Efficient Multi-View Inverse Rendering Using a Hybrid Differentiable Rendering Method

Xiangyang Zhu[1,2], **Yiling Pan**[1,2], **Bailin Deng**[3] and **Bin Wang**[1,2*]
[1]School of Software, Tsinghua University, China
[2]Beijing National Research Center for Information Science and Technology (BNRist), China
[3]School of Computer Science and Informatics, Cardiff University, UK
{zhuxy20, pyl16}@mails.tsinghua.edu.cn, DengB3@cardiff.ac.uk, wangbins@tsinghua.edu.cn

Abstract

Recovering the shape and appearance of real-world objects from natural 2D images is a long-standing and challenging inverse rendering problem. In this paper, we introduce a novel hybrid differentiable rendering method to efficiently reconstruct the 3D geometry and reflectance of a scene from multi-view images captured by conventional hand-held cameras. Our method follows an analysis-by-synthesis approach and consists of two phases. In the initialization phase, we use traditional SfM and MVS methods to reconstruct a virtual scene roughly matching the real scene. Then in the optimization phase, we adopt a hybrid approach to refine the geometry and reflectance, where the geometry is first optimized using an approximate differentiable rendering method, and the reflectance is optimized afterward using a physically-based differentiable rendering method. Our hybrid approach combines the efficiency of approximate methods with the high-quality results of physically-based methods. Extensive experiments on synthetic and real data demonstrate that our method can produce reconstructions with similar or higher quality than state-of-the-art methods while being more efficient.

1 Introduction

Inverse rendering, which recovers 3D geometry, reflection properties and illumination from 2D images, is a long-standing and challenging problem in computer graphics and vision [Patow and Pueyo, 2003]. Benefiting from the advances in deep neural networks, many deep learning-based methods learn to obtain the materials and plane normals of near-flat objects in a data-driven way [Li *et al.*, 2018b; Aittala *et al.*, 2016; Gao *et al.*, 2019]. However, these methods struggle to deal with complex geometry, resulting in limited application in practice. Some other deep learning-based methods use 3D geometric representations such as signed distance fields [Yariv *et al.*, 2020; Zhang *et al.*, 2021b], tetrahedral meshes [Munkberg *et al.*, 2021], and occupancy functions [Mildenhall *et al.*, 2020; Zhang *et al.*, 2021c; Boss *et al.*, 2021; Srinivasan *et al.*, 2021; Zhang *et al.*, 2021a; Zhang *et al.*, 2022], to handle more complex geometries. However, such geometry representations may require post-processing in order to be used in traditional graphics pipelines [Lorensen and Cline, 1987; Remelli *et al.*, 2020], which may cause material information loss and affect the rendering image quality.

The emergence of differentiable rendering techniques [Kato *et al.*, 2020] means that image loss can be back-propagated along the rendering pipeline to solve inverse rendering problems, which promotes the development of multi-view inverse rendering tasks using triangular meshes as the geometry representation. Approximate differentiable rendering methods [Loper and Black, 2014; Kato *et al.*, 2018; Liu *et al.*, 2019; Chen *et al.*, 2019] utilize simplified rendering processes and can solve the inverse rendering problem efficiently, but the simple material models they use usually lead to poor visual effects. On the contrary, physically-based differentiable rendering methods [Li *et al.*, 2018a; Li *et al.*, 2021] can reconstruct high-quality physically-based rendering (PBR) materials in the path tracing manner, but their realistic results come at a high computational cost. Although some methods have been proposed to accelerate the reconstruction process [Luan *et al.*, 2021], they often impose strong assumptions on the lighting conditions to narrow down the parameter search space, which hinders their wide application.

In this paper, we propose a hybrid differentiable rendering method to reconstruct triangular meshes and PBR materials from multi-view real-world images captured by conventional hand-held cameras (*e.g.* mobile phones) with uncontrolled lighting conditions. Our method consists of two phases. In the initialization phase, we use traditional methods to reconstruct a rough triangular mesh for the scene. Then in the optimization phase, we take a hybrid approach to optimize the scene geometry and PBR materials, where we first optimize the geometry using an approximate method, followed by the PBR materials using a physically-based method. Our novel formulation benefits from both the efficiency of approximate methods and the high quality of physically-based methods. Extensive experiments show that our method achieves a significantly faster training and rendering speed than state-of-the-art methods, while achieving results of comparable or better quality.

In summary, our contributions include:

- We propose a novel hybrid differentiable rendering optimization method based on triangular meshes, which utilizes an

*Corresponding Author

(a) (b) (c) (d) (e)

Figure 1: (a) Our method takes as input a set of images obtained by conventional hand-held cameras from several viewpoints and gets a rough initial model (b) by traditional methods. Then, we perform a novel analysis-by-synthesis optimization to refine the model's shape and reflectance separately, yielding a high-quality 3D model. In (c) and (d), we show a re-rendering of the result under a novel viewpoint and environmental lighting. In addition, we can edit the material (e).

approximate differentiable rendering method to optimize the geometry, and a physically-based differentiable rendering method to get the PBR materials.

- The proposed pipeline is user-friendly and can work end-to-end. Furthermore, the optimized scene parameters can be easily deployed on mainstream commercial graphics engines without the need for conversion, making it applicable to a wide range of applications such as virtual reality and augmented reality.
- We conduct extensive experiments on both synthetic and real-world data, which demonstrate that our method outperforms or achieves comparable results to the state-of-the-art methods while being significantly more efficient.

2 Related Work

Shape Reconstruction. Reconstructing object geometry is a long-standing problem in computer vision and graphics. In traditional methods, the Structure-from-Motion (SfM) method [Schonberger and Frahm, 2016] is first applied to generate sparse feature points and find their correspondence to further generate a sparse point cloud and a rough mesh. Then, the Multi-View-Stereo (MVS) method [Schönberger *et al.*, 2016] is leveraged to generate dense pixel-level matching. Finally, a dense mesh with vertex color is generated by the Poisson reconstruction method [Kazhdan and Hoppe, 2013]. A few recent learning-based methods assume that the target object mesh is homeomorphic with the sphere. The method in [Wang *et al.*, 2018] uses an image feature network (2D CNN) to extract perceptual features from the input image, and a cascaded mesh deformation network (GCN) to progressively deform an ellipsoid mesh into the desired 3D model. These methods can only reconstruct rough geometry, which is insufficient for downstream tasks such as VR and AR.

