Unified Shape and SVBRDF Recovery using Differentiable Monte Carlo Rendering

FUJUN LUAN, Cornell University and Facebook Reality Labs
SHUANG ZHAO, University of California, Irvine
KAVITA BALA, Cornell University
ZHAO DONG, Facebook Reality Labs

Reconstructing the shape and appearance of real-world objects using measured 2D images has been a long-standing problem in computer vision. In this paper, we introduce a new analysis-by-synthesis technique capable of producing high-quality reconstructions through robust coarse-to-fine optimization and physics-based differentiable rendering.

Unlike most previous methods that handle geometry and reflectance largely separately, our method unifies the optimization of both by leveraging image gradients with respect to both object reflectance *and* geometry. To obtain physically accurate gradient estimates, we develop a new GPU-based Monte Carlo differentiable renderer leveraging recent advances in differentiable rendering theory to offer unbiased gradients while enjoying better performance than existing tools like PyTorch3D and redner. To further improve robustness, we utilize several shape and material priors as well as a coarse-to-fine optimization strategy to reconstruct geometry. We demonstrate that our technique can produce reconstructions with higher quality than previous methods such as COLMAP and Kinect Fusion.

CCS Concepts: • **Computing methodologies** → **Rendering**.

Additional Key Words and Phrases: global illumination, photorealistic rendering, Langevin Monte Carlo

ACM Reference Format:
Fujun Luan, Shuang Zhao, Kavita Bala, and Zhao Dong. 2021. Unified Shape and SVBRDF Recovery using Differentiable Monte Carlo Rendering. *ACM Trans. Graph.* 39, 4, Article 140 (July 2021), 10 pages. https://doi.org/10.1145/3386569.3392382

1 Introduction

Reconstructing the shape and appearance of real-world objects from 2D images has been a long-standing problem in computer vision and graphics. Previously, the acquisition of object geometry and (spatially varying) reflectance has been studied largely independently. For instance, many techniques based on multiview-stereo (MVS) [Aittala et al. 2015; Ghosh et al. 2008; Hui et al. 2017; Nam et al. 2016; Riviere et al. 2016, 2017; Schwartz et al. 2013; Tunwattanapong et al. 2013] and time-of-flight imaging [Izadi et al. 2011; Newcombe et al. 2011] have been introduced for the reconstruction of 3D shapes. Although these methods can also provide rough estimations of surface reflectance, they usually rely on the assumption of simple (e.g., diffuse-dominated) reflectance and can produce unsatisfactory results for glossy objects. On the other hand, previous approaches that specialized at recovering an object's spatially varying reflectance [Gao et al. 2019a; Guo et al. 2020; Zhou et al. 2016a] typically require object geometries to be predetermined, limiting their practical usefulness for many applications where such information is unavailable.

In this paper, we demonstrate that detailed geometry and spatially varying reflectance of a real-world object can be recovered using a unified analysis-by-synthesis framework. To this end, we apply gradient-based optimization of the rendering loss (i.e., the difference between rendered and target images) that are affected by both object geometry and reflectance. Although such gradients with respect to appearance are relatively easy to compute, the geometric gradients are known to be much more challenging to compute and, therefore, have been mostly approximated in the past using techniques like soft rasterization [Liu et al. 2019] in computer vision. We, on the other hand, leverage recent advances in physics-based differentiable rendering to obtain *unbiased* and *consistent* geometric gradients that are crucial for obtaining high-quality reconstructions.

Concretely, our contributions include:

- An efficient Monte Carlo differentiable renderer that utilizes edge sampling [Li et al. 2018a] to produce unbiased gradient estimates.
- A new analysis-by-synthesis pipeline that enables high-quality reconstruction of spatially varying reflectance and, more importantly, mesh-based object geometry.
- A coarse-to-fine scheme as well as geometry and reflectance priors for ensuring robust reconstructions.

We evaluate various key aspects of our method via several ablation studies. Further, we demonstrate the effectiveness of our technique via several synthetic and real examples.

2 Related work

Shape reconstruction. Reconstructing object geometry has been a long-standing problem in computer vision.

Multi-view Stereo (MVS) recovers the 3D geometry of sufficiently textured objects using multiple images of an object by matching feature correspondences across views and optimizing photo-consistency (e.g., [Furukawa and Ponce 2009; Schönberger et al. 2016; Seitz and Dyer 1999; Vogiatzis et al. 2005]).

Authors' addresses: Fujun Luan, Cornell University and Facebook Reality Labs, fujun@cs.cornell.edu; Shuang Zhao, University of California, Irvine, shz@ics.uci.edu; Kavita Bala, Cornell University, kb@cs.cornell.edu; Zhao Dong, Facebook Reality Labs, zhaodong@fb.com.

0730-0301/2021/7-ART140 $15.00
https://doi.org/10.1145/3386569.3392382

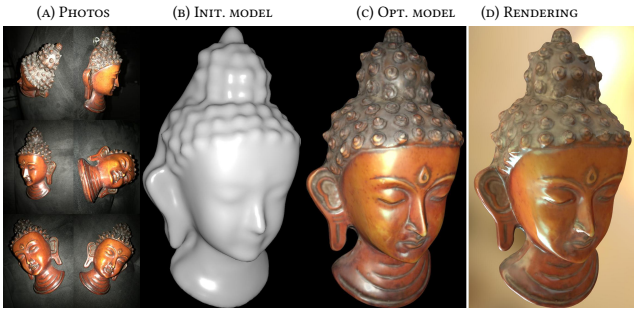

Fig. 1. **Our method** takes as input: multi-view photos (100 used and 6 shown for this example) of an object (a) and a rough initial model with its geometry obtained using standard methods (b). Then, a novel analysis-by-synthesis optimization is performed to refine the model's shape and reflectance in a unified fashion, yielding a high-quality 3D model (c). We show in (d) a re-rendering of the result under environmental lighting. *(Use Adobe Acrobat and click on (c) or (d) to see an animation of the optimization process.)*

Shape from Shading (SfS) relates surface normals to image intensities [Haefner et al. 2018; Horn 1970; Ikeuchi and Horn 1981; Maier et al. 2017; Quéau et al. 2017a,b]. Unfortunately, these methods have difficulties handling illumination changes, non-diffuse reflectance, and textureless surfaces.

Photometric Stereo (PS) takes three or more images captured with a static camera and varying illumination or object pose, and directly estimate surface normals from measurements [Holroyd et al. 2008; Papadhimitri and Favaro 2014; Quéau et al. 2015, 2016; Tunwattanapong et al. 2013; Woodham 1980; Zhou and Tan 2010]. There methods typically do not recover reflectance properties beyond diffuse albedo.

Reflectance reconstruction. Real-world objects exhibit richly diverse reflectance that can be described with spatially-varying bidirectional reflectance distribution functions (SVBRDFs).

Traditional SVBRDF acquisition techniques rely on dense input images measured using light stages or gantry (e.g, [Chen et al. 2014; Dong et al. 2010, 2015; Holroyd et al. 2010; Kang et al. 2018; Lensch et al. 2003; Matusik 2003]). To democratize the acquisition, some recent works exploit the structure (e.g., sparsity) of SVBRDF parameter spaces to allow reconstructions using fewer input images (e.g., [Dong et al. 2014; Kim et al. 2017; Park et al. 2018; Wu et al. 2015; Yu et al. 1999; Zhou et al. 2016b]). Additionally, a few recent works have been introduced to produce plausible SVBRDF estimations for flat objects using a small number of input images (e.g., [Aittala et al. 2016, 2015; Deschaintre et al. 2019; Gao et al. 2019b; Guo et al. 2020; Hui et al. 2017]). Despite their ease of use, these techniques cannot be easily generalized to handle more complex shapes.

Joint estimation of shape and reflectance. Several prior works jointly estimate object shape and reflectance. Higo et al. [Higo et al. 2009] presented a plane-sweeping method for albedo, normal and depth estimation. Xia et al. [Xia et al. 2016] optimized an apparent normal field with corresponding reflectance. Nam et al. [Nam et al. 2018] proposed a technique that alternates between material-, normal-, and geometry-optimization stages. Schmitt et al. [Schmitt et al. 2020] perform joint estimation using a hand-held sensor rig with and 12 point light sources. Bi et al. [Bi et al. 2020] use six images and optimize object geometry and reflectance in two separate stages.

All these methods either rely on MVS for geometry reconstruction or perform alternative optimization of shape and reflectance, offering little to no guarantee on qualities of the reconstruction results. We, in contrast, formulate the problem as a unified analysis-by-synthesis optimization, ensuring locally optimal results.

