}_k\}_{k=1}^{M}$, $\{\lambda, \boldsymbol{\mu}\}$, and $\boldsymbol{\Phi}$, which are parameters of the environment map, specular BRDF, and spatially-varying diffuse albedo, respectively.

Forward rendering. Given our geometric and appearance components, we perform forward rendering of a ray's color as follows: (1) use sphere tracing to find the intersection point $\mathbf{x}$ between the ray $\mathbf{r} = \mathbf{o} + t\mathbf{d}$ and the surface $S(\mathbf{x}; \boldsymbol{\Theta})$; (2) compute the surface normal $\mathbf{n} = \nabla_{\mathbf{x}} S$ at $\mathbf{x}$ via automatic differentiation [36]; (3) compute the diffuse albedo $\mathbf{a}(\mathbf{x}; \boldsymbol{\Phi})$ at $\mathbf{x}$; (4) use the surface normal $\mathbf{n}$, environment map $\{\boldsymbol{\xi}_k, \lambda_k, \boldsymbol{\mu}_k\}_{k=1}^{M}$, diffuse albedo $\mathbf{a}$, specular BRDF $\{\lambda, \boldsymbol{\mu}\}$, and viewing direction $\mathbf{d}$, to compute the color for ray $\mathbf{r}$ by evaluating the rendering equation in closed form with our SG approximation. This procedure is illustrated in Fig. 2.

We now show that our pipeline is fully differentiable, in that its output (the rendered color) is differentiable w.r.t. all the optimizable parameters. First, the rendered color is differentiable w.r.t. the variables $\mathbf{n}$, $\{\boldsymbol{\xi}_k, \lambda_k, \boldsymbol{\mu}_k\}_{k=1}^{M}$, $\mathbf{a}$, $\{\lambda, \boldsymbol{\mu}\}$ in step (4), because the SG renderer is simply the closed-form integration of spherical Gaussians. Since the diffuse albedo $\mathbf{a} = \mathbf{a}(\mathbf{x}; \boldsymbol{\Phi})$ is an MLP in step (3), the rendered color is differentiable w.r.t. $\mathbf{x}$ and $\boldsymbol{\Phi}$ by the chain rule. For our geometric model, we have shown that there exist gradients of both the surface location $\mathbf{x}$ and surface normal $\mathbf{n}$ w.r.t. the SDF parameters $\boldsymbol{\Theta}$. Thus by the chain rule, the rendered color is differentiable w.r.t. $\boldsymbol{\Theta}$ as well.

Loss functions. To optimize parameters given a set of images, we render images from the same viewpoints as the input images, and compute an ℓ_1 image reconstruction loss. We also enforce non-negative minimum SDF values along non-object pixel rays indicated by object segmentation masks, and regularize the SDF's gradient to have unit norm [12]. Concretely, at each training iteration, we first randomly sample a batch of pixels consisting of: object pixels $\mathbf{r}_i^{obj}$ with ground-truth color $\{\mathbf{c}_i^{gt}\}_{i=1}^{N_{obj}}$, and non-object pixels $\{\mathbf{r}_i^{nobj}\}_{i=1}^{N_{nobj}}$. Then we render colors $\mathbf{c}_i^{obj}$ for $\mathbf{r}_i^{obj}$, while finding the minimal SDF value S_i^{nobj} along camera rays $\mathbf{r}_i^{nobj}$ by taking the minimal SDF value among 100 points uniformly lying on the ray segment inside object bounding box. We also randomly sample $\{\mathbf{x}_i\}_{i=1}^{N_x}$ inside the object bounding box. Our full loss is:

$$\begin{aligned}\ell = & \frac{1}{N_{obj}} \sum_{i=1}^{N_{obj}} \left\| \mathbf{c}_i^{obj} - \mathbf{c}_i^{gt} \right\|_1 \\ & + \beta_1 \frac{1}{N_{nobj}} \sum_{i=1}^{N_{nobj}} \frac{\ln(1 + e^{-\alpha S_i^{nobj}})}{\alpha} \\ & + \beta_2 \frac{1}{N_x} \sum_{i=1}^{N_x} \left\| \|\nabla_{\mathbf{x}_i} S\|_2 - 1 \right\|_2^2,\end{aligned} \tag{10}$$

where $\frac{\ln(1+e^{-\alpha S_i^{nobj}})}{\alpha}, \alpha{>}0$ is a smooth approximation of a horizontally flipped ReLU $\max\{-S_i^{nobj}, 0\}$ (larger α yields tighter approximation); and β_1 and β_2 are weights balancing different loss terms. We set $\beta_1{=}100, \beta_2{=}0.1, N_{obj}{+}N_{nobj}{=}2048, N_x{=}1024$ in our experiments; α gradually grows from 50 to 1600 as suggested in [54]. Finally, rather than sampling $N_{obj} + N_{nobj}$ independent pixels, we sample $\frac{N_{obj}+N_{nobj}}{4}$ patches of size 2×2, and add an additional loss term to penalize the variance of surface normals inside patches consisting only of object pixels. We set the weight for this smoothness loss to 10. We train on a single 12GB NVIDIA GPU for 250k iterations.

