VMINer: Versatile Multi-view Inverse Rendering with Near- and Far-field Light Sources

Fan Fei[1,2,4] Jiajun Tang[1,2,4] Ping Tan[3,4] Boxin Shi[1,2#]

[1]National Key Laboratory for Multimedia Information Processing, School of Computer Science, Peking University
[2]National Engineering Research Center of Visual Technology, School of Computer Science, Peking University
[3]Hong Kong University of Science and Technology [4]Light Illusions

Abstract

This paper introduces a versatile multi-view inverse rendering framework with near- and far-field light sources. Tackling the fundamental challenge of inherent ambiguity in inverse rendering, our framework adopts a lightweight yet inclusive lighting model for different near- and far-field lights, thus is able to make use of input images under varied lighting conditions available during capture. It leverages observations under each lighting to disentangle the intrinsic geometry and material from the external lighting, using both neural radiance field rendering and physically-based surface rendering on the 3D implicit fields. After training, the reconstructed scene is extracted to a textured triangle mesh for seamless integration into industrial rendering software for various applications. Quantitatively and qualitatively tested on synthetic and real-world scenes, our method shows superiority to state-of-the-art multi-view inverse rendering methods in both speed and quality.

1. Introduction

The reconstruction of 3D scenes from multi-view RGB imagery has experienced significant advancements following the development of Neural Radiance Fields (NeRF) [23], and both the speed and quality of the reconstruction have reached an unprecedented level [22, 24]. Despite these improvements, a common limitation is to represent only the radiance field, which is a complex product of the external lighting interacting with the intrinsic geometry and material of the scene [13]. The entangled nature of this representation generally hampers the ability to accurately render the scene under unseen lighting conditions, because the influence of the original lighting is embedded within the newly rendered scene. Recent approaches [34, 51, 56] have incorporated inverse rendering [27] to separate material properties and lighting effects, extending the application of the reconstruction beyond novel view synthesis to novel scenarios involving relighting and material editing [2]. Some methods [2, 6, 11, 25, 42] have advanced to the extraction of detailed triangle meshes with material UV textures, which can serve as economical and lifelike 3D models for gaming and cinematography industries [11, 25], thereby marking a transformative step in digital asset creation.

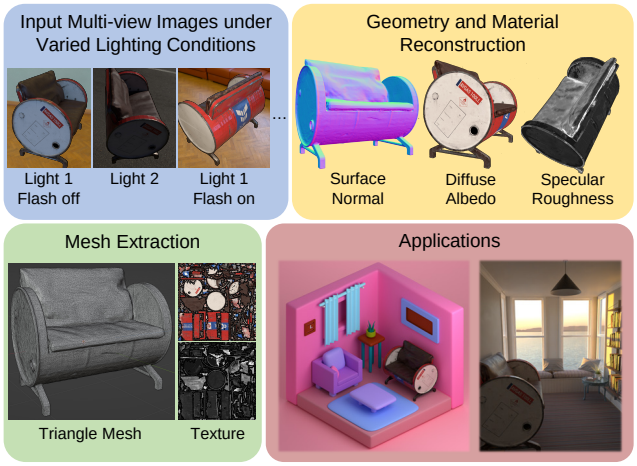

Figure 1. Given multi-view images under possibly varied lighting, our method leverages all present lighting conditions to reconstruct scene geometry and material disentangled with lighting. The trained fields are extracted to textured meshes for seamless integration into industrial renderers for various applications.

Inverse rendering presents several fundamental challenges, one of them being its severe inherent ambiguity [51]. To combat this, multi-view inverse rendering methodologies usually reduce the ambiguity by imposing various constraints on each scene component. These methods can be categorized based on the assumption on the amount and types of lighting conditions present in the input images (Tab. 1). The majority of methods [11, 25, 34, 45, 46, 49, 51, 52] assumes the imagery to be captured under one fixed lighting condition (rows 1, 2). However, in such scenarios, only one case of scene appearances among all possible cases under different lighting is observed, causing severe **lighting-material ambiguity**, posing considerable difficulties to material estimation. In a bid to alleviate this ambiguity, recent techniques [2, 6, 12, 14, 28, 42] utilize in-

#Corresponding author. E-mail: shiboxin@pku.edu.cn.

Table 1. A summary of multi-view inverse rendering methods based on their assumptions and usages of input lighting conditions.

Methods	# of far-field light	# of near-field light	Material optimization	Lighting-material ambiguity	Capture workload
(1) [34, 49, 51, 52]	Single	None	Only from single	Strong	Low
(2) NeILF [45]	Single 5D incident light field				
(3) [2, 12, 14, 42] [a]	Multiple	None	Only from far-field	Moderate	High
(4) WildLight [6]	Single	Single	Only from near-field	Moderate	Low
(5) Ours	Single	Single	From near- and far-field	Weak	Low
	Multiple	Single or multiple		Almost none	High

[a] These methods also support input images under single far-field lighting. In that case, they belong to single far-field lighting methods (row 1).

put images under varied lighting to discern materials from lighting (rows 3, 4). This demands more complex lighting models. Existing varied-lighting methods [2, 12, 14, 28, 42] typically model multiple, but only far-field, lighting conditions to maintain manageable unknown lighting parameters. This means throughout the capture, the lighting setups should be largely changed by either moving the objects to reconstruct or adapting far-field light sources, causing **increased workload** to the data collection process. Alternatively, near-field light sources, such as flashlights, can function as easily controllable light sources for disambiguation. WildLight [6] suggests the use of a camera-collocated flashlight in addition to the ambient lighting for inverse rendering. Nevertheless, it limits itself by using only the isolated appearance under the flashlight for material estimation, **not harnessing available observations** under the ambient lighting and degrading the quality of its estimated material (row 4).