Differentiable rendering of meshes. We now briefly review differentiable rendering techniques closely related to our work. For a more comprehensive summary, please see the survey by Kato et al. [Kato et al. 2020].

Specialized differentiable renderers have long existed in computer graphics and vision [Azinovic et al. 2019; Che et al. 2020; Gkioulekas et al. 2016, 2013; Tsai et al. 2019]. Recently, several general-purpose ones [Li et al. 2018a; Nimier-David et al. 2019] have been developed.

A key technical challenge in differentiable rendering is to estimate gradients with respect to object geometry (e.g., positions of mesh vertices). To this end, several approximated methods (e.g., [Liu et al. 2019; Loubet et al. 2019; Ravi et al. 2020]) have been proposed. Unfortunately, inaccuracies introduced by these techniques can lead to degraded result quality. On the contrary, Monte Carlo edge sampling [Li et al. 2018a; Zhang et al. 2019], which we use for our differentiable renderer, provides unbiased gradient estimates capable of producing higher-quality reconstructions.

3 Our method

We formulate the problem of joint estimation of object geometry and reflectance as an *analysis-by-synthesis* (aka. inverse-rendering) optimization. Let $\boldsymbol{\xi}$ be some vector that depicts both the geometry and the reflectance of a real-world object. Taking as input a set of images $\tilde{\boldsymbol{I}}$ of this object, we estimate $\boldsymbol{\xi}$ by minimizing a predefined loss $\mathcal{L}$:

$$\boldsymbol{\xi}^* = \arg\min_{\boldsymbol{\xi}} \mathcal{L}(\boldsymbol{I}(\boldsymbol{\xi}), \boldsymbol{\xi}; \tilde{\boldsymbol{I}}), \tag{1}$$

where $\boldsymbol{I}(\boldsymbol{\xi})$ are a set of renderings of the object generated using the geometry and reflectance provided by $\boldsymbol{\xi}$. We further allow the loss $\mathcal{L}$ to directly depend on the parameters $\boldsymbol{\xi}$ for regularization.

Acquisition setup. Similar to recent works on reflectance capture [Aittala et al. 2016, 2015; Albert et al. 2018; Deschaintre et al. 2019; Gao et al. 2019b; Hui et al. 2017; Ren et al. 2011; Riviere et al. 2016], we utilize an acquisition setup where the object is illuminated with a point light collocated with the camera. Common examples include a smart phone's flash and camera as well as a consumer-grade RGBD sensor mounted with an LED light.

Overview of our method. Efficiently solving the optimization of Eq. (1) requires computing gradient $\mathrm{d}\mathcal{L}/\mathrm{d}\boldsymbol{\xi}$ of the loss $\mathcal{L}$ with respect to the geometry and reflectance parameters $\boldsymbol{\xi}$. According to the chain rule, we know that

$$\frac{\mathrm{d}\mathcal{L}}{\mathrm{d}\boldsymbol{\xi}} = \frac{\partial \mathcal{L}}{\partial \boldsymbol{I}} \frac{\mathrm{d}\boldsymbol{I}}{\mathrm{d}\boldsymbol{\xi}} + \frac{\partial \mathcal{L}}{\partial \boldsymbol{\xi}}, \tag{2}$$

where $\partial\mathcal{L}/\partial\boldsymbol{I}$ and $\partial\mathcal{L}/\partial\boldsymbol{\xi}$ can be computed using automatic differentiation [Paszke et al. 2017]. Further, estimating gradients $\mathrm{d}\boldsymbol{I}/\mathrm{d}\boldsymbol{\xi}$

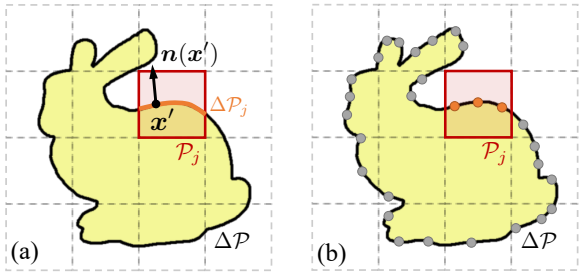

Fig. 2. **Differentiable rendering:** (a) To properly differentiate the intensity I_j of a pixel $\mathcal{P}_j$ (illustrated as red squares) with respect to object geometry, a boundary integral over $\Delta\mathcal{P}_j$ (illustrated as the orange curve) needs to be calculated. (b) We perform Monte Carlo edge sampling [Li et al. 2018a; Zhang et al. 2019] by (i) sampling points (illustrated as small discs) from pre-computed discontinuity curves $\Delta\mathcal{P}$, and (ii) accumulating their contributions in the corresponding pixels (e.g., the orange samples contribute to the pixel $\mathcal{P}_j$).

of rendered images requires performing differentiable rendering. Despite being relatively easy when the parameters $\boldsymbol{\xi}$ only capture reflectance, differentiating the rendering function $\boldsymbol{\mathcal{I}}$ becomes much more challenging when $\boldsymbol{\xi}$ also controls object geometry [Li et al. 2018a]. To this end, we develop a new differentiable renderer that is specific to our acquisition setup and provides unbiased gradient estimates.

In the rest of this section, we provide a detailed description of our technique that solves the analysis-by-synthesis optimization (1) in an efficient and robust fashion. In §3.1, we detail our forward-rendering model and explain how it can be differentiated. In §3.2, we discuss our choice of the loss $\mathcal{L}$ and optimization strategy.

3.1 Forward and differentiable rendering

In what follows, we describe (i) our representation of object geometry and reflectance; and (ii) how we render these representations in a differentiable fashion.

Object geometry and reflectance. We express object geometries using standard triangle meshes. Compared to other representations that are popular in 3D reconstruction, such as SDF volumes [Jiang et al. 2020; Park et al. 2019], occupancy networks [Mescheder et al. 2019] or sphere-based clouds [Lassner 2020], triangle meshes can be efficiently rendered and edited with many 3D digital content creation tools. Further, as a widely adopted format, triangle meshes can be easily imported into numerous applications in computer graphics, vision, and virtual/augmented reality (VR/AR).

One of the biggest challenges when using meshes for 3D reconstruction is that topological changes are difficult. We show in the following sections that this can be addressed by using reasonable initial geometries and a coarse-to-fine optimization process.

To depict an object's spatially varying reflectance, we use the Disney BRDF [Karis and Games 2013], a parametric model offering a good balance between simplicity and flexibility. This model has also been used by many prior works (e.g., [Bi et al. 2020; Li et al. 2020, 2018b]). Using this BRDF model, the spatially varying reflectance of an object is described using four 2D texture maps specifying, respectively, diffuse albedo a_d, specular albedo a_s, surface roughness α, and normal $\boldsymbol{n}$.

Forward rendering. Given a virtual object depicted using parameters $\boldsymbol{\xi}$, we render one-bounce reflection (aka. direct illumination) of the object. Specifically, assume the point light and the camera are collocated at some $\boldsymbol{o} \in \mathbb{R}^3$. Then, the intensity I_j of the j-pixel is given by an area integral over the pixel's footprint $\mathcal{P}_j$, which is typically a square on the image plane:

$$I_j = \frac{1}{|\mathcal{P}_j|} \int_{\mathcal{P}_j} \underbrace{I_\mathrm{e} \frac{f_\mathrm{r}(\boldsymbol{o} \to \boldsymbol{y} \to \boldsymbol{o})}{\|\boldsymbol{y} - \boldsymbol{o}\|^2}}_{=: I(\boldsymbol{x})} \mathrm{d}A(\boldsymbol{x}), \tag{3}$$

where $\boldsymbol{y}$ is the intersection between the object geometry $\mathcal{M}$ and a ray that originates at $\boldsymbol{o}$ and passes through $\boldsymbol{x}$ on the image plane. Further, I_e denotes the intensity of the point light; $f_\mathrm{r}(\boldsymbol{o} \to \boldsymbol{y} \to \boldsymbol{o})$ indicates the cosine-weighted BRDF at $\boldsymbol{y}$ (evaluated with both the incident and the outgoing directions pointing toward $\boldsymbol{o}$); and A is the surface-area measure. We note that no visibility check is needed in Eq. (3) since, under the collocated configuration, any point $\boldsymbol{y} \in \mathcal{M}$ visible to the camera must be also visible to the light source.

We estimate Eq. (3) using Monte Carlo integration by uniformly sampling N locations $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_N \in \mathcal{P}_j$ and computing $I_j \approx \frac{1}{N} \sum_{i=1}^{N} I(\boldsymbol{x}_i)$ where I is the integrand defined in Eq. (3).