Initialization. The SDF weights $\boldsymbol{\Theta}$ are initialized using the method of [12] such that the initial shape is roughly a sphere. The diffuse albedo $\mathbf{a}(\mathbf{x}; \boldsymbol{\Phi})$ is initialized such that predicted albedo is ∼0.5 at all locations inside the object bounding box. For the specular BRDF, the initial lobe sharpness λ is randomly drawn from $[95, 125]$, while the initial specular albedo $\boldsymbol{\mu}$ is randomly drawn from $[0.18, 0.26]$. For the environment map, the lobes are initialized to distribute uniformly on the unit sphere using a spherical Fibonacci lattice [18], with monochrome colors; we also scale the randomly initialized lobes' amplitude so that the initial rendered pixel intensity output by our pipeline is $\sim$0.5. In addition, since different captures can vary significantly in exposure, we scale all input images of an object with the same constant such that the median intensity of all scaled images is 0.5. We empirically find that if the initial environment map is too bright or too dark, the diffuse albedo MLP sometimes gets stuck, predicting all zeros or ones during training. Our proposed initialization addresses this issue.

4. Experiments

We perform experiments on both synthetic and real-world data to validate our PhySG pipeline.

4.1. Synthetic data

To create synthetic data, we use objects from [10, 57]; for each object, we render 200 images with colored environmental lighting using the Mitsuba renderer [14], 100 each for training and testing. To test the extrapolation capability of different algorithms, the test images are distributed inside a 70-degree cone around the object's north pole, while the training images cover the rest of the upper hemisphere (see Fig. 3). We use the Ward BRDF model [49] included in Mitsuba, and set the specular albedo to $(0.3, 0.3, 0.3)$ and roughness values along the tangent and bitangent directions to 0.05. Ground truth surface normal maps and diffuse albedos are also rendered to quantitatively evaluate our inverse rendering results. To evaluate the relighting performance of our pipeline, we also render the same object with two other environment maps in Mitsuba to serve as ground truth.

We report image quality metrics: LPIPS [56], SSIM, and PSNR on held-out test viewpoints. As there is an inherent scale ambiguity in inverse rendering problems, before computing the metrics, we align our predicted image $\hat{I}$ to ground-truth I via channel-wise scale factors. Specifically, let $\hat{I}_r, I_r$ denote the red channel of $\hat{I}, I$, respectively. Then the scale factor s_r for the red channel is estimated via:

$$s_r = \text{Median}(I_r / \hat{I}_r). \tag{11}$$

The green and blue channels are scaled similarly.

As shown by the quantitative evaluation in Tab. 1 and qualitative evaluation in Fig. 7, our synthesized novel test views, estimated diffuse albedo and surface normal, as well as material editing and relighting results closely match the ground truth on synthetic data, despite the test viewpoints representing a difficult view extrapolation scenario. Note especially that our method correctly extrapolates the challenging specular highlight in Fig. 7.

	↓LPIPS	↑SSIM	↑PSNR
Diffuse albedo	0.0339	0.989	33.43
Novel view	0.0170	0.990	35.93
Relighting	0.0227	0.988	33.25
Surface normal error (°)		2.528	

Table 1: Quantitative evaluation of our inverse rendering results on the synthetic dataset. We compare predictions of our pipeline against the ground-truth rendered with Mitsuba. Since there is a scale ambiguity in inverse rendering problems, we align our predictions to ground-truth before evaluating the metrics (see Eq. 11 for details). We also report the average angular error of our estimated surface normals.

	↓Surface Normal Error (°)	↓Chamfer L_1
Ours	2.528	0.00142
NeRF	36.05	0.01650
IDR	**2.207**	**0.00136**
DVR	38.90	0.13800

Table 2: Evaluation of recovered geometry on synthetic data. We report avg. surface normal error on test views and L_1 Chamfer (point-to-mesh) distance between estimated and GT meshes (normalized to have a unit bounding box).

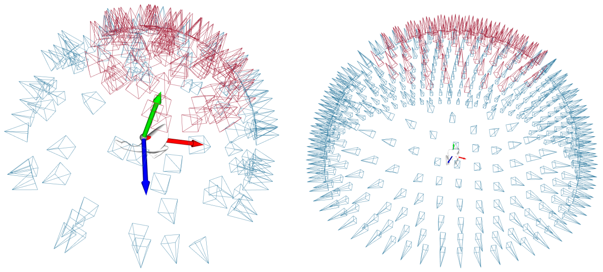

Figure 3: Camera setup for synthetic duck (left) [10] and real SLF fish (right) [50]. Blue for training, red for testing. Note that testing viewpoints deviate significantly from training ones. Latitudes of testing cameras are above 55 degrees, while training cameras are below this latitude threshold.

4.2. Real-world data

We test our method on multiple real-world captures from datasets including SLF [50], DeepVoxels [43], Bag of Chips [35] and DTU [2]. The objects in these captures are glossy and the illumination is static across different views.