Reflectance Reconstruction. Reflectance model, *i.e.*, spatially-varying bidirectional reflectance distribution function (SVBRDF), describes how light interacts with surfaces in the scene. Traditional SVBRDF reconstruction methods [Matusik, 2003; Lensch *et al.*, 2003; Holroyd *et al.*, 2010; Dong *et al.*, 2010; Chen *et al.*, 2014; Dong *et al.*, 2015; Kang *et al.*, 2018] rely on dense input images measured using auxiliary equipment, *e.g.*, gantry. Some other works focus on [Zhou *et al.*, 2016; Kim *et al.*, 2017; Park *et al.*, 2018] exploiting the structure of SVBRDF parameter spaces to reduce the requirement for the number of input images. Additionally, some data-driven works [Li *et al.*, 2018b; Aittala *et al.*, 2016; Gao *et al.*, 2019] have been introduced recently to produce plausible SVBRDF estimations for near-flat objects using a small number of input images. Despite their ease of use, these methods struggle to handle more complex objects.

Differentiable Rendering. Differentiable methdos are reviewed in detail in [Kato *et al.*, 2020]. Here we focus on methods closely related to our work. Traditional rendering pipelines, *e.g.*, rasterization and ray-tracing, are not differentiable due to some discrete parts, which means that they cannot work with the gradient descent method just like neural networks. The emergence of differentiable rendering has changed all. Some approximately-based methods have been proposed recently. [Loper and Black, 2014; Kato *et al.*, 2018; Kato and Harada, 2019] compute approximated gradients to optimize scene parameters. Besides, [Liu *et al.*, 2019; Chen *et al.*, 2019] replace the z-buffer-based triangle selection of a vanilla rasterization process with a probabilistic manner. Unfortunately, the result is not so good due to inaccuracies introduced by these methods. On the contrary, Monte Carlo edge sampling based methods [Li *et al.*, 2018a; Zhang *et al.*, 2019] provide unbiased gradient estimates capable of producing plausible results. But these methods are resource-consuming because of their path tracing module, hindering their generalization.

Differentiable Rendering based Multi-view Inverse Rendering. The emergence of differentiable rendering boosts the development of the inverse rendering techniques. Several prior works leverage differentiable rendering methods to solve the inverse problem. Wu et al. [2020] propose a completely unsupervised method for face reconstruction from just one face image using the approximately-based differentiable method, which does not rely on existing 3D morphable face models. Luan et al. [2021] leverage the Monte Carlo edge sampling based methods to reconstruct fine geometry and plausible material. However, they assume that the camera and light are collocated. In addition, a rough geometry scanned by professional equipment is also needed. Although these settings could narrow down the solution space, they also reduce the generalization of the method. The method proposed by Li et

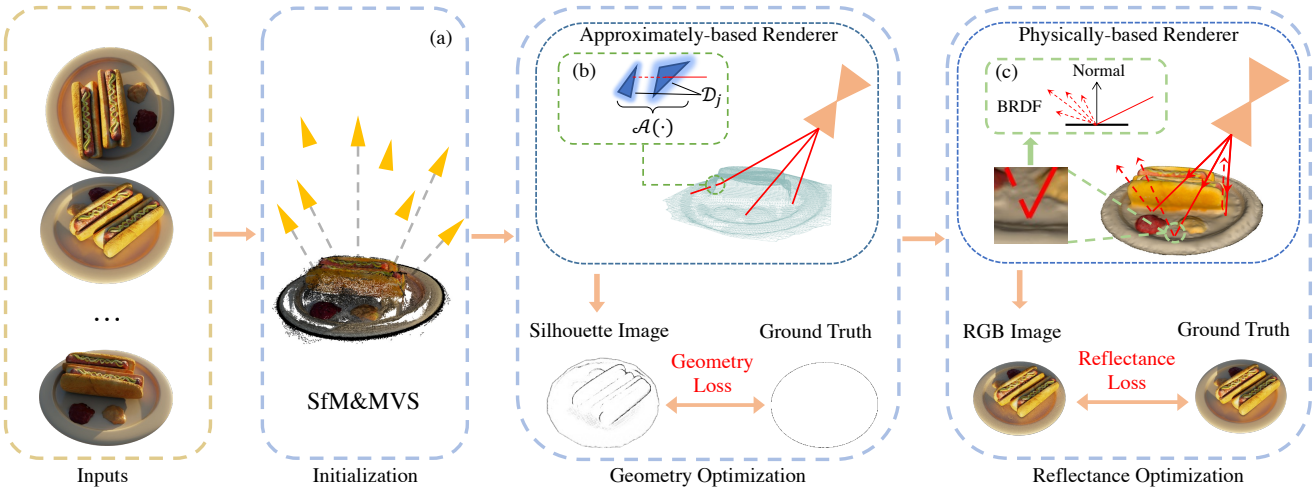

Figure 2: Overview of our inverse rendering pipeline. Our method takes as input a set of RGB images of some object obtained by conventional hand-held cameras from several viewpoints with uncontrolled lighting conditions. Then, we utilize SfM and MVS method to reconstruct a virtual scene roughly matching the real scene in the initialization phase. In the optimization phase, we first optimize the geometry using the approximately-based differentiable rendering method, then optimize reflection using the physically-based differentiable rendering method. Both methods work iteratively by pushing the loss between rendering images and the ground truth in the camera pose.

al. [2021] reduces the requirements for input. It takes as input multi-view wild scene images, and reconstructs the initial geometry through MVS, then use the general Monte Carlo path tracing differentiable renderer to optimize the material, illumination, and geometry. Although it can achieve good results, it is resource-consuming and time-consuming.

3 Our Method

3.1 Overview

Our inverse rendering pipeline, shown in Figure 2, takes as input a set of RGB images captured by conventional hand-held cameras from multiple viewpoints. Using traditional methods (SfM and MVS), our method first reconstructs a virtual scene (in the form of a triangle mesh with vertex color) that roughly matches the real scene in the initialization phase. Afterward, in the optimization phase, our method first uses an approximate differentiable rendering method to optimize the geometry, then uses a physically-based differentiable rendering method to optimize the reflectance. Both methods iteratively improve the image loss between rendered images and the ground truth for the same viewpoint. Finally, the optimized scene parameters can be used for a variety of applications such as novel view synthesis and relighting. Conceptually, our pipeline can be formulated as an analysis-by-synthesis problem

$$\Theta^* = \arg\min_{\Theta} \mathcal{L}(\mathcal{I}(\Theta), \Theta; \tilde{\mathcal{I}}),$$

where Θ represents the scene parameters that need to be estimated, including geometry and reflection parameters; $\mathcal{L}$ is a loss function to be minimized; $\tilde{\mathcal{I}}$ is the ground truth image, and $\mathcal{I}(\Theta)$ is the image rendered by our hybrid differentiable rendering method using the same camera parameters as $\tilde{\mathcal{I}}$. In the following, we present the details of each phase.