Differentiable rendering. Computing image gradients $\mathrm{d}\boldsymbol{\mathcal{I}}/\mathrm{d}\boldsymbol{\xi}$ in Eq. (2) largely boils down to differentiating pixel intensities Eq. (3) with respect to $\boldsymbol{\xi}$. Although this can sometimes be done by differentiating the integrand I—that is, by estimating $\int_{\mathcal{P}_j} (\mathrm{d}I/\mathrm{d}\boldsymbol{\xi})\, \mathrm{d}A$—doing so is insufficient when computing gradients with respect to object geometry (e.g., vertex positions). Consequently, the gradient $\mathrm{d}\boldsymbol{\mathcal{I}}/\mathrm{d}\boldsymbol{\xi}$ has usually been approximated using soft rasterization [Liu et al. 2019; Ravi et al. 2020] or reparameterized integrals [Loubet et al. 2019]. Biased gradient estimates, unfortunately, can reduce the quality of optimization results, which we will demonstrate in §4.

On the other hand, a few general-purpose unbiased techniques [Li et al. 2018a; Zhang et al. 2019] have been introduced recently. Unfortunately, these methods focus on configurations without point light sources—which is not the case under our collocated configuration. We, therefore, derive the gradient $\mathrm{d}I_j/\mathrm{d}\boldsymbol{\xi}$ utilizing mathematical tools used by these works. Specifically, according to Reynolds transport theorem [Reynolds 1903], the gradient involves an *interior* and a *boundary* integrals:

$$\frac{\mathrm{d}I_j}{\mathrm{d}\boldsymbol{\xi}} = \frac{1}{|\mathcal{P}_j|} \Bigg[\underbrace{\int_{\mathcal{P}_j} \frac{\mathrm{d}I}{\mathrm{d}\boldsymbol{\xi}}(\boldsymbol{x})\, \mathrm{d}A(\boldsymbol{x})}_{\text{interior}} + \underbrace{\int_{\Delta\mathcal{P}_j} \left(\boldsymbol{n}(\boldsymbol{x}') \cdot \frac{\mathrm{d}\boldsymbol{x}'}{\mathrm{d}\boldsymbol{\xi}}\right) \Delta I(\boldsymbol{x}')\, \mathrm{d}\ell(\boldsymbol{x}')}_{\text{boundary}} \Bigg], \tag{4}$$

where the interior term is simply Eq. (3) with its integrand I differentiated. The boundary one, on the contrary, is over curves $\Delta\mathcal{P}_j := \Delta\mathcal{P} \cap \mathcal{P}_j$ with $\Delta\mathcal{P}$ comprised of jump discontinuity points of I. In practice, $\Delta\mathcal{P}$ consists of image-plane projections of the object's silhouettes. Further, $\boldsymbol{n}(\boldsymbol{x})$ is the curve normal within the

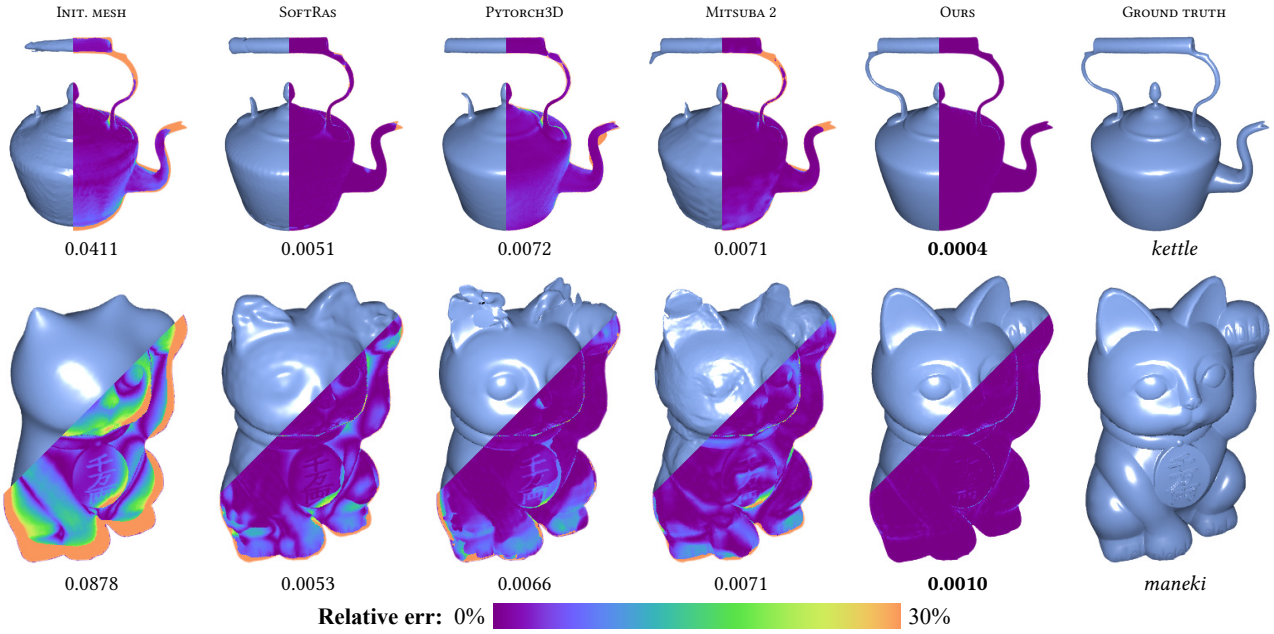

Fig. 3. **Comparison** with SoftRas [Liu et al. 2019], PyTorch3D [Ravi et al. 2020] and Mitsuba 2 [Nimier-David et al. 2019]. We render all reconstructed geometries using Phong shading and visualize depth errors (wrt. the ground-truth geometry). Initialized with the same mesh (shown in the left column), optimizations using gradients obtained with SoftRas and PyTorch3D tend to converge to low-quality results due to gradient inaccuracies caused by soft rasterization. Mitsuba 2, a ray-tracing-based system, also produces visible artifacts due to biased gradients resulting from an approximated reparameterization [Loubet et al. 2019]. When using gradients generated with our differentiable renderer, optimizations under identical configurations produce results closely resembling the targets. The number below each result indicates the average point-to-mesh distance capturing the Euclidean accuracy [Jensen et al. 2014] of the reconstructed geometry (normalized to have a unit bounding box).

image plane, $\Delta I(\boldsymbol{x})$ denotes the difference in I across discontinuity boundaries, and ℓ is the curve-length measure (see Figure 2-a).

Similar to the Monte Carlo estimation of Eq. (3), we estimate the interior integral in Eq. (4) by uniformly sampling $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N \in \mathcal{P}_j$. To handle the boundary integral, we precompute the discontinuity curves $\Delta\mathcal{P}$ (as polylines) by projecting the object's silhouette onto the image plane. At runtime, we draw $\boldsymbol{x}'_1, \ldots, \boldsymbol{x}'_M \in \Delta\mathcal{P}$ uniformly. Then,

$$\frac{\mathrm{d}I_j}{\mathrm{d}\boldsymbol{\xi}} \approx \underbrace{\frac{1}{N}\sum_{i=1}^{N} \frac{\mathrm{d}I}{\mathrm{d}\boldsymbol{\xi}}(\boldsymbol{x}_i)}_{\text{interior}} + \underbrace{\frac{1}{M}\frac{|\Delta\mathcal{P}|}{|\mathcal{P}_j|}\sum_{i=1}^{M} \mathbb{1}\left[\boldsymbol{x}'_i \in \mathcal{P}_j\right]\left(\boldsymbol{n}(\boldsymbol{x}'_i)\cdot\frac{\mathrm{d}\boldsymbol{x}'_i}{\mathrm{d}\boldsymbol{\xi}}\right)\Delta I(\boldsymbol{x}'_i)}_{\text{boundary}}, \tag{5}$$

where $|\Delta\mathcal{P}|$ denotes the total length of the discontinuity curves $\Delta\mathcal{P}$, and $\mathbb{1}[\cdot]$ is the indicator function.

In practice, we estimate gradients of pixel intensities via Eq. (5) in two rendering passes. In the first pass, we evaluate the interior component independently for each pixel. In the second pass, we evaluate the boundary component for each $\boldsymbol{x}'_i$ in parallel and accumulate the results in the corresponding pixel (see Figure 2-b).

3.2 Analysis-by-synthesis optimization

We now present our analysis-by-synthesis optimization pipeline that minimizes Eq. (1).