Addressing the challenges and restrictions previously outlined, this paper introduces VMINer, a **V**ersatile **M**ulti-view **I**nverse rendering framework with **Ne**a**r**- and far-field light sources (row 5). The distinctive feature of our framework is that it uses a lightweight yet inclusive lighting model for different far- and near-field light sources and efficiently leverages available observations under each lighting to disentangle lighting from other components, **making the most of whatever lighting conditions are at hand** to boost the practicality and quality of the reconstruction, as shown in Fig. 1. Although our method accepts the simplest setting of inputs under single far-field lighting, a more effective compromise is to employ an additional flashlight to enhance quality without overly complicating the capture process. The ideal scenario for the highest quality reconstruction would involve capturing the scene under varied far-field lighting and using a flashlight.

Nonetheless, implementing such a versatile framework is far from straightforward. The method has to render the scene under different lighting conditions, including spatially-varying and changing ones to represent possibly moving near-field lights. We propose to solve the problem by modeling **near-field lights as point sources** in 3D space with adjustable positions and intensities, which could be **anisotropic and not necessarily camera-aligned** (*e.g.*, stationary desk lamps). This representation effectively approximates a broad range of common near-field lights and is easy for spatially-consistent editing and optimization. We use 3D implicit fields to model the scenes' radiance, shape, and material and minimize the error of the neural or physically-based re-rendered scene appearance under each input lighting. To account for multiple lighting situations and moving near-field lights, we integrate specially tailored multi-layer perceptrons (MLPs) that additionally process light directions and embeddings. Post-training, the fields are converted into a textured triangle mesh, ready for **seamless integration** into industrial rendering software like Blender [1], facilitating a broad spectrum of applications. Our extensive experiments, covering both synthetic and real-world scenes, demonstrate that VMINer surpasses prior methods quantitatively and qualitatively.

2. Related Work

Preliminary. The rendering equation [13] models the reflected radiance from any point on a surface as a result of an intricate surface integral. This integral comprises contributions from three fundamental scene components: lighting, material, and geometry. The equation is expressed as:

$$L_{\mathrm{o}}(\mathbf{x}, \boldsymbol{\omega}_{\mathrm{o}}) = \int_{\Omega} \underbrace{L_{\mathrm{i}}(\mathbf{x}, \boldsymbol{\omega}_{\mathrm{i}})}_{\text{lighting}} \underbrace{f(\mathbf{x}, \boldsymbol{\omega}_{\mathrm{i}}, \boldsymbol{\omega}_{\mathrm{o}})}_{\text{material}} \underbrace{(\boldsymbol{\omega}_{\mathrm{i}} \cdot \mathbf{n})^{+}}_{\text{geometry}} \mathrm{d}\,\boldsymbol{\omega}_{\mathrm{i}}. \quad (1)$$

In this equation, $L_{\mathrm{o}}(\mathbf{x}, \boldsymbol{\omega}_{\mathrm{o}})$ represents the radiance reflected in the outgoing direction $\boldsymbol{\omega}_{\mathrm{o}}$ from a surface point $\mathbf{x}$ in 3D space. Ω denotes the unit hemisphere encompassing all incident directions $\boldsymbol{\omega}_{\mathrm{i}}$ with $\boldsymbol{\omega}_{\mathrm{i}} \cdot \mathbf{n} > 0$. $L_{\mathrm{i}}(\mathbf{x}, \boldsymbol{\omega}_{\mathrm{i}})$ is the incident radiance at $\mathbf{x}$ from direction $\boldsymbol{\omega}_{\mathrm{i}}$, and $f(\mathbf{x}, \boldsymbol{\omega}_{\mathrm{i}}, \boldsymbol{\omega}_{\mathrm{o}})$ is the bidirectional reflectance distribution function (BRDF), denoting the ratio of light reflected along $\boldsymbol{\omega}_{\mathrm{o}}$ at $\mathbf{x}$ from $\boldsymbol{\omega}_{\mathrm{i}}$. The term $(\boldsymbol{\omega}_{\mathrm{i}} \cdot \mathbf{n})^{+}$, where $x^{+} \triangleq \max(x, 0)$, signifies the cosine weakening factor. The incident lighting $L_{\mathrm{i}}(\mathbf{x}, \boldsymbol{\omega}_{\mathrm{i}})$ may originate from other points within the scene, necessitating recursive evaluation of this integral.

Multi-view Inverse Rendering. Multi-view inverse rendering techniques [2–4, 6, 11, 12, 14, 25, 28, 34, 44–46, 48–53] generally employ implicit or explicit 3D fields to model