3.2 Initialization Phase

Our reconstruction pipeline is designed to work in complex, unstructured scenes where the input images have no additional information, such as depth. To tackle these challenges, we leverage traditional methods to obtain an initial geometry. Specifically, as shown in Figure 2(a), we use the SfM method from [Schonberger and Frahm, 2016] to generate sparse feature points in each image. These points are then used to reconstruct a sparse point cloud and a rough mesh by matching the feature points across images. Next, we use the MVS reconstruction method from [Schönberger *et al.*, 2016] to generate dense pixel-level matching. Finally, a dense mesh is then obtained from Poisson surface reconstruction [Kazhdan *et al.*, 2006], with vertex colors derived from the input images.

In terms of lighting, we use an environment map as our lighting source. The resolution of the environment map is 512×128. The environment map is a learnable parameter that is inferred during the material optimization phase. In the beginning, we assume that the light is white and the three channels of our environment map are set to $(0.5, 0.5, 0.5)$.

3.3 Optimization Phase

In the optimization phase, starting from the initial geometry obtained in the previous step, we adopt a hybrid differentiable rendering method to further optimize the geometry and reflectance. Specifically, we utilize an approximate differentiable rendering method to optimize the geometry, and a physically-based differentiable rendering method to optimize the reflectance. Our hybrid approach is motivated by the following observations in experiments:

- Although the geometry reconstructed by SfM and MVS in the initialization phase provides a good approximation of the scene, there can be some defects in the boundary regions.
- Approximate differentiable rendering methods dedicated to

calculating the gradient of geometry can be very efficient, but the visual effect of their results may be of lower quality due to their simplification of the material model.

- Physically-based methods can accurately reconstruct complex materials, but their high computational costs and resource demands may impede their efficiency.

Specifically, starting from the triangle mesh obtained in the initialization phase, we further optimize its vertex positions θ_{p} which define the geometry, as well as two 2D texture maps that contain the diffuse albedo θ_{d} and specular albedo θ_{s} respectively and define the SVBRDF. In this way, our scene parameters can be represented as $\mathbf{\Theta} = (\theta_{\mathrm{p}}, \theta_{\mathrm{d}}, \theta_{\mathrm{s}})$. We first optimize the vertex positions using an approximate differentiable rendering method. Afterward, we optimize the diffuse albedo and specular albedo using a physically-based differentiable rendering method. Details of the optimization are presented below.

Geometry Optimization. To optimize the geometry, we adopt a differentiable model similar to Soft Rasterizer [Liu *et al.*, 2019] to compute a silhouette image for each input image using the same camera parameters (see the supplementary materials for details of the computation), to indicate the occupancy from the same view directions as the input images. Then we minimize the following loss function to derive the vertex positions θ_{p}:

$$\mathcal{L}_{\text{geo}} = \lambda_{\text{sil}}\mathcal{L}_{\text{sil}} + \lambda_{\text{lap}}\mathcal{L}_{\text{lap}} + \lambda_{\text{normal}}\mathcal{L}_{\text{normal}} + \lambda_{\text{edge}}\mathcal{L}_{\text{edge}}\,, \quad (1)$$

where $\mathcal{L}_{\text{sil}}$ is a silhouette loss that indicates the consistency between the computed silhouette images $\mathcal{I}_{\text{sil}}(\theta_{\mathrm{p}})$ and the ground-truth ones $\tilde{\mathcal{I}}_{\text{sil}}$ derived from the input [Liu *et al.*, 2019]:

$$\mathcal{L}_{\text{sil}} = 1 - \left\|\tilde{\mathcal{I}}_{\text{sil}} \otimes \mathcal{I}_{\text{sil}}\right\|_1 \Big/ \left\|\tilde{\mathcal{I}}_{\text{sil}} \oplus \mathcal{I}_{\text{sil}} - \tilde{\mathcal{I}}_{\text{sil}} \otimes \mathcal{I}_{\text{sil}}\right\|_1,$$

with $\otimes$ and $\oplus$ being the element-wise product and sum operators, respectively. The other terms in $\mathcal{L}_{\text{geo}}$ are regularizers. Among them, $\mathcal{L}_{\text{lap}} = \|\boldsymbol{L}\boldsymbol{V}\|^2$ is a mesh Laplacian loss $\mathcal{L}_{\text{lap}}$ of a mesh with n vertices, $\boldsymbol{V}$ is the $n \times 3$ coordinates matrix, and $\boldsymbol{L} \in \mathbb{R}^{n\times n}$ is the Laplacian matrix of the mesh (See [Nealen *et al.*, 2006] for details). $\mathcal{L}_{\text{normal}} = \sum_{i,j}\left[1 - (\boldsymbol{n}_i \cdot \boldsymbol{n}_j)\right]^2$ is a normal consistency loss to make the normals of adjacent faces to vary slowly, where the sum is over all triangle pairs (i, j) sharing a common edge, and $\boldsymbol{n}_i$ and $\boldsymbol{n}_j$ are the face normals of the two specific triangles. $\mathcal{L}_{\text{edge}} = \sqrt{\sum_i e_i^2}$ is an edge length loss to avoid long edges that can cause ill-shaped triangles, where e_i denotes the length of the i-th edge.

Reflectance Optimization. Our reflectance optimization is based on the rendering equation proposed in [Kajiya, 1986]. For a surface point x with surface normal n, let $L_i(\omega_i; x)$ be the incident light intensity at location x along the direction ω_i, and SVBRDF $f_r(\omega_o, \omega_i; x)$ be the reflectance coefficient of the material at location x for incident light direction ω_i and viewing direction ω_o. Then the observed light intensity $L_o(\omega_o; x)$ is an integral over the upper hemisphere Ω:

$$L_o(\omega_o; x) = \int_{\Omega} L_i(\omega_i) f_r(\omega_o, \omega_i; x)(\omega_i \cdot n) d_{\omega_i}.$$

We leverage a modified Cook-Torrance (CT) model [Cook and Torrance, 1982] based on [Zeltner *et al.*, 2021] as our reflectance model, which contains a rough surface with diffuse and specular reflection without refraction. Mathematically, our reflectance model can be described as the following,

$$f_r\left(\omega_o, \omega_i; x\right) = \theta_d(x) + \theta_s\left(\omega_o, \omega_i; x\right),$$

where θ_d and θ_s are the diffuse and specular reflectance:

$$\theta_d(x) = \rho_d(x)/\pi,$$
$$\theta_s\left(\omega_o, \omega_i; x\right) = \rho_s(x)\frac{D(h,\alpha)G\left(\omega_o, \omega_i, n(x)\right)}{\left(n(x)\cdot\omega_i\right)\left(n(x)\cdot\omega_o\right)\pi}.$$