Object parameters. As stated in §3.1, we depict object geometry using a triangle mesh (which is comprised of per-vertex positions $\boldsymbol{p}$ and UV coordinates $\boldsymbol{u}$ as well as per-triangle vertex indices) and reflectance using three 2D texture maps specifying the object's spatially varying diffuse albedo a_d, specular albedo a_s, and surface roughness α, respectively. In this way, our combined geometry and reflectance parameters are given by $\boldsymbol{\xi} = (\boldsymbol{p}, \boldsymbol{u}, a_\mathrm{d}, a_\mathrm{s}, \alpha)$. Note, we do not modify the connectivity of the triangle vertices and rely on additional re-meshing steps, which we will discuss in §3.4, to improve mesh topology.

Loss. A key ingredient in our analysis-by-synthesis optimization is the loss $\mathcal{L}$. Let $\tilde{\boldsymbol{I}} := (\tilde{I}_1, \tilde{I}_2, \ldots)$ be a set of images of some object (with camera location and pose calibrated for each image $\tilde{I}_k$). Then, our loss takes the form:

$$\mathcal{L}(\boldsymbol{I}(\boldsymbol{\xi}), \boldsymbol{\xi};\ \tilde{\boldsymbol{I}}) := \mathcal{L}_\mathrm{rend}(\boldsymbol{I}(\boldsymbol{\xi});\ \tilde{\boldsymbol{I}}) + \mathcal{L}_\mathrm{reg}(\boldsymbol{\xi}), \tag{6}$$

where $\mathcal{L}_\mathrm{rend}$ is the *rendering* loss that measures the difference between rendered and target object appearances. Specifically, we set

$$\mathcal{L}_\mathrm{rend}(\boldsymbol{I}(\boldsymbol{\xi});\ \tilde{\boldsymbol{I}}) := \lambda_\mathrm{rend}\sum_k \left\|\Phi_k(I_k(\boldsymbol{\xi})) - \Phi_k(\tilde{I}_k)\right\|_1, \tag{7}$$

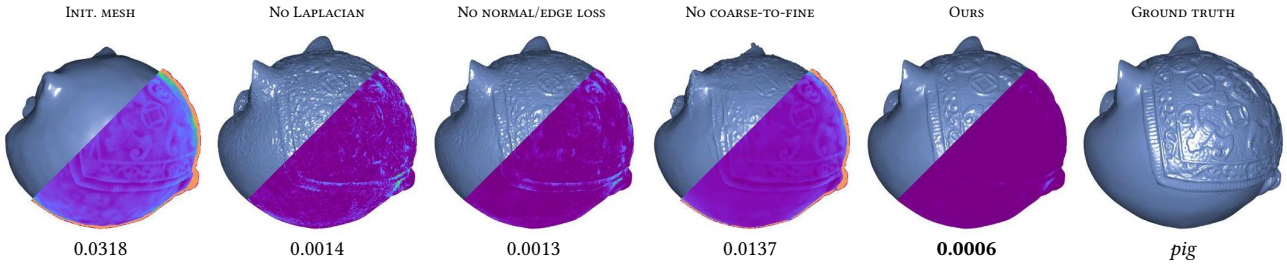

Fig. 4. **Ablation study** on our mesh loss of Eq. (9) and coarse-to-fine framework. Using the identical initializations and optimization settings, we show geometries (rendered under a novel view) optimized with (i) various components of the mesh loss; and (ii) the coarse-to-fine process disabled. Similar to Figure 3, the number below each result indicates the average point-to-mesh distance.

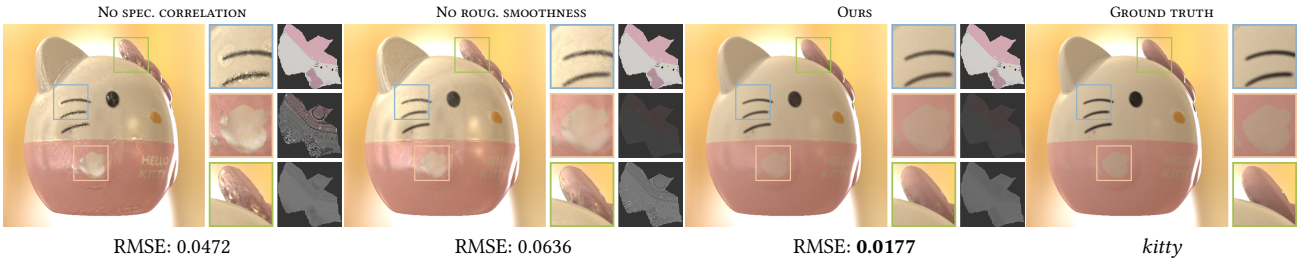

Fig. 5. **Ablation study** on our material loss of Eq. (10). Using identical initial reflectance maps and optimization configurations, models optimized with various components of the material loss neglected are rendered under a novel environmental illumination. On the right of each reconstruction result, we show the optimized reflectance maps (from top to bottom: diffuse albedo, specular albedo, and roughness).

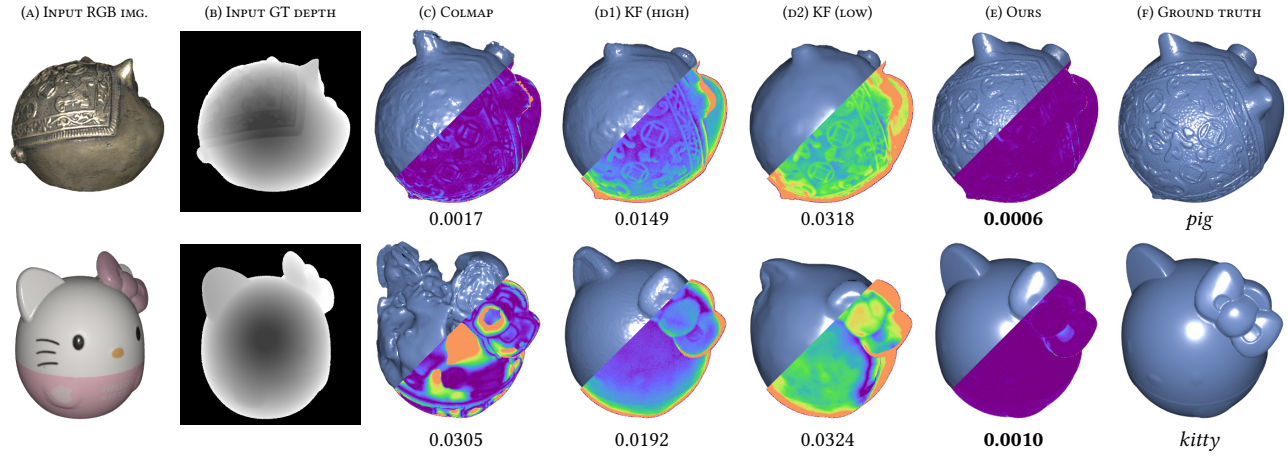

Fig. 6. **Comparison** with COLMAP [Schönberger et al. 2016] and Kinect Fusion [Newcombe et al. 2011] using synthetic inputs. The *COLMAP* results (c) are generated using 50 RGB images (a) with exact camera poses; the *KF-High* (d1) and *KF-Low* (d2) results are created using 50 ground-truth depth images (b) and low-resolution noisy ones, respectively. Our method (e), when initialized with *KF-Low* (d2) and using RGB inputs (a), produces much more accurate geometries than *COLMAP* and *KF-High*. Similar to Figure 3, the numbers indicate average point-to-mesh distances.

where $\lambda_{\text{rend}} > 0$ is a user-specified weight, $\boldsymbol{I}(\boldsymbol{\xi}) := (\boldsymbol{I}_1(\boldsymbol{\xi}), \boldsymbol{I}_2(\boldsymbol{\xi}), \ldots)$ denotes images rendered using our forward-rendering model of Eq. (3) with object geometry and reflectance specified by $\boldsymbol{\xi}$ (under identical camera configurations as the input images), and Φ_k captures pixel-wise post-processing operations such as tone-mapping and background-removing masking.

We estimate gradients of the rendering loss of Eq. (7) with respect to the object parameters $\boldsymbol{\xi}$ using our differentiable rendering method

described in §3.1. We will demonstrate in §4.1 that accurate gradients are crucial to obtain high-quality optimization results.

In Eq. (6), $\mathcal{L}_{\text{reg}}(\boldsymbol{\xi})$ is a *regularization* term for improving the robustness of the optimization, which we will discuss in §3.3. Gradients of this term can be obtained easily using automatic differentiation.