SLF dataset. We use the glossy *fish* from [50]. This dataset is captured with a gantry in a lab-controlled environment. The cameras are distributed on a hemisphere around the center object. We discard images in which the center object has noticeable shadows cast by the gantry or is partly occluded by the platform. Then we split the data according to the cameras' latitudes, with test cameras' latitudes above 55 degrees,

	Synthetic Kitty			Synthetic Bear			Synthetic Duck			Synthetic Mouse			SLF fish [50]			Chips Corncho1 [35]		
# train/test (HxW)	100/100 (512x512)			100/100 (512x512)			100/100 (512x512)			100/100 (512x512)			419/106 (480x640)			1661/345 (480x640)		
	↓LPIPS	↑SSIM	↑PSNR	↓LPIPS	↑SSIM	↑PSNR	↓LPIPS	↑SSIM	↑PSNR	↓LPIPS	↑SSIM	↑PSNR	↓LPIPS	↑SSIM	↑PSNR	↓LPIPS	↑SSIM	↑PSNR
Ours	**0.0189**	**0.989**	**36.45**	0.0200	**0.987**	33.76	**0.0081**	**0.994**	**38.70**	**0.0209**	**0.987**	**34.81**	0.0142	0.969	30.27	0.0477	0.969	27.44
NeRF [29]	0.0534	0.971	30.75	0.0493	0.964	28.17	0.0338	0.976	29.72	0.0772	0.948	26.30	0.0255	0.966	28.90	0.0478	0.969	27.64
IDR [54]	0.0202	0.987	35.21	**0.0169**	0.986	**33.88**	0.0121	0.991	36.24	0.0259	0.983	32.67	**0.0129**	**0.977**	**31.48**	**0.0414**	**0.975**	**27.92**
DVR [31]	0.132	0.926	24.69	0.124	0.911	19.24	0.100	0.944	25.78	0.165	0.903	22.37	0.0494	0.952	22.90	0.255	0.848	16.06

Table 3: We compare the novel test view quality of our method with that of NeRF [29], IDR [54] and DVR [31]. For synthetic data, HDR images are tonemapped with $I_{out} = I_{in}^{1/2.2}$ and clipped to $[0, 1]$ before computing the metrics. Note that the baseline methods model appearance as surface light field [50], hence they can not do editing/relighting like ours. On real-world data, our metric numbers are slightly worse than IDR — this is likely caused by the bias in our physics-based appearance modeling that does not align perfectly with real material properties, while surface light field modeling has little bias but precludes material editing and relighting.

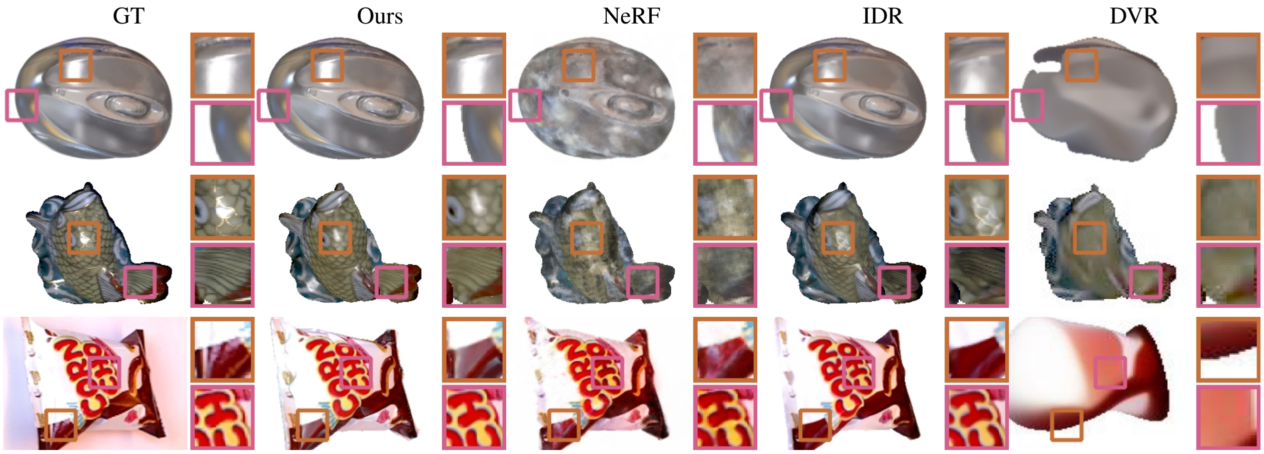

Figure 4: On synthetic and real data, we qualitatively compare our novel view extrapolation quality with most related neural rendering techniques: NeRF [29], IDR [54] and DVR [31]. Our method extrapolates the specularity more reasonably than the baseline methods thanks to our physics-based modeling of the approximate light transport.

as shown in Fig. 3. We render object segmentation masks from the provided laser-scanned meshes.

DeepVoxels. We use the glossy *globe* and *coffee* objects from DeepVoxels [43]. These are real-world handheld captures. The camera parameters are recovered with COLMAP [41, 42]. We use background removal tools [1] to automatically generate the object segmentation masks. We leave ∼25% images for testing.

Bag of Chips. We use the glossy *cans* and *corncho1* data from this dataset [34]. We render object segmentation masks from the provided mesh scanned by RGBD sensors. We leave ∼25% images for testing.