Specifically, ρ_d is diffuse albedos, ρ_s is specular albedos, and h is the halfway vector. $D(h, \alpha)$ is the microfacet distribution function which is GGX [Walter *et al.*, 2007] used in our work. α and $n(x)$ denote the surface's roughness and normal, respectively. G is a shadowing-masking function. Like other works, we also ignore the Fresnel effect which cannot be observed in our scene. To optimize the reflectance parameters $\theta_{\mathrm{r}} = (\theta_{\mathrm{d}}, \theta_{\mathrm{s}})$, we minimize the following reflectance loss:

$$\mathcal{L}_{\text{ref}} = \lambda_{\text{rgb}}\mathcal{L}_{\text{rgb}}(\mathcal{I}_{\text{rgb}}(\theta_{\mathrm{r}}); \tilde{\mathcal{I}}_{\text{rgb}}) + \lambda_{\text{reg}}\mathcal{R}(\theta_{\mathrm{r}}), \quad (2)$$

where $\mathcal{L}_{\text{rgb}} = \left\|\tilde{\mathcal{I}}_{\text{rgb}} - \mathcal{I}_{\text{rgb}}(\theta_{\mathrm{r}})\right\|_1$ measures the ℓ_1 norm of the difference between the rendered color image $\mathcal{I}_{\text{rgb}}(\theta_{\mathrm{r}})$ and the ground truth $\tilde{\mathcal{I}}_{\text{rgb}}$. $\mathcal{R}(\theta_{\mathrm{r}})$ is a regularizer for the diffuse albedo θ_{d} and specular albedo θ_{s}. Similar to [Schmitt *et al.*, 2020], we assume that nearby pixels with similar diffuse albedos have similar specular albedos. So we choose

$$\mathcal{R}(\theta_{\mathrm{r}}) = \sum\nolimits_{\boldsymbol{p}} \left\|\theta_{\mathrm{s}}[\boldsymbol{p}] - \left(\sum\nolimits_{\boldsymbol{q}} \theta_{\mathrm{s}}[\boldsymbol{q}] k_{\boldsymbol{p},\boldsymbol{q}}\right) / \left(\sum\nolimits_{\boldsymbol{q}} k_{\boldsymbol{p},\boldsymbol{q}}\right)\right\|_1,$$

where $k_{\boldsymbol{p},\boldsymbol{q}} = \exp\left(-\frac{\|\boldsymbol{p}-\boldsymbol{q}\|_2^2}{2\sigma_1^2} - \frac{(\theta_{\mathrm{d}}[\boldsymbol{p}]-\theta_{\mathrm{d}}[\boldsymbol{q}])^2}{2\sigma_2^2}\right)$ and $\boldsymbol{p}, \boldsymbol{q}$ are two specific mesh vertices.

4 Experiments

We perform quantitative and qualitative evaluation of our method using extensive experiments on both synthetic and real data, and compare it with state-of-the-art methods.

Implementation Details. We implement the geometry reconstruction module in the initialization phase using COLMAP[1] [Schönberger *et al.*, 2016; Schonberger and Frahm, 2016], a general-purpose Structure-from-Motion and Multi-View Stereo pipeline. In the optimization phase, we implement the approximate differentiable rendering based on Pytorch3d [Ravi *et al.*, 2020], and the physically-based differentiable rendering based on Mitsuba 2 [Nimier-David *et al.*, 2019]. The rest of our optimization pipeline is implemented with PyTorch using the Adam optimizer. For geometry optimization, we use the weights $(\lambda_{\text{sil}}, \lambda_{\text{lap}}, \lambda_{\text{edge}}, \lambda_{\text{normal}}) = (1.0, 1.0, 1.0, 0.01)$ for Eq. 1, and 0.001 for the learning rate. The ground-truth silhouette images required for Eq. 1 are obtained either from rendering (for synthetic data) or using a background removal tool[2] (for real data). For reflectance optimization, we use the weights $(\lambda_{\text{rgb}}, \lambda_{\text{reg}}) = (0.1, 1.0)$ for

[1] https://github.com/colmap/colmap

[2] https://www.remove.bg/

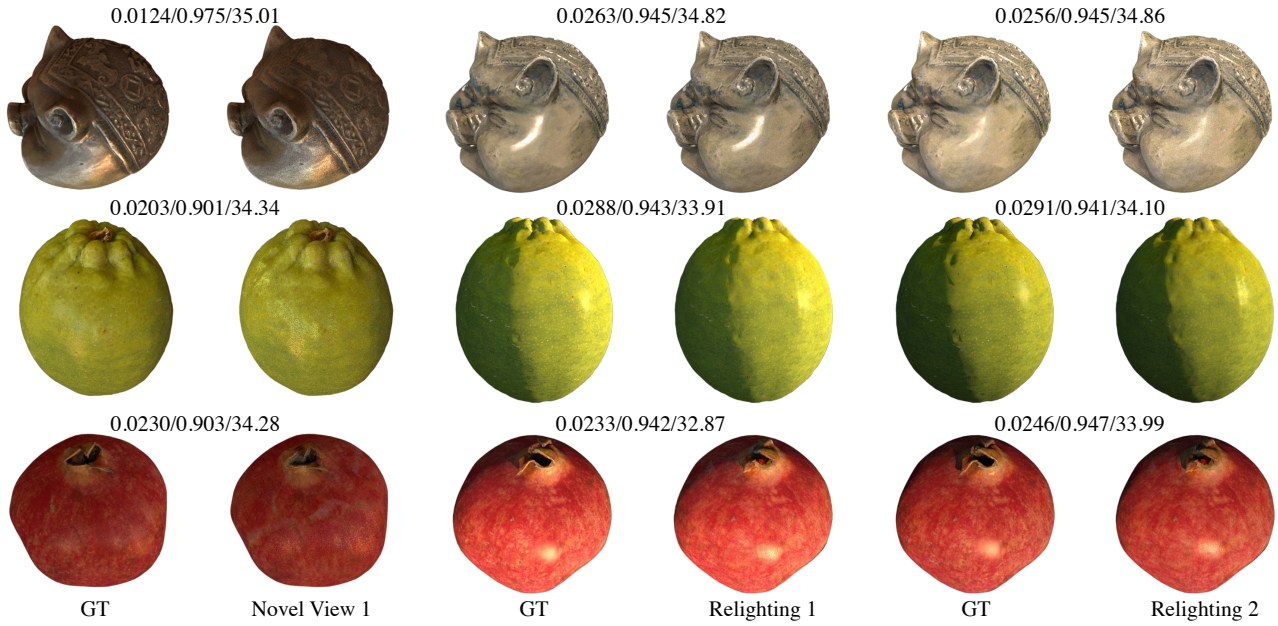

Figure 3: Results of our method on the synthetic data. We compare our predicted novel views and relighting results to the ground truth images. The number above each result indicates LPIPS, SSIM, and PSNR metrics calculated in 512×512 size pictures, respectively. **Please zoom in to see the details, especially the challenging specular highlights on the surface of the objects.**

Eq. 2, and 0.0001 for the learning rate. To avoid excessive memory consumption in the physically-based rendering module, we set the maximum depth of ray tracing bounce to 3, the downsampling factor of raw images to 4, and the number of sampling per pixel *spp* to 4 in the iterative optimization phase.