Optimization process. Like any other analysis-by-synthesis method, our technique takes as input an initial configuration of an object's geometry and reflectance. In practice, we initialize object geometry using MVS or Kinect Fusion. Our technique is capable of producing high-quality reconstructions using crude initializations (obtained using low-resolution and noisy inputs). For the reflectance maps, we simply initialize them as constant-valued textures.

Provided an initial configuration of the object's geometry and reflectance, we minimize the loss of Eq. (6) using the Adam algorithm [Kingma and Ba 2014].

Further, to make the optimization more robust, we leverage a coarse-to-fine approach that periodically performs remeshing and upsamples the reflectance-describing textures. We will provide more details on this process in §3.4.

3.3 Regularization

Using only the rendering loss $\mathcal{L}_{\text{rend}}$ expressed in Eq. (7) can make the optimization unstable and/or converge to local minima. To address this problem, we regularize the optimization by introducing another loss $\mathcal{L}_{\text{reg}}$ that in turn consists of a *material* loss $\mathcal{L}_{\text{mat}}$ and a *mesh* one $\mathcal{L}_{\text{mesh}}$:

$$\mathcal{L}_{\text{reg}}(\boldsymbol{\xi}) := \mathcal{L}_{\text{mesh}}(\mathcal{M}) + \mathcal{L}_{\text{mat}}(a_{\text{d}}, a_{\text{s}}, \alpha), \tag{8}$$

which we will discuss in the following.

Mesh loss. We encourage our optimization to return "smooth" object geometry by introducing a *mesh loss*:

$$\mathcal{L}_{\text{mesh}}(\mathcal{M}) := \mathcal{L}_{\text{lap}}(\mathcal{M}) + \mathcal{L}_{\text{normal}}(\mathcal{M}) + \mathcal{L}_{\text{edge}}(\mathcal{M}), \tag{9}$$

where the *mesh-Laplacian loss* $\mathcal{L}_{\text{lap}}$ of a mesh with n vertices is given by $\mathcal{L}_{\text{lap}}(\mathcal{M}) := \lambda_{\text{lap}}\|\boldsymbol{L}\boldsymbol{V}\|^2$ where $\boldsymbol{V}$ is an $n \times 3$ matrix with its i-th row storing coordinates of the i-th vertex, and $\boldsymbol{L} \in \mathbb{R}^{n\times n}$ is the mesh's Laplacian matrix [Nealen et al. 2006].

Additionally, we use a *normal-consistency loss* $\mathcal{L}_{\text{normal}}$ to encourage normals of adjacent faces to vary slowly by setting $\mathcal{L}_{\text{normal}}(\mathcal{M}) := \lambda_{\text{normal}} \sum_{i,j}[1 - (\boldsymbol{n}_i \cdot \boldsymbol{n}_j)]^2$, where the sum is over all pairs (i, j) such that the i-th and the j-th triangles share a common edge, and $\boldsymbol{n}_i$ and $\boldsymbol{n}_j$ denote the normals of these triangles.

Lastly, we penalize the mesh for having long edges, which usually yield ill-shaped triangles, by utilizing an *edge-length loss* $\mathcal{L}_{\text{edge}} := \lambda_{\text{edge}} (\sum_i e_i^2)^{1/2}$, where e_i denotes the length of the i-th face edge.

Material loss. Our material loss $\mathcal{L}_{\text{mat}}$ regularizes the reflectance maps representing diffuse albedo a_{d}, specular albedo a_{s}, and surface roughness α. Specifically, we set

$$\mathcal{L}_{\text{mat}}(a_{\text{d}}, a_{\text{s}}, \alpha) := \mathcal{L}_{\text{spec}}(a_{\text{d}}, a_{\text{s}}) + \mathcal{L}_{\text{roug}}(\alpha), \tag{10}$$

where $\mathcal{L}_{\text{spec}}$ correlates diffuse and specular albedos [Schmitt et al. 2020]: assuming nearby pixels with similar diffuse albedos to have similar specular ones, we set $\mathcal{L}_{\text{spec}}(a_{\text{s}}, a_{\text{d}}) := \lambda_{\text{spec}} \sum_{\boldsymbol{p}} \|a_{\text{s}}[\boldsymbol{p}] - (\sum_{\boldsymbol{q}} a_{\text{s}}[\boldsymbol{q}]\, \mu_{\boldsymbol{p},\boldsymbol{q}})/(\sum_{\boldsymbol{q}} \mu_{\boldsymbol{p},\boldsymbol{q}})\|_1$, where $\mu_{\boldsymbol{p},\boldsymbol{q}} := \exp\left(-\frac{\|\boldsymbol{p}-\boldsymbol{q}\|_2^2}{2\sigma_1^2} - \frac{(a_{\text{d}}[\boldsymbol{p}]-a_{\text{d}}[\boldsymbol{q}])^2}{2\sigma_2^2}\right)$ is the bilateral weight between pixels with indices $\boldsymbol{p}, \boldsymbol{q} \in \mathbb{Z}^2$.

Table 1. **Rendering performance**. We report the rendering cost in seconds (averaged across 100 times) of each differentiable renderer in resolution 512×512 and 4 samples per pixel on a Titan RTX graphics card.

	SoftRas	PyTorch3D	Mitsuba 2	Redner	Ours
Kettle	0.0184	0.0202	0.0877	0.1196	**0.0143**
Maneki	0.0192	0.0224	0.0863	0.1029	**0.0146**
Pig	0.0971	0.0772	0.0913	0.1336	**0.0263**
Kitty	0.0249	0.0334	0.0889	0.1190	**0.0225**

Spatially varying surface roughness is known to be challenging to optimize even when the object geometry is known [Gao et al. 2019b]. To regularize our optimization of surface roughness, we introduce a smoothness term that measures its total variation: $\mathcal{L}_{\text{roug}}(\alpha) := \lambda_{\text{roug}} \sum_{i,j}(|\alpha[i+1, j] - \alpha[i, j]| + |\alpha[i, j+1] - \alpha[i, j]|)$, where $\alpha[i, j]$ indicate the value of the (i, j)-th pixel in the roughness map.

3.4 Improving robustness

As described in §3.2, when minimizing the loss of Eq. (6), we keep the mesh topology unchanged. This, unfortunately, can severely limit the flexibility of our optimization of object geometry, making the result highly sensitive to the quality of the initial mesh. Additionally, without taking precautions, updating vertex positions can introduce artifacts (e.g., self intersections) to the mesh that cannot be easily fixed by later iterations.

To address these problems, we utilize a few extra steps.

Coarse-to-fine optimization. Instead of performing the entire optimization at a single resolution, we utilize a coarse-to-fine process for improved robustness. Similar steps have been taken in several prior works, although typically limited to either geometry [Kazhdan and Hoppe 2013; Sahillioğlu and Yemez 2010; Sharf et al. 2006; Tsai et al. 2019] or reflectance [Dong et al. 2014; Hui et al. 2017; Riviere et al. 2016].

Specifically, we start the optimization by using low-resolution meshes and reflectance maps. If the input already has high resolutions, we simplify them via remeshing and image downsampling. Our optimization process then involves multiple stages. During each stage, we iteratively refine the object geometry and reflectance with fixed mesh topology. After each stage (except the final one), we upsample the mesh (using instant meshes [Jakob et al. 2015]) and the texture maps (using simple bilinear interpolation).

Robust surface evolution. During optimization, if the vertex positions are updated naïvely (i.e., using simple gradient-based updates with no validation checks), the mesh can have degraded quality and even become non-manifold (i.e., with artifacts like holes and self-intersections). Motivated by other optimization-driven mesh editing algorithms [Liu et al. 2018; Sacht et al. 2015], we evolve a mesh using a pipeline implemented in the El Topo library [Brochu et al. 2009]: Given the initial positions of a set of vertices with associated displacements, El Topo moves each vertex along its displacement vector while ensuring no self-intersection is generated.