DTU dataset. We use the shiny *scan114 buddha* object from this dataset [2]. We discard images in which the camera casts noticeable shadows on the object. The object segmentation masks are automatically generated using background removal tools [1]. We leave ∼25% images for testing.

Our inverse rendering results are qualitatively shown in Fig. 5. Video demos are shown in our supplemental material. We can see that our pipeline generates photo-realistic novel views, plausible material editing and relighting results.

4.3. Comparison with baselines

We could not identify prior work tackling exactly the same problem as us: simultaneously reconstructing lighting, material, and geometry from scratch from 2D images captured under static illumination. Hence, we compare our PhySG to the most related neural rendering approaches, including NeRF [25], IDR [54] and DVR [31], in terms of novel view extrapolation quality.

Like PhySG, these approaches can also be trained end-to-end from 2D image supervision only, but they differ from our method in the way that appearance is modelled. Loosely speaking, they all model appearance as an MLP-represented surface light field. In particular, NeRF maps location $\mathbf{x}$ and viewing direction $\mathbf{d}$ to a color; IDR maps $\mathbf{x}$, $\mathbf{d}$, and surface normal $\mathbf{n}$ to a color, and DVR only takes location $\mathbf{x}$. Tab. 3 and Fig. 4 compares different methods on synthetic and real data. NeRF does poorly in view extrapolation because its volumetric representation does not concentrate colors around surfaces as well as surface-based approaches. Although DVR uses a surface-based shape model, its model does not support view-dependent effects, and so it fails to model this kind of glossy data. Compared with DVR, IDR models view-dependence and does a good job in view extrapolation.

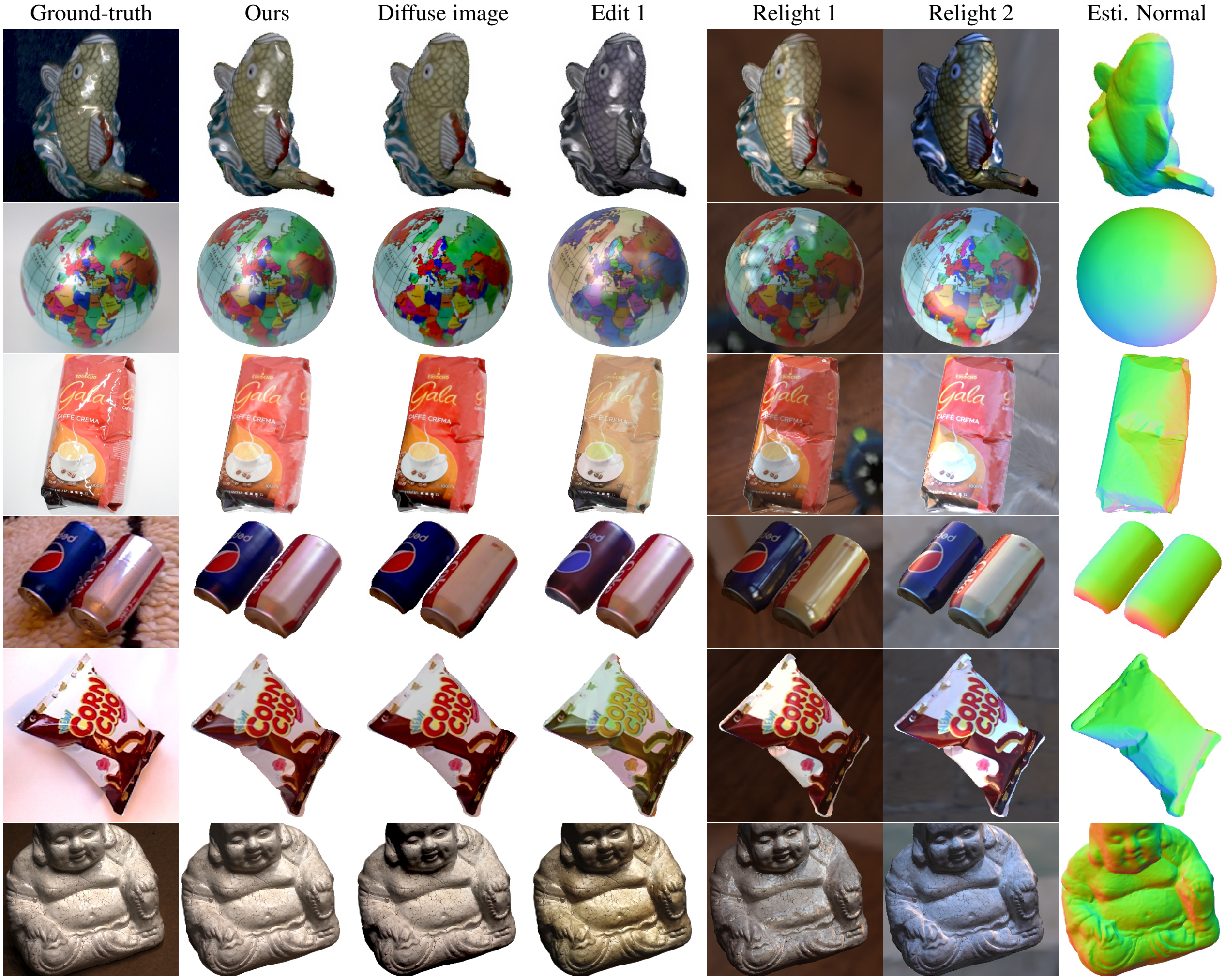

Figure 5: With our pipeline, we can edit the materials and lighting of the real-world captures. For several input captures, we show from left to right: a real photo in the test set, our synthesized image, estimated diffuse image, editing results by painting diffuse albedo, relighting results under two novel environmental illuminations, and estimated surface normal.