We train and test our method on one NVIDIA RTX 3090 GPU. For geometry optimization, our implementation uses 200 iterations per view and takes about 0.5 hours per example. For reflectance optimization phase, our implementation uses 400 iterations per view and takes $1.5\sim2.5$ hours per example.

Synthetic Data. Our synthetic data is created using meshes and textures from [Zhou *et al.*, 2016] and the Internet, covering different materials and geometries. Concretely, for each object, we render 400 images with colored environmental lighting using graphics engines, 300 for training, and the left 100 for novel view synthesis testing, whose viewpoints are evenly distributed inside the upper hemisphere. In addition, we also render the same object with two other environment lighting maps as the ground truth for the following relighting performance testing. To quantitatively evaluate our method, we use three image quality metrics—LPIPS [Zhang *et al.*, 2018], SSIM and PSNR—to compare the rendering results with the corresponding ground truth images. Figure 3 shows examples of results from our method and their ground truth, as well as their evaluation metric values. Both the quantitative metrics and the qualitative visualization show that our novel views and relighting results match the ground truth closely. Fig. 6 shows the diffuse and specular albedo for the lemon model.

Real Data. We evaluate our method on multiple real-world images from the DTU datasets [Aanæs *et al.*, 2016], where the objects are glossy and the illumination is static across different views. We use two objects from the dataset, *shiny scan114 buddha* and *scan110 ghost*, and discard photos with strong shadows. Our inverse rendering results are shown in Figure 4. We can see that our pipeline generates photo-realistic novel views, and results of relighting and material editing.

Comparison with State-of-the-art Methods. As far as we are aware, there is no prior work tackling exactly the same problem as our work: reconstructing triangle meshes and PBR materials from multi-view real-world object images with uncontrolled lighting conditions. The methods closest to our method are [Luan *et al.*, 2021; Li *et al.*, 2021], but no source code or data is released. It is worth noting that our method addresses the shortcomings of [Li *et al.*, 2021] in dealing with geometry, while overcoming the limitation of [Luan *et al.*, 2021] which requires the camera and point lighting to be in the same position and the images to be taken in dark scenes.

We compare our method to the most related neural rendering approaches, including PhySG [Zhang *et al.*, 2021b], NeRF [Mildenhall *et al.*, 2020] and IDR [Yariv *et al.*, 2020], in terms of novel view synthesis. These approaches are different from our method in the model of light transport: NeRF uses the occupancy function to describe geometry and appearance maps and gets the pixel color according to location and viewing direction in a ray-marching way, while PhySG and IDR use SDF to represent geometry and material. In addition, unlike the aforementioned two methods that focus solely on novel view synthesis, PhySG also performs inverse rendering tasks, using Spherical Gaussian functions to describe materials and illumination. Qualitative and quantitative comparisons between different methods on our synthetic data are depicted in Figure 5 and Table 1. The unsatisfactory performance of PhySG may be due to the presence of strong specular highlights in our

Figure 4: With our pipeline, we can synthesize novel views and edit the materials and lighting of the real-world captures.

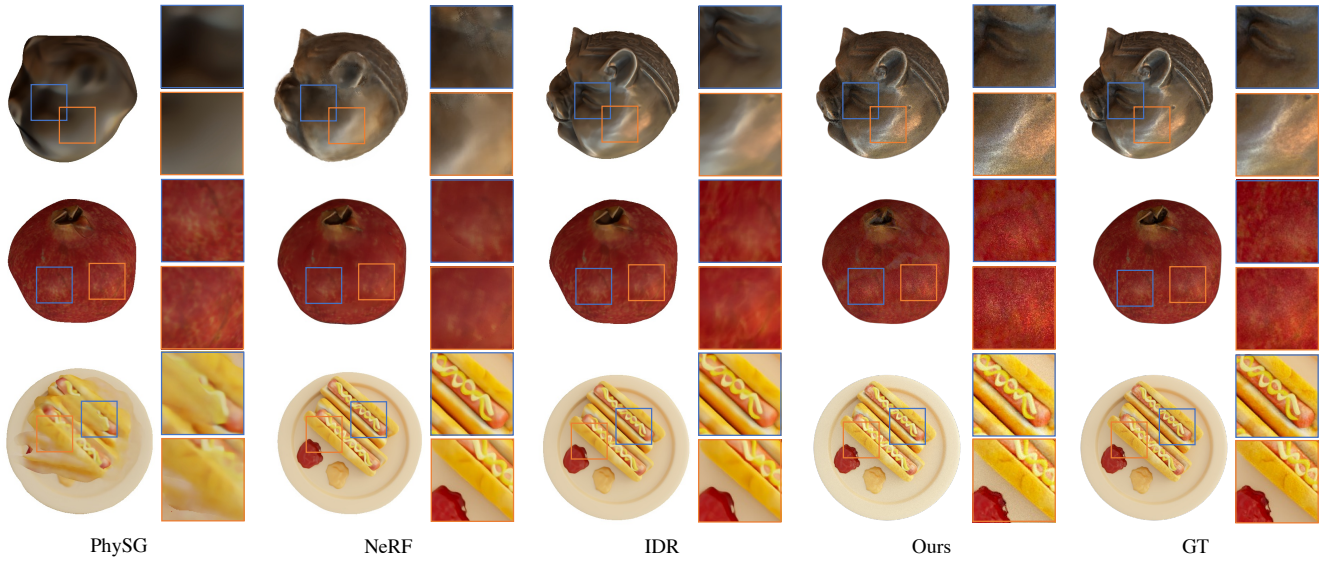

Figure 5: We qualitatively compare our results with PhySG [Zhang *et al.*, 2021b], NeRF [Mildenhall *et al.*, 2020] and IDR [Yariv *et al.*, 2020].