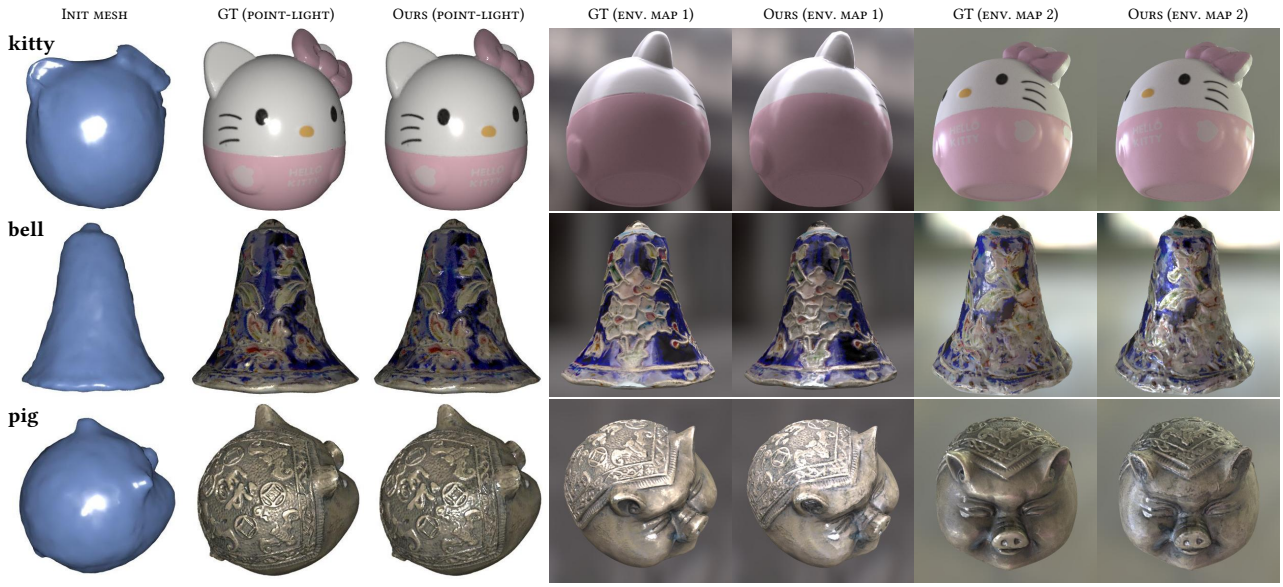

Fig. 7. **Reconstruction results** using synthetic inputs: We obtain the initial meshes (in the left column) using Kinect Fusion [Newcombe et al. 2011] with low-resolution (48×48) and noisy depths. Our analysis-by-synthesis pipeline successfully recovers both geometric and reflectance details, producing high-fidelity results under novel viewing and illumination conditions.

4 Results

We implement our differentiable renderer (§3.1) in C++ with CUDA 11 and OptiX 7.1. For vectorized and differentiable computations, we utilize the Enoki library [Jakob 2019], which has been demonstrated to be more efficient than generic ones like Tensorflow and PyTorch for rendering [Nimier-David et al. 2019].

We implement the rest of our optimization pipeline, including the loss computations, in PyTorch. We use one set of weights for our optimizations: $\lambda_{\text{rend}} = 1$ for the rendering loss of Eq. (7); $(\lambda_{\text{lap}}, \lambda_{\text{edge}}, \lambda_{\text{normal}}) = (0.1, 1, 0.01)$ for the mesh loss of Eq. (9) and $(\lambda_{\text{spec}}, \lambda_{\text{roug}}) = (0.01, 0.001)$ for the material loss of Eq. (10).

In practice, our optimization involves 500–1000 iterations (for all coarse-to-fine stages) and takes 0.5–2 hours per example (see the supplement for more details).

4.1 Evaluations and comparisons

Comparison with differentiable renderers. Our renderer enjoys very high performance, thanks to its specialized nature (i.e., focused on the collocated configuration) and the combined efficiency of RTX ray tracing offered by OptiX and GPU-based differentiable computations by Enoki. As demonstrated in Table 1, our renderer is faster than SoftRas [Liu et al. 2019], PyTorch3D [Ravi et al. 2020], and Mitsuba 2 [Nimier-David et al. 2019] without the need to introduce bias to the gradient estimates. Compared to Redner [Li et al. 2018a], another differentiable renderer that produces unbiased gradients, our renderer enjoys an even greater advantage in performance.

To further demonstrate the importance for having accurate gradients, we conduct a synthetic experiment where the shape of an object is optimized (with known diffuse reflectance). Using identical input images, initial configurations, losses, and optimization settings (e.g., learning rate), we ran multiple optimizations using Adam [Kingma and Ba 2014] with gradients produced by SoftRas, PyTorch3D, Mitsuba 2, and our technique, respectively. As shown in Figure 3, using biased gradients yields various artifacts in the optimized results.

Effectiveness of shape optimization. To ensure robustness when optimizing the shape of an object, our technique utilizes a mesh loss (§3.3) as well as a coarse-to-fine framework (§3.4). We conduct another experiment to evaluate the effectiveness of these steps. Specifically, we optimize the shape of the *pig* model using identical optimization configurations except for (i) having various components of the mesh loss turned off; and (ii) not using the coarse-to-fine framework. As shown in Figure 4, with the mesh Laplacian loss $\mathcal{L}_{\text{lap}}$ neglected (by setting $\lambda_{\text{lap}} = 0$), the resulting geometry becomes "bumpy"; without the normal and edge-length losses $\mathcal{L}_{\text{normal}}$ and $\mathcal{L}_{\text{edge}}$, the optimized geometry also has artifacts due to sharp normal changes and ill-shaped triangles. Additionally, without performing the optimization in a coarse-to-fine fashion (by directly starting with a subdivided version of the initial mesh), the optimization gets stuck in a local optimum (with all losses enabled).

To further evaluate the effectiveness of our technique for recovering object geometry, we compare with two baseline methods: COLMAP [Schönberger et al. 2016] and Kinect Fusion [Newcombe et al. 2011]. As demonstrated in Figure 6, our technique, when using crude initial geometries (obtained using Kinect Fusion with low-resolution and noisy depth images), produces results with much higher quality than the baselines. COLMAP fails badly for the *kitty* example since the object contains insufficient textures for correspondences to be reliable established.

Fig. 8. **Reconstruction results** of real-world objects: Similar to the synthetic examples in Figure 7, our technique recovers detailed object geometries and reflectance details, producing high-quality results under novel viewing and illumination conditions.

Effectiveness of material loss. Our material loss of Eq. (10) is important for obtaining clean reflectance maps that generalize well to novel settings. As shown in Figure 5, without correlating diffuse and specular albedos (by having $\lambda_{\text{spec}} = 0$), diffuse colors are "baked" into specular albedo, leading to heavy artifacts under novel environmental illumination. With the roughness smoothness $\mathcal{L}_{\text{roug}}$ disabled, the resulting roughness map contains high-frequency noise that leads to artifacts in rendered specular highlights.

4.2 Reconstruction results

Figure 7 shows reconstruction results obtained using synthetic input images and rendered under novel views and illuminations. The initial geometries are obtained using Kinect Fusion with low-resolution noisy depth inputs. Using 50 input images, our technique offers the robustness for recovering both smooth (e.g., the *kitty* example) and detailed (e.g., the *pig* example) geometries and reflectance.

We show in Figure 8 reconstruction results using as input 100 real photographs per example. The initial geometries are obtained using COLMAP. Our analysis-by-synthesis technique manages to accurately recover the detailed geometry and reflectance of each model.

We note that, in Figures 7 and 8, the detailed geometric structures (e.g., those in the *bell*, *pig*, *chime*, and *buddha* examples) fully emerge from the mesh-based object geometries: no normal or displacement mapping is used.

Lastly, since our reconstructed models use standard mesh-based representations, they can be used in a broad range of applications (see Figure 9).

5 Conclusion and Discussion

We introduce a new approach to jointly recover the shape and reflectance of real-world objects. At the core of our technique is a unified analysis-by-synthesis pipeline that iteratively refines object geometry and reflectance. Our custom Monte Carlo differentiable renderer enjoys higher performance than many existing tools (such as SoftRas, PyTorch3D, and Mitsuba 2). More importantly, our renderer produces unbiased geometric gradients that are crucial for obtaining high-quality reconstructions. To further improve the robustness of our optimization, we leverage a coarse-to-fine framework regularized using a few geometric and reflectance priors. We conduct several ablation studies to evaluate the effectiveness of our differentiable renderer, losses, and optimization strategies.

Fig. 9. **Applications:** The high-quality models generated by our technique can be used for 3D digital modeling (top) and object insertion in augmented reality (bottom).

Limitations and future work. Our technique is specialized to using a collocated camera and point light. Generalization to other configurations would be useful in the future.

To refine mesh topology, our technique relies on remeshing steps (between coarse-to-fine stages). How topology can be optimized in a robust and flexible fashion is an important problem for future investigation.

References

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

Miika Aittala, Tim Weyrich, Jaakko Lehtinen, et al. 2015. Two-shot SVBRDF capture for stationary materials. *ACM Trans. Graph.* 34, 4 (2015), 110–1.