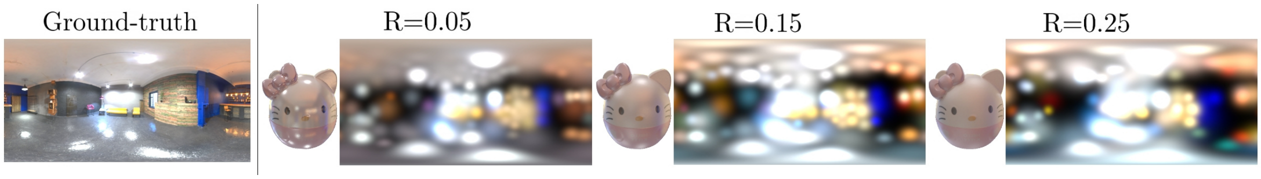

Figure 6: Ground truth and our reconstructed environment maps for the synthetic Kitty data with varying Ward BRDF roughness R. For each roughness setting, an example training image rendered by Mitsuba [14] is also shown. For rough surfaces (R=0.25), PhySG still recovers an environment map that resembles the ground truth, though blurrier. Nonetheless, this is sufficient to reconstruct the material accurately.

However, it still has trouble synthesizing specular highlights, due to the lack of a physical model of appearance. In contrast, our method models such highlights well. As for geometry, our estimated geometry is nearly as good as IDR's (and much better than other baselines) shown in Tab. 2, while we also allow for relighting and material editing.

We also tried redner [19], a Monte Carlo differentiable renderer representing shapes as meshes. We let redner jointly optimize lighting, texture, BRDF and geometry with an image reconstruction loss. We initialized the mesh a sphere at the beginning of training, and found that redner got stuck in the initial mesh and failed to converge.

4.4. Robustness to material roughness

Our method relies on specular highlights to estimate the lighting and material properties. As a result, if the material of interest is purely Lambertian, we face a lighting-texture ambiguity and cannot recover lighting without additional