Figure 6: Our diffuse and specular albedo. We have adjusted the tonal range to make the variation of specular albedo more visible.

synthetic data, which are difficult to describe using the Spherical Gaussian functions employed by PhySG. NeRF performs relatively poorly in view synthesis because its volumetric representation does not concentrate colors around surfaces as well as surface-based approaches. IDR does a better job in view synthesis because of its view-dependence model. However, it still struggles to synthesize specular highlights due to the non-physical model of appearance. In contrast, our method models highlight well, benefiting from our PBR materials. We also compare the running time, where our training time is the sum of initialization time and optimization time. Thanks to the combination of efficiency from the approximate method and the accuracy from the physically-based method, our hybrid approach runs significantly faster than the two baselines both in training and testing. In addition, our optimized geometry and material can be deployed directly on mainstream graphics engines.

We also compare our method with the latest related work IRON [Zhang *et al.*, 2022] which adopts neural representations and also leverages a hybrid optimization scheme. Note that IRON assumes a point light source, while our method allows for uncontrolled lighting conditions. As a result, IRON performs poorly in complex lighting situations. Figure 9 com-

	Synthetic Pig (512×512) #train=300, #test=100					Synthetic Pomegranate (512×512) #train=300, #test=100					Synthetic Hotdog (512×512) #train=300, #test=100				
	LPIPS↓	SSIM↑	PSNR↑	Training Time (h)↓	Testing Time (s)↓	LPIPS↓	SSIM↑	PSNR↑	Training Time (h)↓	Testing Time (s)↓	LPIPS↓	SSIM↑	PSNR↑	Training Time (h)↓	Testing Time (s)↓
PhySG	0.2200	0.784	17.39	∼ 15	∼ 4	0.0425	0.928	23.98	∼ 15	∼ 4	0.1473	0.8319	22.30	∼ 15	∼ 4
NeRF	0.0878	0.859	27.19	∼ 15	∼ 10	0.0600	0.918	31.91	∼ 15	∼ 10	0.0453	0.921	27.90	∼ 15	∼ 10
IDR	0.0183	0.955	29.94	∼ 18	∼ 5	**0.0211**	**0.942**	26.85	∼ 18	∼ 5	0.0303	0.945	30.97	∼ 18	∼ 5
Ours	**0.0124**	**0.975**	**35.01**	∼ **3**	∼ **1**	0.0230	0.903	**34.28**	∼ **2.5**	∼ **1**	**0.0088**	**0.951**	**33.89**	∼ **3**	∼ **1**

Table 1: Comparison between PhySG [Zhang *et al.*, 2021b], NeRF [Mildenhall *et al.*, 2020], IDR [Yariv *et al.*, 2020] and our method. #train and #test denote the size of training set and test set, respectively. Training time is in hours and rendering time is in seconds.

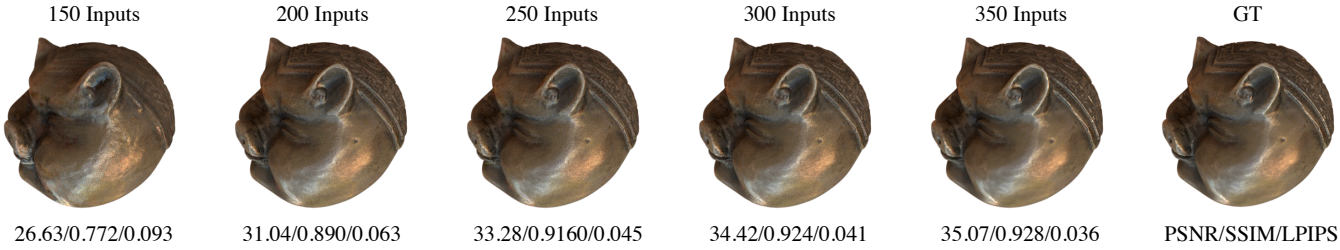

Figure 7: Ablation study on the number of input images. We show a novel view synthesis result of the reconstruction for each object, given a different number of input images, whose viewpoints are evenly distributed inside the upper hemisphere. **We recommend that readers zoom in to see the details of the picture, especially the folds at the top and the specular highlights on the face.**

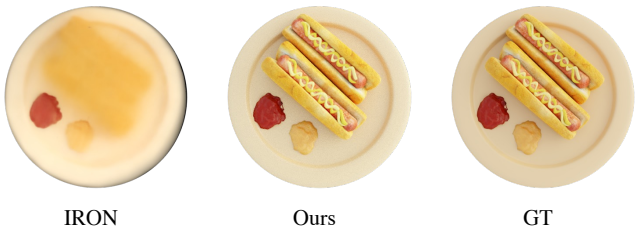

Figure 8: Comparison between our method and IRON [Zhang *et al.*, 2022] on the hotdog data.

pares the two methods in different lighting conditions. Our method can achieve similarly good results as IRON on point-lighting data, while being superior with more complex lighting. In particular, IRON cannot handle the highlights on the surface of objects under environment lighting because their light source cameras are in the same position, while our method can handle it well. In addition, for some flat data like the hotdog in Figure 8, IRON struggles to obtain appropriate geometry, while our method can produce an accurate reconstruction.

Ablation Study. Figure 7 shows our ablation study on the number of input images, which depicts how the number of input images affects our reconstruction quality. With too few input images, the optimization may become highly under-constrained, making it difficult to produce accurate synthesis results. In our experiments, 150 images are sufficient to produce a quite good result. With 250 or more input images, our results will closely match the ground truth. More ablation studies can be found in the supplementary materials.

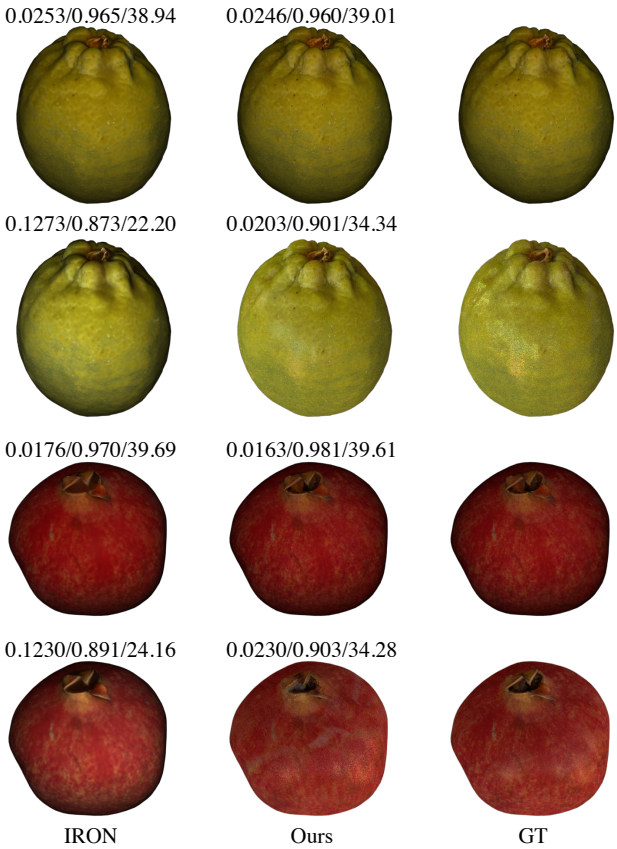

Figure 9: Comparison between our method and IRON [Zhang *et al.*, 2022]. The first and third rows are point lighting, and the second and fourth rows are environment lighting. The numbers above each result show the LPIPS, SSIM and PSNR in 512×512 pictures, respectively.