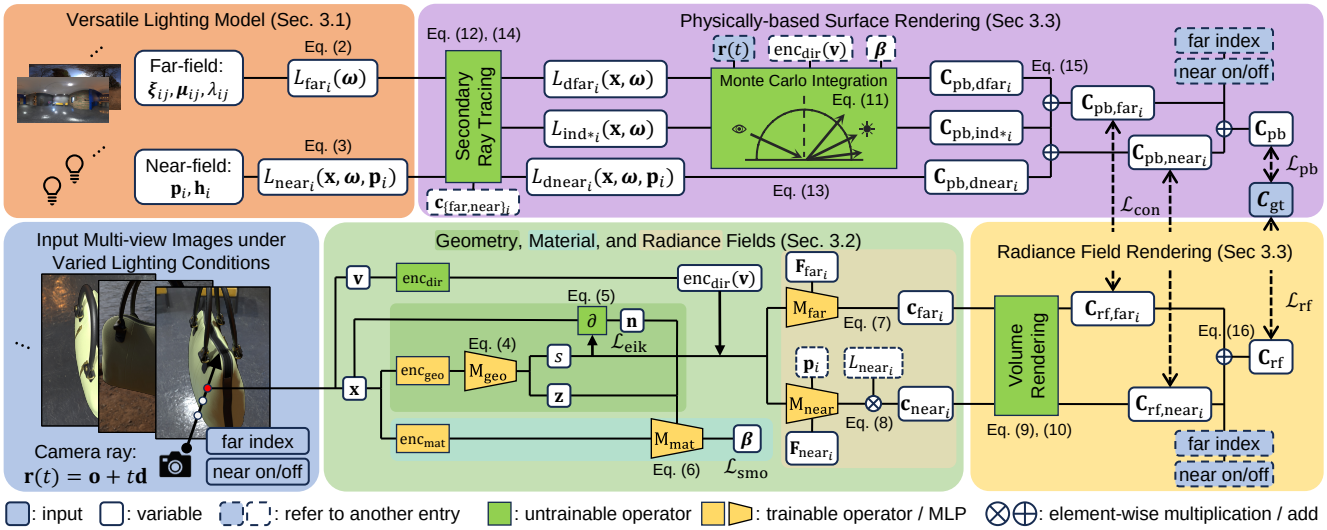

Figure 2. **Bottom-left**: VMINer takes as input multi-view RGB images with foreground masks under possibly varied far- and near-field lighting. The total number of lighting conditions and their types should be given. Each image is illuminated by one far-field lighting whose index is "far index", and near-field lights whose on/off states are "near on/off". **Top-left**: VMINer models each far-field lighting (far_i) and near-field lighting (near_i) separately as parametric models, from which incoming radiance is queried at any position $\mathbf{x}$ and direction $\boldsymbol{\omega}$. **Bottom-mid**: It utilizes 3D implicit representations for scene geometry, material, and radiance. Four MLPs process $\mathbf{x}$ and view direction $\mathbf{v}$ encoded by different encoders $\text{enc}_{\{\text{geo,mat,dir}\}}$ with light embeddings $\mathbf{F}_{\{\text{far,near}\}_i}$ to get the SDF s, the neural appearance descriptor $\mathbf{z}$, the BRDF parameters $\boldsymbol{\beta}$, and the radiance $\mathbf{c}_{\{\text{far,near}\}_i}$ under each lighting. **Bottom-right**: It then uses radiance field rendering to re-render the appearance $\mathbf{C}_{\text{rf},\{\text{near,far}\}_\text{i}}$ separately under each lighting condition, added up to $\mathbf{C}_\text{rf}$ according to the per-image lighting condition. **Top-right**: Physically-based surface rendering uses Monte Carlo integration to evaluate both direct illumination $L_{\text{d}\{\text{near,far}\}_\text{i}}$ with secondary visibility and indirect illumination $L_{\text{ind}\{\text{near,far}\}_\text{i}}$. The rendered appearances are aggregated to $\mathbf{C}_\text{pb}$. $\mathbf{C}_{\{\text{rf,pb}\}}$ are compared with the observation $\mathbf{C}_\text{gt}$ to train the scene model. $\mathbf{C}_{\text{rf},\{\text{near,far}\}_\text{i}}$ are also used as additional supervision signals for $\mathbf{C}_{\text{pb},\{\text{near,far}\}_\text{i}}$.

the scene's geometry and material, while also estimating lighting to recreate the scene's appearance. They optimize each scene component from scratch by aligning the rendered with the observed appearance, while avoiding potential local minima, which, although do not faithfully represent the scene, can replicate the observed input. Addressing this inherently ambiguous nature of inverse rendering, these methods typically impose constraints on each scene component. For geometry and material, assumptions of known geometry [45, 46, 56] are made, along with leveraging the recovered shape from radiance fields [6, 12, 51], adopting Lambertian or uniform materials [28, 49], controlling BRDF parameter smoothness [12], or integrating a learned BRDF latent space [51]. Regarding lighting, many approaches presuppose known lighting conditions [34, 50] or fixed lighting shared across input images [45, 48, 51, 52]. More recent techniques [2, 6, 12, 14, 28, 42] have turned to using images with varied lighting conditions to mitigate the lighting-material ambiguity. They typically accept input images that can be grouped into sets, each illuminated by one distinct and fixed far-field lighting. WildLight [6] exceptionally employs a camera-collocated flashlight along with fixed ambient lighting. However, it does not model the ambient lighting and thus its material property estimation relies exclusively on flashlight observations, where the light and view direction always coincide [5], limiting its BRDF estimation ability. Our VMINer, in contrast, accommodates input images under single or multiple far-field lighting conditions with near-field light sources and leverages each of them to enhance the quality of the estimated material.

Lighting Models. Spatially-uniform (SU) lighting models rely on the assumption that the scene's lighting originates from a distant source, modeled as spherical Gaussians (SGs) [2, 12], spherical harmonics (SH) [14, 28], or MLPs [42]. In contrast, spatially-varying (SV) lighting models acknowledge the presence of near-field light sources, causing different locations to receive different incident lighting. They include parametric 3D lights [8, 21, 46, 47], outgoing light fields such as volumetric SGs [41] and neural out-of-view lighting volumes [55], and incident light fields like SV environment maps [57], SVSGs [20], and neural incident light fields (NeILF) [45, 48]. Our VMINer integrates distant lights as SGs and near-field lights as point lights with adjustable position and intensity. It is less complex compared to other SV lighting models, facilitating spatially consistent editing and optimization, while being able to approximate common lighting setups in real life.