Rachel A Albert, Dorian Yao Chan, Dan B Goldman, and James F O'Brien. 2018. Approximate svBRDF estimation from mobile phone video. In *Proc. EGSR: Experimental Ideas & Implementations*. Eurographics Association, 11–22.

Dejan Azinovic, Tzu-Mao Li, Anton Kaplanyan, and Matthias Nießner. 2019. Inverse path tracing for joint material and lighting estimation. In *Proc. IEEE/CVF CVPR*. 2447–2456.

Sai Bi, Zexiang Xu, Kalyan Sunkavalli, David Kriegman, and Ravi Ramamoorthi. 2020. Deep 3D Capture: Geometry and Reflectance from Sparse Multi-View Images. In *Proc. IEEE/CVF CVPR*. 5960–5969.

Tyson Brochu et al. 2009. El Topo: Robust Topological Operations for Dynamic Explicit Surfaces. https://github.com/tysonbrochu/eltopo.

Chengqian Che, Fujun Luan, Shuang Zhao, Kavita Bala, and Ioannis Gkioulekas. 2020. Towards Learning-based Inverse Subsurface Scattering. *ICCP* (2020), 1–12.

Guojun Chen, Yue Dong, Pieter Peers, Jiawan Zhang, and Xin Tong. 2014. Reflectance scanning: estimating shading frame and BRDF with generalized linear light sources. *ACM Trans. Graph.* 33, 4 (2014), 1–11.

Valentin Deschaintre, Miika Aittala, Frédo Durand, George Drettakis, and Adrien Bousseau. 2019. Flexible SVBRDF Capture with a Multi-Image Deep Network. In *Computer Graphics Forum*, Vol. 38. Wiley Online Library, 1–13.

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

Yue Dong, Jiaping Wang, Xin Tong, John Snyder, Yanxiang Lan, Moshe Ben-Ezra, and Baining Guo. 2010. Manifold bootstrapping for SVBRDF capture. *ACM Trans. Graph.* 29, 4 (2010), 1–10.

Zhao Dong, Bruce Walter, Steve Marschner, and Donald P Greenberg. 2015. Predicting appearance from measured microgeometry of metal surfaces. *ACM Trans. Graph.* 35, 1 (2015), 1–13.

Yasutaka Furukawa and Jean Ponce. 2009. Accurate, dense, and robust multiview stereopsis. *IEEE Trans. Pattern Anal. Mach. Intell.* 32, 8 (2009), 1362–1376.

Duan Gao, Xiao Li, Yue Dong, Pieter Peers, Kun Xu, and Xin Tong. 2019a. Deep inverse rendering for high-resolution SVBRDF estimation from an arbitrary number of images. *ACM Trans. Graph.* 38, 4 (2019).

Duan Gao, Xiao Li, Yue Dong, Pieter Peers, Kun Xu, and Xin Tong. 2019b. Deep inverse rendering for high-resolution SVBRDF estimation from an arbitrary number of images. *ACM Trans. Graph.* 38, 4 (2019), 134–1.

Abhijeet Ghosh, Tim Hawkins, Pieter Peers, Sune Frederiksen, and Paul Debevec. 2008. Practical Modeling and Acquisition of Layered Facial Reflectance. *ACM Trans. Graph.* 27, 5, Article 139 (Dec. 2008), 10 pages.

Ioannis Gkioulekas, Anat Levin, and Todd Zickler. 2016. An evaluation of computational imaging techniques for heterogeneous inverse scattering. In *ECCV*. Springer, 685–701.

Ioannis Gkioulekas, Shuang Zhao, Kavita Bala, Todd Zickler, and Anat Levin. 2013. Inverse volume rendering with material dictionaries. *ACM Trans. Graph.* 32, 6 (2013), 1–13.

Yu Guo, Cameron Smith, Miloš Hašan, Kalyan Sunkavalli, and Shuang Zhao. 2020. MaterialGAN: Reflectance Capture using a Generative SVBRDF Model. *ACM Trans. Graph.* 39, 6 (2020), 254:1–254:13.

Bjoern Haefner, Yvain Quéau, Thomas Möllenhoff, and Daniel Cremers. 2018. Fight ill-posedness with ill-posedness: Single-shot variational depth super-resolution from shading. In *Proc. IEEE/CVF CVPR*. 164–174.

Tomoaki Higo, Yasuyuki Matsushita, Neel Joshi, and Katsushi Ikeuchi. 2009. A hand-held photometric stereo camera for 3-d modeling. In *Proc. ICCV*. IEEE, 1234–1241.

Michael Holroyd, Jason Lawrence, Greg Humphreys, and Todd Zickler. 2008. A photometric approach for estimating normals and tangents. *ACM Trans. Graph.* 27, 5 (2008), 1–9.

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

Berthold KP Horn. 1970. Shape from shading: A method for obtaining the shape of a smooth opaque object from one view. (1970).

Zhuo Hui, Kalyan Sunkavalli, Joon-Young Lee, Sunil Hadap, Jian Wang, and Aswin C Sankaranarayanan. 2017. Reflectance capture using univariate sampling of BRDFs. In *ICCV*. IEEE, 5362–5370.

Katsushi Ikeuchi and Berthold KP Horn. 1981. Numerical shape from shading and occluding boundaries. *Artificial intelligence* 17, 1-3 (1981), 141–184.

Shahram Izadi, David Kim, Otmar Hilliges, David Molyneaux, Richard Newcombe, Pushmeet Kohli, Jamie Shotton, Steve Hodges, Dustin Freeman, Andrew Davison, and Andrew Fitzgibbon. 2011. KinectFusion: Real-Time 3D Reconstruction and Interaction Using a Moving Depth Camera. In *Proc. ACM UIST*. 559–568.

Wenzel Jakob. 2019. Enoki: structured vectorization and differentiation on modern processor architectures. https://github.com/mitsuba-renderer/enoki.

Wenzel Jakob, Marco Tarini, Daniele Panozzo, and Olga Sorkine-Hornung. 2015. Instant Field-Aligned Meshes. *ACM Trans. Graph.* 34, 6 (2015).

Rasmus Jensen, Anders Dahl, George Vogiatzis, Engin Tola, and Henrik Aanæs. 2014. Large scale multi-view stereopsis evaluation. In *Proc. IEEE CVPR*. 406–413.

Yue Jiang, Dantong Ji, Zhizhong Han, and Matthias Zwicker. 2020. Sdfdiff: Differentiable rendering of signed distance fields for 3D shape optimization. In *Proc. IEEE/CVF CVPR*. 1251–1261.

Kaizhang Kang, Zimin Chen, Jiaping Wang, Kun Zhou, and Hongzhi Wu. 2018. Efficient reflectance capture using an autoencoder. *ACM Trans. Graph.* 37, 4 (2018), 127–1.

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

Hiroharu Kato, Deniz Beker, Mihai Morariu, Takahiro Ando, Toru Matsuoka, Wadim Kehl, and Adrien Gaidon. 2020. Differentiable Rendering: A Survey. arXiv:2006.12057 [cs.CV]

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

Kihwan Kim, Jinwei Gu, Stephen Tyree, Pavlo Molchanov, Matthias Nießner, and Jan Kautz. 2017. A lightweight approach for on-the-fly reflectance estimation. In *ICCV*. IEEE, 20–28.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

Christoph Lassner. 2020. Fast Differentiable Raycasting for Neural Rendering using Sphere-based Representations. *arXiv preprint arXiv:2004.07484* (2020).

Hendrik PA Lensch, Jan Kautz, Michael Goesele, Wolfgang Heidrich, and Hans-Peter Seidel. 2003. Image-based reconstruction of spatial appearance and geometric detail. *ACM Trans. Graph.* 22, 2 (2003), 234–257.

Tzu-Mao Li, Miika Aittala, Frédo Durand, and Jaakko Lehtinen. 2018a. Differentiable Monte Carlo ray tracing through edge sampling. *ACM Trans. Graph.* 37, 6 (2018), 1–11.

Zhengqin Li, Mohammad Shafiei, Ravi Ramamoorthi, Kalyan Sunkavalli, and Manmohan Chandraker. 2020. Inverse rendering for complex indoor scenes: Shape, spatially-varying lighting and SVBRDF from a single image. In *Proc. IEEE/CVF CVPR*. 2475–2484.

Zhengqin Li, Kalyan Sunkavalli, and Manmohan Chandraker. 2018b. Materials for masses: SVBRDF acquisition with a single mobile phone image. In *ECCV*. Springer, 72–87.