5 Conclusion

We introduce a novel efficient hybrid differentiable rendering method to reconstruct triangular object mesh and PBR material from multi-view real-world images with uncontrolled lighting conditions. Unlike prior works that require massive resource consumption or approximated rendering process, we utilize an approximate method to optimize the geometry and a physically-based method to estimate the reflectance so that we could benefit from both the efficiency of the former and the high quality of the latter. In general, our method can handle a wide range of real-world scenes, providing an attractive and efficient solution and enabling photo-realistic novel view synthesis and relighting applications.

Limitations and future work. Our method can have difficulties with very thin geometry, which is a common problem of mesh-based methods. In addition, our method optimizes geometry and material separately. A potential future work is a unified pipeline to optimize geometry and material simultaneously, which should further improve the result quality.

Acknowledgments

This work was supported by the NSFC under Grant 62072271.

References

[Aanæs *et al.*, 2016] Henrik Aanæs, Rasmus Ramsbøl Jensen, George Vogiatzis, and et al. Large-scale data for multiple-view stereopsis. *International Journal of Computer Vision*, 120(2):153–168, 2016.

[Aittala *et al.*, 2016] Miika Aittala, Timo Aila, and Jaakko Lehtinen. Reflectance modeling by neural texture synthesis. *ACM Trans. Graph. (TOG)*, 35(4):1–13, 2016.

[Boss *et al.*, 2021] Mark Boss, Raphael Braun, Varun Jampani, and et al. Nerd: Neural reflectance decomposition from image collections. In *Proceedings of the IEEE/CVF ICCV*, pages 12684–12694, 2021.

[Chen *et al.*, 2014] Guojun Chen, Yue Dong, Pieter Peers, and et al. Reflectance scanning: Estimating shading frame and brdf with generalized linear light sources. *ACM Trans. Graph. (TOG)*, 33(4):1–11, 2014.

[Chen *et al.*, 2019] Wenzheng Chen, Huan Ling, Jun Gao, and et al. Learning to predict 3d objects with an interpolation-based differentiable renderer. *Advances in NeurIPS*, 32, 2019.

[Cook and Torrance, 1982] Robert L Cook and Kenneth E. Torrance. A reflectance model for computer graphics. *ACM Trans. Graph. (TOG)*, 1(1):7–24, 1982.

[Dong *et al.*, 2010] Yue Dong, Jiaping Wang, Xin Tong, and et al. Manifold bootstrapping for svbrdf capture. *ACM Trans. Graph. (TOG)*, 29(4):1–10, 2010.

[Dong *et al.*, 2015] Zhao Dong, Bruce Walter, Steve Marschner, and et al. Predicting appearance from measured microgeometry of metal surfaces. *ACM Trans. Graph. (TOG)*, 35(1):1–13, 2015.

[Gao *et al.*, 2019] Duan Gao, Xiao Li, Yue Dong, and et al. Deep inverse rendering for high-resolution svbrdf estimation from an arbitrary number of images. *ACM Trans. Graph. (TOG)*, 38(4):134–1, 2019.

[Holroyd *et al.*, 2010] Michael Holroyd, Jason Lawrence, and Todd Zickler. A coaxial optical scanner for synchronous acquisition of 3d geometry and surface reflectance. *ACM Trans. Graph. (TOG)*, 29(4):1–12, 2010.

[Kajiya, 1986] James T Kajiya. The rendering equation. In *Proceedings of the 13th ACM SIGGRAPH*, pages 143–150, 1986.

[Kang *et al.*, 2018] Kaizhang Kang, Zimin Chen, Jiaping Wang, and et al. Efficient reflectance capture using an autoencoder. *ACM Trans. Graph. (TOG)*, 37(4):127–1, 2018.

[Kato and Harada, 2019] Hiroharu Kato and Tatsuya Harada. Learning view priors for single-view 3d reconstruction. In *Proceedings of the IEEE/CVF CVPR*, pages 9778–9787, 2019.

[Kato *et al.*, 2018] Hiroharu Kato, Yoshitaka Ushiku, and Tatsuya Harada. Neural 3d mesh renderer. In *Proceedings of the IEEE/CVF CVPR*, pages 3907–3916, 2018.

[Kato *et al.*, 2020] Hiroharu Kato, Deniz Beker, Mihai Morariu, and et al. Differentiable rendering: A survey. *arXiv preprint arXiv:2006.12057*, 2020.

[Kazhdan and Hoppe, 2013] Michael Kazhdan and Hugues Hoppe. Screened poisson surface reconstruction. *ACM Trans. Graph. (TOG)*, 32(3):1–13, 2013.

[Kazhdan *et al.*, 2006] Michael Kazhdan, Matthew Bolitho, and Hugues Hoppe. Poisson surface reconstruction. In *Proceedings of the fourth Eurographics SGP*, volume 7, 2006.

[Kim *et al.*, 2017] Kihwan Kim, Jinwei Gu, Stephen Tyree, and et al. A lightweight approach for on-the-fly reflectance estimation. In *Proceedings of the IEEE ICCV*, pages 20–28, 2017.

[Lensch *et al.*, 2003] Hendrik PA Lensch, Jan Kautz, Michael Goesele, and et al. Image-based reconstruction of spatial appearance and geometric detail. *ACM Trans. Graph. (TOG)*, 22(2):234–257, 2003.

[Li *et al.*, 2018a] Tzu-Mao Li, Miika Aittala, Frédo Durand, and et al. Differentiable monte carlo ray tracing through edge sampling. *ACM Trans. Graph. (TOG)*, 37(6):1–11, 2018.

[Li *et al.*, 2018b] Zhengqin Li, Kalyan Sunkavalli, and Manmohan Chandraker. Materials for masses: Svbrdf acquisition with a single mobile phone image. In *Proceedings of the ECCV*, pages 72–87, 2018.

[Li *et al.*, 2021] Rui Li, Guangmin Zang, Miao Qi, and et al. Shape and reflectance reconstruction in uncontrolled environments by differentiable rendering. *arXiv preprint arXiv:2110.12975*, 2021.