3. Proposed Method

As illustrated in Fig. 2, VMINer reconstructs the scene lighting (Sec. 3.1) and 3D fields (Sec. 3.2) from input images using differentiable rendering (Sec. 3.3). The training

scheme and loss functions are described in Sec. 3.4.

3.1. Versatile Lighting Model

This subsection models the lighting term in Eq. (1) and corresponds to the top-left part of Fig. 2.

VMINer harnesses the diversity of lighting conditions present in the input RGB images to disentangle lighting effects. The lighting model plays a crucial role here: 1) Since lighting conditions are unknown initially, they must be optimized alongside other scene parameters using physically-based rendering during training. 2) Different lighting setups, each compatible with the model, are necessary during the capture phase to achieve this goal. Given these prerequisites, it is essential to design a lighting model that accommodates common far-field and near-field light sources, while being lightweight enough for efficient and robust optimization and physically-based rendering. With these considerations, we propose a versatile lighting model that is both inclusive and lightweight. This representation models far-field and near-field lighting separately as below.

Far-field Lighting. In our model, each far-field lighting condition i, where i belongs to the set $\mathbb{N}_{N_{\text{far}}} \triangleq \{1, \dots, N_{\text{far}}\}$ and N_{far} denotes the number of far-field lighting, is represented using 128-lobe SGs [38] commonly adopted in existing far-field lighting techniques [43, 49, 52]. For each lobe j of lighting i, there are six parameters: lobe axis $\boldsymbol{\xi}_{ij} \in \mathbb{S}^2$, lobe sharpness $\lambda_{ij} \in \mathbb{R}_+$, and lobe RGB amplitude $\boldsymbol{\mu}_{ij} \in \mathbb{R}^3_+$. The incident radiance from far-field lighting i along direction $\boldsymbol{\omega}$ at any 3D position $\mathbf{x}$ is calculated using the formula (ignoring visibility for now):

$$L_{\text{far}_i}(\boldsymbol{\omega}) = \sum_{j=1}^{128} c_{ij} \boldsymbol{\mu}_{ij} e^{\lambda_i (\boldsymbol{\omega} \cdot \boldsymbol{\xi}_{ij} - 1)}, \tag{2}$$

where $c_{ij} = \lambda_{ij} / (2\pi(1 - e^{-2\lambda_{ij}}))$ acts as the normalization factor related to roughness. Notably, $\mathbf{x}$ does not appear as an input to L_{far_i}, reflecting the nature of distant lighting.

Near-field Lighting. Each near-field lighting i, included in the set $\mathbb{N}_{N_{\text{near}}}$ with N_{near} representing the number of near-field light sources, is modeled as a point light. These point lights can have moving positions $\mathbf{p}_i \in \mathbb{R}^3$ and exhibit anisotropic radiation characterized by l^{th}-order (l can be $0, 1$, or 2) SH coefficients $\mathbf{h}_i \in \mathbb{R}^{3 \times (l+1)^2}$. Our method accommodates two types of near-field lighting: camera-collocated lights, positioned at the camera ray origin for each image, and stationary lights, which remain fixed across all images. We observe that in neural radiance field rendering, the radiance under a stationary near-field light, especially when the light is active in all images, can be challenging to distinguish from radiance under far-field lighting. Also, the radiance from these two sources aids in material estimation in a similar way. Therefore, in practical applications, we favor using a moving flashlight as the near-field light to provide unique information for material estimation. The incident radiance from near-field lighting i at a 3D position $\mathbf{x}$ is computed as (also ignoring visibility for now):

$$L_{\text{near}_i}(\mathbf{x}, \boldsymbol{\omega}, \mathbf{p}_i) = \begin{cases} \frac{\text{SH}(\boldsymbol{\omega}; \mathbf{h}_i)}{\|\mathbf{p}_i - \mathbf{x}\|_2^2} & \text{if } \boldsymbol{\omega} = \frac{\mathbf{p}_i - \mathbf{x}}{\|\mathbf{p}_i - \mathbf{x}\|_2} \\ 0 & \text{otherwise} \end{cases}, \tag{3}$$

where $\text{SH}(\boldsymbol{\omega}; \mathbf{h})$ calculates the SH at direction $\boldsymbol{\omega}$ with coefficients $\mathbf{h}$, and $1/\|\mathbf{p}_i - \mathbf{x}\|_2^2$ signifies the inverse-square lighting attenuation for point lights. Here we include $\mathbf{p}_i$ as an input because it may be set differently across images. It is important to note that with respect to the incident direction $\boldsymbol{\omega}$, L_{far} is a continuous function, while L_{near} is a discrete function, being non-zero only in a single direction. As a result, to render appearances under direct lighting, Monte Carlo integration is essential for far-field lighting, while a simple multiplication suffices for near-field lighting. Further details about this process are provided in Sec. 3.3.

3.2. Geometry, Material, and Radiance Fields

This subsection models the material and geometry terms in Eq. (1) and corresponds to the bottom-mid part of Fig. 2.