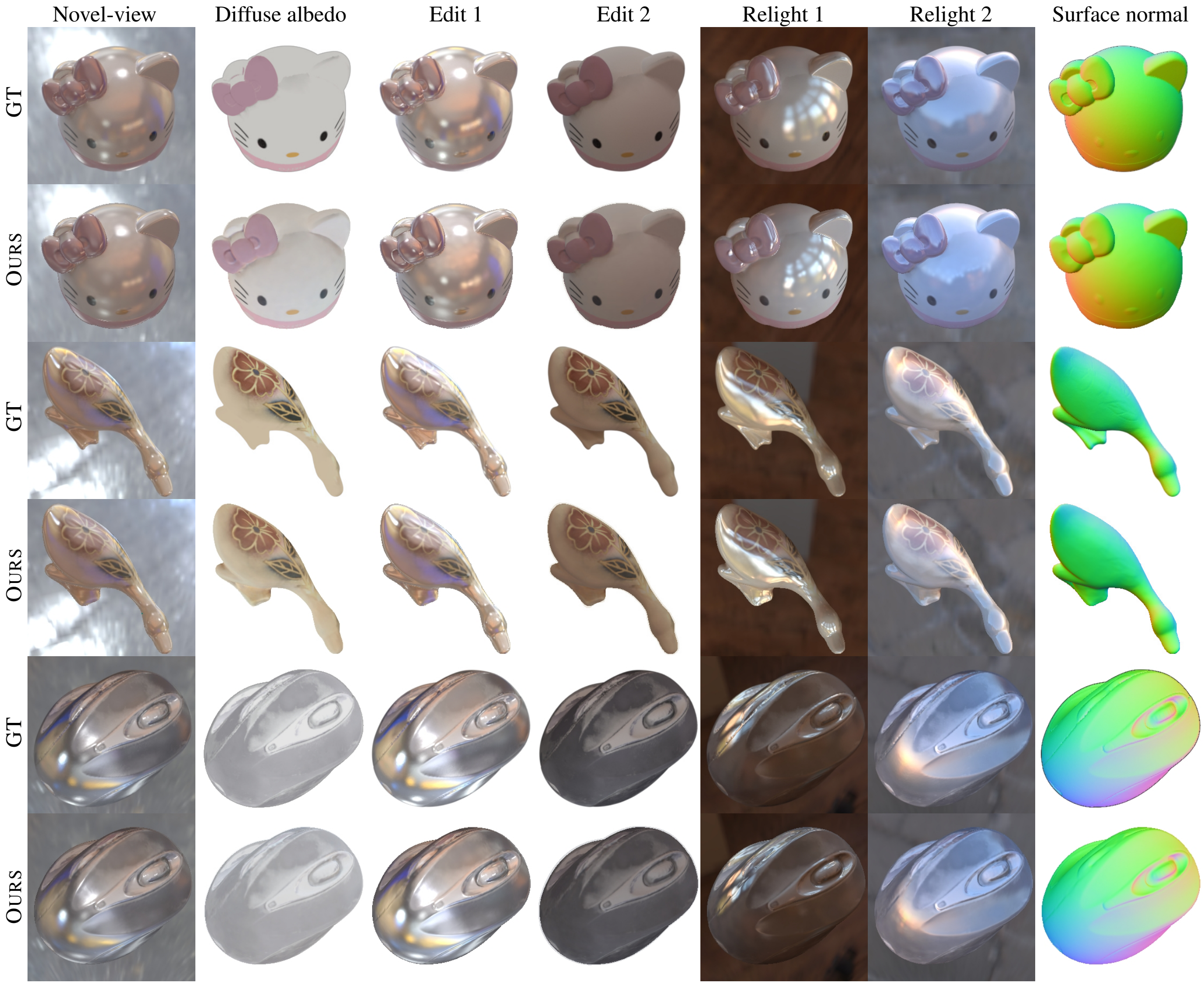

Figure 7: Results of our pipeline on synthetic data. For a novel test view, we compare our predicted image, estimated diffuse albedo, specular BRDF editing results and relighting results to ground truth images rendered by Mitsuba [14]. Note that there is a scale ambiguity in inverse rendering problems; hence we align our estimated diffuse albedo to the ground truth for visualization here (see Eq. 11 for details of the alignment we apply). More examples are available in the supplemental material.

priors. We empirically test the robustness of our pipeline to material roughness on synthetic data. As shown in Fig. 6, even from very weak specular highlights, our method can reconstruct a reasonable-looking environment map.

5. Conclusion

We proposed PhySG, an end-the-end inverse rendering pipeline that uses physics-based differentiable rendering. PhySG uses signed distance functions (SDFs) and spherical Gaussians (SG) to represent geometry and appearance, respectively. We show that PhySG can jointly recover environment maps, material BRDFs and geometry from multi-view inputs captured under static illumination, enabling physics-based material editing and relighting.

Limitations. Our method has a few limitations that can be the subject of future work. First, indirect illumination is not modelled by our SG approximation of the rendering equation, which limits our method to object-level data. To lift this restriction and extend to scene-level data, differentiable path tracing combined with deferred neural textures [11, 47] can be explored, where only rough geometry is required for guidance. Second, we assume constant and monochrome specular BRDFs (with spatially-varying diffuse components). This assumption is due to the scale ambiguity between illumination and reflectance. Similar to intrinsic image decomposition, learning-based priors could help alleviate such ambiguities. Last, our work can also be extended to handle anisotropic or data-driven BRDFs, e.g., by fitting a mixture of anisotropic SGs [52].

Acknowledgements. This work was supported in part by the National Science Foundation (IIS-2008313, CHS-1900783, CHS-1930755).

References

[1] Remove image background. https://www.remove.bg/. Accessed: 2020-10-15.

[2] Henrik Aanæs, Rasmus Ramsbøl Jensen, George Vogiatzis, Engin Tola, and Anders Bjorholm Dahl. Large-scale data for multiple-view stereopsis. *Int. J. Comput. Vis.*, pages 1–16, 2016.

[3] Jens Ackermann and Michael Goesele. A survey of photometric stereo techniques. *Foundations and Trends® in Computer Graphics and Vision*, 9(3-4):149–254, 2015.

[4] Dejan Azinovic, Tzu-Mao Li, Anton Kaplanyan, and Matthias Nießner. Inverse path tracing for joint material and lighting estimation. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 2447–2456, 2019.

[5] Jonathan T Barron and Jitendra Malik. Shape, illumination, and reflectance from shading. *IEEE Trans. Pattern Anal. Mach. Intell.*, 37(8):1670–1687, 2014.

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

[7] Sai Bi, Zexiang Xu, Kalyan Sunkavalli, Miloš Hašan, Yannick Hold-Geoffroy, David Kriegman, and Ravi Ramamoorthi. Deep reflectance volumes: Relightable reconstructions from multi-view photometric images. *arXiv preprint arXiv:2007.09892*, 2020.

[8] Sai Bi, Zexiang Xu, Kalyan Sunkavalli, David Kriegman, and Ravi Ramamoorthi. Deep 3d capture: Geometry and reflectance from sparse multi-view images. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 5960–5969, 2020.

[9] Brent Burley and Walt Disney Animation Studios. Physically-based shading at disney. In *SIGGRAPH*, volume 2012, pages 1–7. vol. 2012, 2012.

[10] Yue Dong, Guojun Chen, Pieter Peers, Jiawan Zhang, and Xin Tong. Appearance-from-motion: Recovering spatially varying surface reflectance under unknown lighting. *ACM Trans. Graph.*, 33(6):1–12, 2014.