[Liu *et al.*, 2019] Shichen Liu, Tianye Li, Weikai Chen, and et al. Soft rasterizer: A differentiable renderer for image-based 3d reasoning. In *Proceedings of the IEEE/CVF ICCV*, pages 7708–7717, 2019.

[Loper and Black, 2014] Matthew M Loper and Michael J Black. Opendr: An approximate differentiable renderer. In *Proceedings of the ECCV*, pages 154–169. Springer, 2014.

[Lorensen and Cline, 1987] William E Lorensen and Harvey E Cline. Marching cubes: A high resolution 3d surface construction algorithm. *Proceedings of the 14th ACM SIGGRAPH*, 21(4):163–169, 1987.

[Luan *et al.*, 2021] Fujun Luan, Shuang Zhao, Kavita Bala, and et al. Unified shape and svbrdf recovery using differentiable monte carlo rendering. In *Computer Graphics Forum*, volume 40, pages 101–113, 2021.

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

[Mildenhall *et al.*, 2020] Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, and et al. Nerf: Representing scenes as neural radiance fields for view synthesis. In *Proceedings of the ECCV*, 2020.

[Munkberg *et al.*, 2021] Jacob Munkberg, Jon Hasselgren, Tianchang Shen, and et al. Extracting triangular 3d models, materials, and lighting from images. *arXiv preprint arXiv:2111.12503*, 2021.

[Nealen *et al.*, 2006] Andrew Nealen, Takeo Igarashi, Olga Sorkine, and et al. Laplacian mesh optimization. In *Proceedings of the 4th international conference on Computer graphics and interactive techniques in Australasia and Southeast Asia*, pages 381–389, 2006.

[Nimier-David *et al.*, 2019] Merlin Nimier-David, Delio Vicini, Tizian Zeltner, and et al. Mitsuba 2: A retargetable forward and inverse renderer. *ACM Trans. Graph. (TOG)*, 38(6):1–17, 2019.

[Park *et al.*, 2018] Jeong Joon Park, Richard Newcombe, and Steve Seitz. Surface light field fusion. In *2018 International Conference on 3D Vision (3DV)*, pages 12–21. IEEE, 2018.

[Patow and Pueyo, 2003] Gustavo Patow and Xavier Pueyo. A survey of inverse rendering problems. In *Computer Graphics Forum*, volume 22, pages 663–687, 2003.

[Ravi *et al.*, 2020] Nikhila Ravi, Jeremy Reizenstein, David Novotny, and et al. Accelerating 3d deep learning with pytorch3d. *arXiv preprint arXiv:2007.08501*, 2020.

[Remelli *et al.*, 2020] Edoardo Remelli, Artem Lukoianov, Stephan Richter, and et al. Meshsdf: Differentiable iso-surface extraction. *Advances in Neural Information Processing Systems*, 33:22468–22478, 2020.

[Schmitt *et al.*, 2020] Carolin Schmitt, Simon Donne, Gernot Riegler, and et al. On joint estimation of pose, geometry and svbrdf from a handheld scanner. In *Proceedings of the IEEE/CVF CVPR*, pages 3493–3503, 2020.

[Schonberger and Frahm, 2016] Johannes L Schonberger and Jan-Michael Frahm. Structure-from-motion revisited. In *Proceedings of the IEEE CVPR*, pages 4104–4113, 2016.

[Schönberger *et al.*, 2016] Johannes L Schönberger, Enliang Zheng, Jan-Michael Frahm, and et al. Pixelwise view selection for unstructured multi-view stereo. In *Proceedings of the ECCV*, pages 501–518. Springer, 2016.

[Srinivasan *et al.*, 2021] Pratul P Srinivasan, Boyang Deng, Xiuming Zhang, and et al. Nerv: Neural reflectance and visibility fields for relighting and view synthesis. In *Proceedings of the IEEE/CVF CVPR*, pages 7495–7504, 2021.

[Walter *et al.*, 2007] Bruce Walter, Stephen R Marschner, Hongsong Li, and et al. Microfacet models for refraction through rough surfaces. *Rendering techniques*, 2007:18th, 2007.

[Wang *et al.*, 2018] Nanyang Wang, Yinda Zhang, Zhuwen Li, and et al. Pixel2mesh: Generating 3d mesh models from single rgb images. In *Proceedings of the ECCV*, pages 52–67, 2018.

[Wu *et al.*, 2020] Shangzhe Wu, Christian Rupprecht, and Andrea Vedaldi. Unsupervised learning of probably symmetric deformable 3d objects from images in the wild. In *Proceedings of the IEEE/CVF CVPR*, pages 1–10, 2020.

[Yariv *et al.*, 2020] Lior Yariv, Yoni Kasten, Dror Moran, and et al. Multiview neural surface reconstruction by disentangling geometry and appearance. *Advances in NeurIPS*, 33:2492–2502, 2020.

[Zeltner *et al.*, 2021] Tizian Zeltner, Sébastien Speierer, Iliyan Georgiev, and et al. Monte carlo estimators for differential light transport. *ACM Trans. Graph. (TOG)*, 40(4), August 2021.

[Zhang *et al.*, 2018] Richard Zhang, Phillip Isola, Alexei A Efros, and et al. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE/CVF CVPR*, pages 586–595, 2018.

[Zhang *et al.*, 2019] Cheng Zhang, Lifan Wu, Changxi Zheng, and et al. A differential theory of radiative transfer. *ACM Trans. Graph. (TOG)*, 38(6):1–16, 2019.

[Zhang *et al.*, 2021a] Jason Zhang, Gengshan Yang, Shubham Tulsiani, and et al. Ners: Neural reflectance surfaces for sparse-view 3d reconstruction in the wild. *Advances in NeurIPS*, 34:29835–29847, 2021.

[Zhang *et al.*, 2021b] Kai Zhang, Fujun Luan, Qianqian Wang, and et al. Physg: Inverse rendering with spherical gaussians for physics-based material editing and relighting. In *Proceedings of the IEEE/CVF CVPR*, pages 5453–5462, 2021.

[Zhang *et al.*, 2021c] Xiuming Zhang, Pratul P Srinivasan, Boyang Deng, and et al. Nerfactor: Neural factorization of shape and reflectance under an unknown illumination. *ACM Trans. Graph. (TOG)*, 40(6):1–18, 2021.

[Zhang *et al.*, 2022] Kai Zhang, Fujun Luan, Zhengqi Li, and et al. Iron: Inverse rendering by optimizing neural sdfs and materials from photometric images. In *Proceedings of the IEEE/CVF CVPR*, pages 5565–5574, 2022.

[Zhou *et al.*, 2016] Zhiming Zhou, Guojun Chen, Yue Dong, and et al. Sparse-as-possible svbrdf acquisition. *ACM Trans. Graph. (TOG)*, 35(6):1–12, 2016.