Geometry. Multi-view reconstruction methodologies generally hinge on two geometry representations: volume density [2, 12, 23, 51] and the signed distance function (SDF) [43, 49, 52, 54] fields. We choose the SDF field for our geometry representation due to its clearly defined surface at the zero-level isosurface, which simplifies and enhances the post-training mesh extraction process. Our approach utilizes implicit fields as MLPs over explicit structures like voxel grids for their compactness and flexibility. For each 3D position $\mathbf{x}$, a multi-resolution hash grid [24] is first employed for positional encoding, yielding a feature vector $\text{enc}_{\text{geo}}(\mathbf{x}) \in \mathbb{R}^{16}$. The geometry MLP M_{geo} then predicts the SDF value at $\mathbf{x}$:

$$s(\mathbf{x}), \mathbf{z}(\mathbf{x}) = \text{M}_{\text{geo}}(\mathbf{x}, \text{enc}_{\text{geo}}(\mathbf{x})), \tag{4}$$

where $s \in \mathbb{R}$ represents the signed distance (positive outside the surface, negative inside), and $\mathbf{z} \in \mathbb{R}^{13}$ is a descriptor of local appearance. To maintain differentiability in rendering and to aid in geometry optimization, during training we apply NeuS [39] techniques to transform SDF values along rays into volume densities. The surface normal $\mathbf{n} \in \mathbb{S}^2$ is derived as the gradient of the SDF s with respect to $\mathbf{x}$:

$$\mathbf{n}(\mathbf{x}) = \frac{\partial s}{\partial \mathbf{x}} \Big/ \left\| \frac{\partial s}{\partial \mathbf{x}} \right\|_2. \tag{5}$$

M_{geo} only approximates a strict SDF field that has $\left\| \frac{\partial s}{\partial \mathbf{x}} \right\|_2 = 1$, so we normalize $\mathbf{n}$ to ensure it is a unit vector.

Material. For accurate scene material recovery using physically-based re-rendering, it is imperative to model the material in such a way that the renderer can query the BRDF at any point on the surface. Considering that material properties are independent of lighting and view directions, we

represent the scene's material through an implicit 3D field, much like the SDF field utilized for geometry. To achieve this, we employ another multi-resolution hash grid $\mathrm{enc}_{\mathrm{mat}}$ for positional encoding. We adopt the simplified GGX BRDF model [37] with a fixed fresnel parameter to help alleviate ambiguity. The material MLP, $\mathrm{M}_{\mathrm{mat}}$, is tasked with predicting the SVBRDF parameters $\boldsymbol{\beta}(\mathbf{x})$:

$$\boldsymbol{\beta}(\mathbf{x}) = \mathrm{M}_{\mathrm{mat}}(\mathbf{x}, \mathbf{z}, \mathbf{n}, \mathrm{enc}_{\mathrm{mat}}(\mathbf{x})). \tag{6}$$

Radiance. Although our method does not strictly require a radiance field for rendering scene appearance – given that appearance could be rendered solely using physically-based surface rendering (PBR) – we find that incorporating a radiance field at the start of training significantly eases the optimization of scene geometry and appearance. PBR, while physically accurate, tends to introduce instability and slow down the training process due to its inherent ambiguity and computational intensity. Moreover, the radiance field-rendered results can serve as supplementary supervision, aiding PBR in better separating scene appearance under each lighting condition and thus further diminishing lighting-material ambiguity. Hence, we deploy radiance MLPs to predict view-dependent and lighting-dependent radiance at each position. For each lighting condition i, a light embedding $\mathbf{F}_{\{\mathrm{far,near}\}_i} \in \mathbb{R}^{16}$ is utilized. The view direction $\mathbf{v} \in \mathbb{S}^2$ is processed through directional encoding that projects it onto the coefficients of the 3^{rd}-order SH basis, yielding $\mathrm{enc}_{\mathrm{dir}}(\mathbf{v}) \in \mathbb{R}^{4^2=16}$ [24]. The far-field radiance MLP $\mathrm{M}_{\mathrm{far}}$ computes the outgoing radiance $\mathbf{c}_{\mathrm{far}_i}$ from position $\mathbf{x}$ along view direction $\mathbf{v}$ under far-field lighting i:

$$\mathbf{c}_{\mathrm{far}_i}(\mathbf{x}, \mathbf{v}) = \mathrm{M}_{\mathrm{far}}(\mathbf{x}, \mathbf{z}, \mathbf{n}, \mathrm{enc}_{\mathrm{dir}}(\mathbf{v}), \mathbf{F}_{\mathrm{far}_i}), \tag{7}$$

where we incorporate the surface normal $\mathbf{n}$ as an additional MLP input, following Instant-NSR [54] and WildLight [6], as this has shown to aid shape recovery.

The near-field radiance MLP differs slightly, as near-field lighting can move in different images, thus the current light position $\mathbf{p}_i$ is a necessary input. Additionally, the radiance under near-field point light takes the form of the rendering equation Eq. (1) without the integral, allowing the incoming radiance and cosine term to be explicitly included. We follow ReNe [35] to input the relative direction of the point light $\boldsymbol{\omega}_i = \frac{\mathbf{p}_i - \mathbf{x}}{\|\mathbf{p}_i - \mathbf{x}\|_2}$ instead of the absolute position $\mathbf{p}_i$ into the MLP. Therefore, the outgoing neural radiance $\mathbf{c}_{\mathrm{near}_i}$ under near-field lighting i is calculated as:

$$\begin{aligned}\mathbf{c}_{\mathrm{near}_i}(\mathbf{x}, \mathbf{v}, \mathbf{p}_i) = \mathrm{M}_{\mathrm{near}}\,(\mathbf{x}, \mathbf{z}, \mathbf{n}, \mathrm{enc}_{\mathrm{dir}}(\mathbf{v}), \mathbf{F}_{\mathrm{near}_i}, \boldsymbol{\omega}_i) \\ \otimes \frac{L_{\mathrm{near}_i}(\mathbf{x}, \boldsymbol{\omega}_i)}{\|\mathbf{p}_i - \mathbf{x}\|_2^2} (\boldsymbol{\omega}_i \cdot \mathbf{n})^+,\end{aligned} \tag{8}$$

where $\otimes$ denotes element-wise multiplication. On the right side, the second term denotes the incident radiance attenuated by square distance, the third term denotes the cosine weakening factor, and $\mathrm{M}_{\mathrm{near}}$ accounts for other factors like reflectance and visibility. Owing to the instability of these terms at the training's outset, we use an annealing strategy, as recommended in prior works [54], gradually replacing default values (*e.g.*, 1 for incident radiance) with those optimized during training, to ease through the process.

3.3. Differentiable Rendering

To simplify the notation, we omit $\mathbf{p}_i$ and the lighting index i in the equations within this subsection. Our approach optimizes the lighting alongside the implicit fields primarily by reducing the discrepancy between the observed and differentiably re-rendered appearances. The re-rendering is accomplished using two types of renderer: a neural radiance field renderer (Fig. 2 bottom-right) and a physically-based surface renderer (Fig. 2 top-right; Eq. (1)). We handle each lighting condition separately in rendering, and then aggregate the results based on per-image lighting condition.

Volume Rendering for SDF Field. Consider a camera ray $\mathbf{r}(t) = \mathbf{o} + t\mathbf{d}$ $(t > 0)$, with $\mathbf{o} \in \mathbb{R}^3$ as its origin and $\mathbf{d} \in \mathbb{S}^2$ as its direction. We incorporate importance sampling using an occupancy grid [18] to get $N \leq 1024$ points along this ray, denoted as $\mathbf{r}(t_i), i \in \mathbb{N}_N$. Utilizing techniques from NeuS [39], we transform the signed distances $s(\mathbf{r}(t_i))$ into discrete opacity values as follows:

$$\alpha_i = \left(\frac{\Phi_b(s(\mathbf{r}(t_i))) - \Phi_b(s(\mathbf{r}(t_{i+1})))}{\Phi_b(s(\mathbf{r}(t_i)))} \right)^+, \tag{9}$$

where $\Phi_b(s) = 1/(1 + e^{-bs})$ is the cumulative opacity distribution function. The parameter b is trainable and tends to increase during training, focusing opacity more narrowly around the surface where $s = 0$. The accumulated transmittance is then calculated as $T_i = \prod_{j=1}^{i-1}(1 - \alpha_j)$.

Neural Radiance Field Rendering. The color $\mathbf{C}_{\mathrm{rf}}$ of a camera ray under either far- or near-field lighting conditions, as rendered through the neural radiance field, is determined by accumulating the radiance along the ray's path:

$$\mathbf{C}_{\mathrm{rf},\{\mathrm{far,near}\}}(\mathbf{o}, \mathbf{d}) = \sum_{i=1}^{N} T_i \alpha_i \mathbf{c}_{\{\mathrm{far,near}\}}(\mathbf{r}(t_i), -\mathbf{d}). \tag{10}$$

This formula sums the radiance from each sampled point to yield the color of the ray under the specified lighting.

Physically-based Rendering. Our method employs differentiable surface rendering, modeling the observed radiance as reflected from a single surface point. We compute the depth t of this surface point similarly to Eq. (10): $t = \sum_{i=1}^{N} T_i \alpha_i t_i$. Accumulation along the ray also applies to the BRDF parameters $\boldsymbol{\beta}$ and the surface normal $\mathbf{n}$. For each input lighting, we render the appearance that models secondary visibility and indirect illumination. The rendering method under direct illumination differs for far- and near-field lighting. For direct far-field lighting L_{dfar}, we employ Monte Carlo integration to evaluate the rendering equation (Eq. (1)) as the incident light comes from all

directions:

$$\mathbf{C}_{\text{pb,dfar}}(\mathbf{o},\mathbf{d}) = L_{\text{o}}(\mathbf{r}(t),-\mathbf{d}) = \frac{1}{S}\sum_{s=1}^{S}\frac{L_{\text{dfar}}(\mathbf{r}(t),\boldsymbol{\omega}_s)f(\mathbf{r}(t),\boldsymbol{\omega}_s,-\mathbf{d})(\boldsymbol{\omega}_s\cdot\mathbf{n})^{+}}{p(\boldsymbol{\omega}_s)}, \quad (11)$$

where S is the number of sampled directions, $\boldsymbol{\omega}_s$ a sampled incident direction, and $p(\boldsymbol{\omega}_s)$ the probability density function (PDF) for the sampled direction. We use multiple importance sampling [36] to combine evaluations from different strategies: BRDF importance sampling [15], SG lighting importance sampling [43], and cosine importance sampling. we follow differentiable rendering works [19] to use a small $S = 20$, as stochastic gradient descent handles noisy gradients. The incident radiance, now taking the visibility of light sources into account, is given by:

$$L_{\text{dfar}}(\mathbf{r}(t),\boldsymbol{\omega}_s) = L_{\text{far}}(\boldsymbol{\omega}_s)\, V(\mathbf{r}(t),\boldsymbol{\omega}_s), \quad (12)$$