[11] Duan Gao, Guojun Chen, Yue Dong, Pieter Peers, Kun Xu, and Xin Tong. Deferred neural lighting: free-viewpoint relighting from unstructured photographs. *ACM Transactions on Graphics (TOG)*, 39(6):1–15, 2020.

[12] Amos Gropp, Lior Yariv, Niv Haim, Matan Atzmon, and Yaron Lipman. Implicit geometric regularization for learning shapes. *arXiv preprint arXiv:2002.10099*, 2020.

[13] Tom Haber, Christian Fuchs, Philippe Bekaer, Hans-Peter Seidel, Michael Goesele, and Hendrik PA Lensch. Relighting objects from image collections. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 627–634. IEEE, 2009.

[14] Wenzel Jakob. Mitsuba renderer, 2010. http://www.mitsuba-renderer.org.

[15] Yue Jiang, Dantong Ji, Zhizhong Han, and Matthias Zwicker. Sdfdiff: Differentiable rendering of signed distance fields for 3d shape optimization. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 1251–1261, 2020.

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

[17] Brian Karis and Epic Games. Real shading in unreal engine 4.

[18] Benjamin Keinert, Matthias Innmann, Michael Sänger, and Marc Stamminger. Spherical fibonacci mapping. *ACM Trans. Graph.*, 34(6):1–7, 2015.

[19] Tzu-Mao Li, Miika Aittala, Frédo Durand, and Jaakko Lehtinen. Differentiable monte carlo ray tracing through edge sampling. *ACM Trans. Graph. (Proc. SIGGRAPH Asia)*, 37(6):222:1–222:11, 2018.

[20] Zhengqin Li, Mohammad Shafiei, Ravi Ramamoorthi, Kalyan Sunkavalli, and Manmohan Chandraker. Inverse rendering for complex indoor scenes: Shape, spatially-varying lighting and svbrdf from a single image. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2475–2484, 2020.

[21] Zhengqi Li, Wenqi Xian, Abe Davis, and Noah Snavely. Crowdsampling the plenoptic function. In *Eur. Conf. Comput. Vis.*, 2020.

[22] Zhengqin Li, Zexiang Xu, Ravi Ramamoorthi, Kalyan Sunkavalli, and Manmohan Chandraker. Learning to reconstruct shape and spatially-varying reflectance from a single image. *ACM Trans. Graph.*, 37(6):1–11, 2018.

[23] Stephen Lombardi and Ko Nishino. Radiometric scene decomposition: Scene reflectance, illumination, and geometry from rgb-d images. In *3DV*, pages 305–313. IEEE, 2016.

[24] Robert Maier, Kihwan Kim, Daniel Cremers, Jan Kautz, and Matthias Nießner. Intrinsic3D: High-quality 3D reconstruction by joint appearance and geometry optimization with spatially-varying lighting. In *Int. Conf. Comput. Vis.*, 2017.

[25] Ricardo Martin-Brualla, Noha Radwan, Mehdi SM Sajjadi, Jonathan T Barron, Alexey Dosovitskiy, and Daniel Duckworth. Nerf in the wild: Neural radiance fields for unconstrained photo collections. *arXiv preprint arXiv:2008.02268*, 2020.

[26] Julian Meder and Beat Brüderlin. Hemispherical gaussians for accurate light integration. In Leszek J. Chmielewski, Ryszard Kozera, Arkadiusz Orłowski, Konrad Wojciechowski, Alfred M. Bruckstein, and Nicolai Petkov, editors, *Computer Vision and Graphics*, pages 3–15, Cham, 2018. Springer International Publishing.

[27] Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy networks: Learning 3d reconstruction in function space. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 4460–4470, 2019.

[28] Moustafa Meshry, Dan B Goldman, Sameh Khamis, Hugues Hoppe, Rohit Pandey, Noah Snavely, and Ricardo Martin-Brualla. Neural rerendering in the wild. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 6878–6887, 2019.

[29] 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 *Eur. Conf. Comput. Vis.*, 2020.

[30] Giljoo Nam, Joo Ho Lee, Diego Gutierrez, and Min H Kim. Practical svbrdf acquisition of 3d objects with unstructured flash photography. *ACM Trans. Graph.*, 37(6):1–12, 2018.

[31] Michael Niemeyer, Lars Mescheder, Michael Oechsle, and Andreas Geiger. Differentiable volumetric rendering: Learning implicit 3d representations without 3d supervision. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 3504–3515, 2020.

[32] Michael Oechsle, Michael Niemeyer, Lars Mescheder, Thilo Strauss, and Andreas Geiger. Learning implicit surface light fields. *arXiv preprint arXiv:2003.12406*, 2020.

[33] Geoffrey Oxholm and Ko Nishino. Multiview shape and reflectance from natural illumination. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 2155–2162, 2014.

[34] Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. Deepsdf: Learning continuous signed distance functions for shape representation. In *IEEE Conf. Comput. Vis. Pattern Recog.*, June 2019.

[35] Jeong Joon Park, Aleksander Holynski, and Steve Seitz. Seeing the world in a bag of chips. *IEEE Conf. Comput. Vis. Pattern Recog.*, 2020.