Hsueh-Ti Derek Liu, Michael Tao, and Alec Jacobson. 2018. Paparazzi: surface editing by way of multi-view image processing. *ACM Trans. Graph.* 37, 6 (2018), 221–1.

Shichen Liu, Tianye Li, Weikai Chen, and Hao Li. 2019. Soft rasterizer: A differentiable renderer for image-based 3D reasoning. In *ICCV*. IEEE, 7708–7717.

Guillaume Loubet, Nicolas Holzschuch, and Wenzel Jakob. 2019. Reparameterizing discontinuous integrands for differentiable rendering. *ACM Trans. Graph.* 38, 6 (2019), 1–14.

Robert Maier, Kihwan Kim, Daniel Cremers, Jan Kautz, and Matthias Nießner. 2017. Intrinsic3d: High-quality 3D reconstruction by joint appearance and geometry optimization with spatially-varying lighting. In *ICCV*. IEEE, 3114–3122.

Wojciech Matusik. 2003. *A data-driven reflectance model.* Ph.D. Dissertation. Massachusetts Institute of Technology.

Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. 2019. Occupancy networks: Learning 3D reconstruction in function space. In *Proc. IEEE/CVF CVPR*. 4460–4470.

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

Giljoo Nam, Joo Ho Lee, Hongzhi Wu, Diego Gutierrez, and Min H Kim. 2016. Simultaneous acquisition of microscale reflectance and normals. *ACM Trans. Graph.* 35, 6 (2016), 185–1.

Andrew Nealen, Takeo Igarashi, Olga Sorkine, and Marc Alexa. 2006. Laplacian mesh optimization. In *Proc. the 4th international conference on Computer graphics and interactive techniques in Australasia and Southeast Asia*. 381–389.

Richard A Newcombe, Shahram Izadi, Otmar Hilliges, David Molyneaux, David Kim, Andrew J Davison, Pushmeet Kohli, Jamie Shotton, Steve Hodges, and Andrew Fitzgibbon. 2011. KinectFusion: Real-time dense surface mapping and tracking. In *ISMAR*. IEEE, 127–136.

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

Thoma Papadhimitri and Paolo Favaro. 2014. Uncalibrated near-light photometric stereo. (2014).

Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. 2019. Deepsdf: Learning continuous signed distance functions for shape representation. In *Proc. IEEE/CVF CVPR*. 165–174.

Jeong Joon Park, Richard Newcombe, and Steve Seitz. 2018. Surface light field fusion. In *3DV*. IEEE, 12–21.

Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. 2017. Automatic differentiation in pytorch. (2017).

Yvain Quéau, François Lauze, and Jean-Denis Durou. 2015. Solving uncalibrated photometric stereo using total variation. *Journal of Mathematical Imaging and Vision* 52, 1 (2015), 87–107.

Yvain Quéau, Roberto Mecca, and Jean-Denis Durou. 2016. Unbiased photometric stereo for colored surfaces: A variational approach. In *Proc. IEEE CVPR*. 4359–4368.

Yvain Quéau, Jean Mélou, Fabien Castan, Daniel Cremers, and Jean-Denis Durou. 2017a. A variational approach to shape-from-shading under natural illumination. In *International Workshop on Energy Minimization Methods in Computer Vision and Pattern Recognition*. Springer, 342–357.

Yvain Quéau, Jean Mélou, Jean-Denis Durou, and Daniel Cremers. 2017b. Dense multi-view 3D-reconstruction without dense correspondences. *arXiv preprint arXiv:1704.00337* (2017).

Nikhila Ravi, Jeremy Reizenstein, David Novotny, Taylor Gordon, Wan-Yen Lo, Justin Johnson, and Georgia Gkioxari. 2020. Accelerating 3D deep learning with PyTorch3D. *arXiv preprint arXiv:2007.08501* (2020).

Peiran Ren, Jiaping Wang, John Snyder, Xin Tong, and Baining Guo. 2011. Pocket reflectometry. *ACM Trans. Graph.* 30, 4 (2011), 1–10.

Osborne Reynolds. 1903. *Papers on mechanical and physical subjects: the sub-mechanics of the universe*. Vol. 3. The University Press.

Jérémy Riviere, Pieter Peers, and Abhijeet Ghosh. 2016. Mobile surface reflectometry. In *Computer Graphics Forum*, Vol. 35. Wiley Online Library, 191–202.

Jérémy Riviere, Ilya Reshetouski, Luka Filipi, and Abhijeet Ghosh. 2017. Polarization imaging reflectometry in the wild. *ACM Trans. Graph.* 36, 6 (2017), 1–14.

Leonardo Sacht, Etienne Vouga, and Alec Jacobson. 2015. Nested cages. *ACM Trans. Graph.* 34, 6 (2015), 1–14.

Y Sahillioğlu and Yücel Yemez. 2010. Coarse-to-fine surface reconstruction from silhouettes and range data using mesh deformation. *Computer Vision and Image Understanding* 114, 3 (2010), 334–348.

Carolin Schmitt, Simon Donne, Gernot Riegler, Vladlen Koltun, and Andreas Geiger. 2020. On Joint Estimation of Pose, Geometry and svBRDF From a Handheld Scanner. In *Proc. IEEE/CVF CVPR*. 3493–3503.

Johannes L Schönberger, Enliang Zheng, Jan-Michael Frahm, and Marc Pollefeys. 2016. Pixelwise view selection for unstructured multi-view stereo. In *ECCV*. Springer, 501–518.

Johannes Lutz Schönberger, Enliang Zheng, Marc Pollefeys, and Jan-Michael Frahm. 2016. Pixelwise View Selection for Unstructured Multi-View Stereo. In *ECCV*. Springer.

Christopher Schwartz, Ralf Sarlette, Michael Weinmann, and Reinhard Klein. 2013. DOME II: A Parallelized BTF Acquisition System.. In *Material Appearance Modeling*. 25–31.

Steven M Seitz and Charles R Dyer. 1999. Photorealistic scene reconstruction by voxel coloring. *International Journal of Computer Vision* 35, 2 (1999), 151–173.

Andrei Sharf, Thomas Lewiner, Ariel Shamir, Leif Kobbelt, and Daniel Cohen-Or. 2006. Competing fronts for coarse–to–fine surface reconstruction. In *Computer Graphics Forum*, Vol. 25. Wiley Online Library, 389–398.

Chia-Yin Tsai, Aswin C Sankaranarayanan, and Ioannis Gkioulekas. 2019. Beyond Volumetric Albedo–A Surface Optimization Framework for Non-Line-Of-Sight Imaging. In *Proc. IEEE/CVF CVPR*. 1545–1555.

Borom Tunwattanapong, Graham Fyffe, Paul Graham, Jay Busch, Xueming Yu, Abhijeet Ghosh, and Paul Debevec. 2013. Acquiring reflectance and shape from continuous spherical harmonic illumination. *ACM Trans. Graph.* 32, 4 (2013), 1–12.

George Vogiatzis, Philip HS Torr, and Roberto Cipolla. 2005. Multi-view stereo via volumetric graph-cuts. In *Proc. IEEE CVPR*, Vol. 2. 391–398.

Robert J Woodham. 1980. Photometric method for determining surface orientation from multiple images. *Optical engineering* 19, 1 (1980), 191139.

Hongzhi Wu, Zhaotian Wang, and Kun Zhou. 2015. Simultaneous localization and appearance estimation with a consumer RGB-D camera. *IEEE TVCG* 22, 8 (2015), 2012–2023.

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

Yizhou Yu, Paul Debevec, Jitendra Malik, and Tim Hawkins. 1999. Inverse global illumination: Recovering reflectance models of real scenes from photographs. In *Proc. Computer graphics and interactive techniques*. 215–224.

Cheng Zhang, Lifan Wu, Changxi Zheng, Ioannis Gkioulekas, Ravi Ramamoorthi, and Shaung Zhao. 2019. A differential theory of radiative transfer. *ACM Trans. Graph.* 38, 6 (2019), 227:1–227:16.

Zhiming Zhou, Guojun Chen, Yue Dong, David Wipf, Yong Yu, John Snyder, and Xin Tong. 2016a. Sparse-as-Possible SVBRDF Acquisition. *ACM Trans. Graph.* 35, 6, Article 189 (2016), 12 pages.

Zhiming Zhou, Guojun Chen, Yue Dong, David Wipf, Yong Yu, John Snyder, and Xin Tong. 2016b. Sparse-as-possible SVBRDF acquisition. *ACM Trans. Graph.* 35, 6 (2016), 1–12.

Zhenglong Zhou and Ping Tan. 2010. Ring-light photometric stereo. In *ECCV*. Springer, 265–279.