with $V(\mathbf{r}(t),\boldsymbol{\omega}_s) \in [0,1]$ representing visibility, or the inverse of the accumulated opacity along a secondary ray $\mathbf{r}_{\text{sec}}(t') = \mathbf{r}(t) + t'\boldsymbol{\omega}_s,\ (t' > 0)$, computed in a manner akin to Eq. (10). For direct near-field lighting L_{dnear}, the integral in Eq. (1) simplifies to a multiplication, as the light originates from a single direction:

$$\mathbf{C}_{\text{pb,dnear}}(\mathbf{o},\mathbf{d}) = L_{\text{dnear}}(\mathbf{r}(t),\boldsymbol{\omega})f(\mathbf{r}(t),\boldsymbol{\omega},-\mathbf{d})(\boldsymbol{\omega}\cdot\mathbf{n})^{+}, \quad (13)$$

where $\boldsymbol{\omega} = (\mathbf{p}-\mathbf{r}(t))/\left\|\mathbf{p}-\mathbf{r}(t)\right\|_2$ is the direction of incident light, and $L_{\text{dnear}}(\mathbf{r}(t),\boldsymbol{\omega})$ is the potentially occluded incident radiance, computed similarly to Eq. (12).

Indirect illumination L_{ind}, relevant to both far-field and near-field lighting, considers radiance reflected from the scene itself, potentially from all directions, thus necessitating Monte Carlo integration. The neural radiance field $\mathbf{C}_{\text{rf}}$ substitutes multi-bounce path tracing for indirect illumination, given its encapsulation of the scene radiance under infinite lighting bounces. The radiance $\mathbf{C}_{\text{pb,ind}\{\text{far,near}\}}$ from indirect illumination is evaluated similarly to Eq. (12), but with the L_{dfar} replaced by the indirect lighting $L_{\text{ind}\{\text{far,near}\}}$:

$$L_{\text{ind}*}(\mathbf{r}(t),\boldsymbol{\omega}_s) = \mathbf{c}_*(\mathbf{x}',-\boldsymbol{\omega}_s)(1-V(\mathbf{r}(t),\boldsymbol{\omega}_s)), \quad (14)$$

where $\mathbf{x}'$ is the intersection of the secondary ray $\mathbf{r}_{\text{sec}}(t')$ with the scene geometry. The radiance from indirect illumination is also computed for each lighting condition separately, then combined with direct illumination to yield the complete PBR radiance for the camera ray under a specific lighting condition:

$$\mathbf{C}_{\text{pb,}\{\text{far,near}\}} = \mathbf{C}_{\text{pb,d}\{\text{far,near}\}} + \mathbf{C}_{\text{pb,ind}\{\text{far,near}\}}. \quad (15)$$

3.4. Training

Training Schemes. Our training process is divided into two sequential stages. In the first stage, the focus is on training the geometry and radiance field without employing PBR. The objectives are: 1) to recover the scene's geometry, 2) to distinguish the appearance under each lighting condition, a necessity due to the potential presence of multiple light sources in one image, and 3) to utilize the reconstructed radiance field for indirect illumination. In the second stage, we train the material field and scene lighting using PBR, aiming to estimate material properties.

Loss Functions. We average the loss functions across batches of camera rays. During the first stage, our loss function compares the total re-rendered radiance under all lighting conditions present in an image – including a far-field light i and any active near-field lights – with the ground-truth (GT) color $\mathbf{C}_{\text{gt}}$ from the input observation. This comparison trains the geometry and radiance field:

$$\mathcal{L}_{\text{rf}} = \|\mathbf{C}_{\text{rf,far}_i} + \sum_{\text{near } i \text{ is on}} \mathbf{C}_{\text{rf,near}_i} - \mathbf{C}_{\text{gt}}\|_2^2. \quad (16)$$

The Eikonal loss [10] is applied to the gradients of the SDF values s for geometric regularization:

$$\mathcal{L}_{\text{eik}} = \sum_{j=1}^{N} T_j\alpha_j \left(\left\|\frac{\partial s(\mathbf{r}(t_j))}{\partial \mathbf{r}(t_j)}\right\|_2 - 1\right)^2, \quad (17)$$

A silhouette loss using GT alpha α_{gt} from the foreground mask and a normal smoothness loss [51] aid shape recovery:

$$\mathcal{L}_{\text{sil}} = ((1-T_{N+1}) - \alpha_{\text{gt}})^2. \quad (18)$$

$$\mathcal{L}_{\text{ns}} = \sum_{j=1}^{N} T_j\alpha_j \left(\mathbf{n}(\mathbf{r}(t_j)) - \mathbf{n}(\mathbf{r}(t_j)+\boldsymbol{\epsilon})\right)^2, \quad (19)$$

where $\boldsymbol{\epsilon} \in \mathbb{R}^3$ is a small random perturbation. The total loss for stage one is $\mathcal{L}_{\text{rf}} + \lambda_{\text{eik}}\mathcal{L}_{\text{eik}} + \lambda_{\text{sil}}\mathcal{L}_{\text{sil}} + \lambda_{\text{ns}}\mathcal{L}_{\text{ns}}$, with λ_* representing the loss weights. In stage two, PBR colors replace radiance field-rendered colors for comparison with $\mathbf{C}_{\text{gt}}$, and a novel self-consistency loss is introduced between neural and PBR radiance under each lighting for additional supervision. The direct supervision loss $\mathcal{L}_{\text{pb}}$ mirrors $\mathcal{L}_{\text{rf}}$, with $\mathbf{C}_{\text{rf,}\{\text{far,near}\}_i}$ replaced by $\mathbf{C}_{\text{pb,}\{\text{far,near}\}_i}$. The per-lighting self-consistency loss $\mathcal{L}_{\text{con}}$ is defined as:

$$\mathcal{L}_{\text{con}} = \sum_{i=1}^{N_{\{\text{far,near}\}}} \left\|\mathbf{C}_{\text{rf,}\{\text{far,near}\}_i} - \mathbf{C}_{\text{pb,}\{\text{far,near}\}_i}\right\|_2^2. \quad (20)$$