[36] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. PyTorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Adv. Neural Inform. Process. Syst.*, pages 8024–8035. 2019.

[37] Ravi Ramamoorthi and Pat Hanrahan. An efficient representation for irradiance environment maps. In *SIGGRAPH*, pages 497–500, 2001.

[38] Ravi Ramamoorthi and Pat Hanrahan. A signal-processing framework for reflection. *ACM Trans. Graph.*, 23(4):1004–1042, Oct. 2004.

[39] Fabiano Romeiro and Todd Zickler. Blind reflectometry. In Kostas Daniilidis, Petros Maragos, and Nikos Paragios, editors, *Computer Vision – ECCV 2010*, pages 45–58, Berlin, Heidelberg, 2010. Springer Berlin Heidelberg.

[40] Carolin Schmitt, Simon Donne, Gernot Riegler, Vladlen Koltun, and Andreas Geiger. On joint estimation of pose, geometry and svbrdf from a handheld scanner. In *IEEE Conf. Comput. Vis. Pattern Recog.*, June 2020.

[41] Johannes Lutz Schönberger and Jan-Michael Frahm. Structure-from-motion revisited. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2016.

[42] Johannes Lutz Schönberger, Enliang Zheng, Marc Pollefeys, and Jan-Michael Frahm. Pixelwise view selection for unstructured multi-view stereo. In *Eur. Conf. Comput. Vis.*, 2016.

[43] Vincent Sitzmann, Justus Thies, Felix Heide, Matthias Nießner, Gordon Wetzstein, and Michael Zollhofer. Deepvoxels: Learning persistent 3d feature embeddings. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 2437–2446, 2019.

[44] Vincent Sitzmann, Michael Zollhöfer, and Gordon Wetzstein. Scene representation networks: Continuous 3d-structure-aware neural scene representations. In *Adv. Neural Inform. Process. Syst.*, pages 1119–1130, 2019.

[45] Matthew Tancik, Pratul P Srinivasan, Ben Mildenhall, Sara Fridovich-Keil, Nithin Raghavan, Utkarsh Singhal, Ravi Ramamoorthi, Jonathan T Barron, and Ren Ng. Fourier features let networks learn high frequency functions in low dimensional domains. *arXiv preprint arXiv:2006.10739*, 2020.

[46] Ayush Tewari, Ohad Fried, Justus Thies, Vincent Sitzmann, Stephen Lombardi, Kalyan Sunkavalli, Ricardo Martin-Brualla, Tomas Simon, Jason Saragih, Matthias Nießner, et al. State of the art on neural rendering. *arXiv preprint arXiv:2004.03805*, 2020.

[47] Justus Thies, Michael Zollhöfer, and Matthias Nießner. Deferred neural rendering: Image synthesis using neural textures. *ACM Trans. Graph.*, 38(4):1–12, 2019.

[48] Jiaping Wang, Peiran Ren, Minmin Gong, John Snyder, and Baining Guo. All-frequency rendering of dynamic, spatially-varying reflectance. In *SIGGRAPH Asia*, pages 1–10. 2009.

[49] Gregory J Ward. Measuring and modeling anisotropic reflection. In *SIGGRAPH*, pages 265–272, 1992.

[50] Daniel N Wood, Daniel I Azuma, Ken Aldinger, Brian Curless, Tom Duchamp, David H Salesin, and Werner Stuetzle. Surface light fields for 3d photography. In *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, pages 287–296, 2000.

[51] Rui Xia, Yue Dong, Pieter Peers, and Xin Tong. Recovering shape and spatially-varying surface reflectance under unknown illumination. *ACM Trans. Graph.*, 35(6):1–12, 2016.

[52] Kun Xu, Wei-Lun Sun, Zhao Dong, Dan-Yong Zhao, Run-Dong Wu, and Shi-Min Hu. Anisotropic spherical gaussians. *ACM Transactions on Graphics (TOG)*, 32(6):1–11, 2013.

[53] Ling-Qi Yan, Yahan Zhou, Kun Xu, and Rui Wang. Accurate translucent material rendering under spherical gaussian lights. *Computer Graphics Forum*, 31(7):2267–2276, 2012.

[54] Lior Yariv, Yoni Kasten, Dror Moran, Meirav Galun, Matan Atzmon, Ronen Basri, and Yaron Lipman. Multiview neural surface reconstruction with implicit lighting and material. In *Adv. Neural Inform. Process. Syst.*, 2020.

[55] Kai Zhang, Gernot Riegler, Noah Snavely, and Vladlen Koltun. Nerf++: Analyzing and improving neural radiance fields. *arXiv:2010.07492*, 2020.

[56] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2018.

[57] Zhiming Zhou, Guojun Chen, Yue Dong, David Wipf, Yong Yu, John Snyder, and Xin Tong. Sparse-as-possible svbrdf acquisition. *ACM Trans. Graph.*, 35, November 2016.

[58] Michael Zollhöfer, Angela Dai, Matthias Innmann, Chenglei Wu, Marc Stamminger, Christian Theobalt, and Matthias Nießner. Shading-based refinement on volumetric signed distance functions. *ACM Trans. Graph.*, 34(4):1–14, 2015.