This loss is crucial for discerning contributions from each lighting in images with multiple active light sources, thereby reducing ambiguity and improving material estimation. A material smoothness loss $\mathcal{L}_{\text{ms}}$ is used similar to $\mathcal{L}_{\text{ns}}$. The total loss for stage two is $\mathcal{L}_{\text{pb}} + \lambda_{\text{con}}\mathcal{L}_{\text{con}} + \lambda_{\text{ms}}\mathcal{L}_{\text{ms}}$.

Our model is trained on an NVIDIA RTX 3090 GPU for a total of 40,000 steps. The first 30,000 steps of stage one take about 20 minutes, while the subsequent 10,000 steps of stage two require about 40 minutes. Post-training, we follow a procedure similar to WildLight [6] to extract the fields into textured meshes, which can be easily integrated into industry-standard rendering software like Blender [1],

Table 2. Quantitative comparison results with state-of-the-art methods averaged on 6 synthetic scenes. We show results of surface normal, diffuse albedo, view synthesis RGB, free-viewpoint (FV) relit RGB, the specular reflection part of FV relit RGB, and training time on a single RTX 3090 GPU. We mark the **best** and the second best results in each column. ↑ (↓) means bigger (smaller) is better.

Method	Input lighting conditions	Normal	Albedo		View synthesis		FV relit		FV relit (spec)		Time
		MAngE↓	PSNR↑	SSIM↑	PSNR↑	SSIM↑	PSNR↑	SSIM↑	PSNR↑	SSIM↑	
(1) TensoIR [12]	Single far-field	17.66	26.48	0.921	29.48	0.912	28.18	0.901	28.30	0.861	300 mins
(2) NVDiffRecMC [11]		16.24	26.52	0.915	27.13	0.913	26.61	0.901	25.57	0.836	150 mins
(3) Ours		12.39	24.50	0.882	28.20	0.934	27.46	0.921	27.56	0.871	45 mins
(4) WildLight [6]	Single far-field + Flashlight	11.49	28.86	0.940	29.87	0.929	30.44	0.930	27.71	0.863	1440 mins
(5) Ours		10.89	31.62	0.953	32.09	**0.953**	32.00	0.953	30.75	0.906	60 mins
(6) TensoIR [12]	Two far-field	16.24	27.18	0.929	29.68	0.912	28.66	0.902	28.46	0.863	300 mins
(7) Ours		11.70	26.07	0.902	29.59	0.942	29.27	0.934	29.09	0.890	45 mins
(8) Ours	Two far + Single near	**10.79**	**32.04**	**0.957**	**32.10**	0.950	**32.38**	**0.954**	**31.40**	**0.910**	60 mins

enabling fast and high-quality rendering suitable for various applications. Check the supplement for more details.

4. Experiments

Our experiments involve a comprehensive comparison with state-of-the-art methods (Sec. 4.2), ablation studies (Sec. 4.3), and evaluations on both synthetic datasets (Sec. 4.1) and real-world images (Sec. 4.4). The supplement shows further data creation details and more experiments.

4.1. Synthetic Datasets

We collect 6 synthetic scenes comprising a variety of scene geometries and materials. For training, each scene is rendered under four distinct lighting setups: 1) single far-field light, 2) single far-field light with a camera-collocated flashlight, 3) two far-field lights, and 4) two far-field lights with a near-field light source. We render 100 training images per setting using random viewpoints and per-image lighting conditions. The far-field lighting from the first setting is used in subsequent settings, with the last two sharing their far-field lighting. For testing, 200 images under one common far-field lighting are rendered to assess novel view synthesis under seen lighting, and another 200 under unseen far-field lighting to evaluate free-viewpoint (FV) relighting, which largely depends on the geometry and material estimation quality. This methodology provides a relatively fair comparison across methods with different abilities regarding supported input lighting conditions, trained under similar conditions and tested all using the same sets.

4.2. Comparison with State-of-the-art Methods

We compare our method with leading multi-view inverse rendering methods under each input lighting scenario. For single far-field lighting inputs (Tab. 1 row 1), we compare with NVDiffRecMC [11] and TensoIR [12] (single-lighting input). For multiple far-field lighting scenarios (Tab. 1 row 3), we compare with TensoIR [12]. For setups involving a far-field light and a flashlight (Tab. 1 row 4), we compare with WildLight [6]. We use Blender [1] for high-quality relit results for our method, WildLight [6], and NVDiffRecMC [11], importing trained SDF and material fields as textured meshes. Since TensoIR [12] employs density-based geometry and cannot extract high-quality meshes, we utilize its differentiable renderer for relighting.

Our quantitative comparison employs mean angular error (MAngE), peak signal-to-noise ratio (PSNR), and structural similarity (SSIM) [40] metrics for surface normal, diffuse albedo, view synthesis RGB, free-viewpoint (FV) relit RGB, and specular reflection of FV relit RGB. We follow NeRFactor [51] in assuming albedo and lighting brightness as scale-invariant, necessitating metric comparisons using corresponding scales. For visualization and metrics, we scale each RGB channel of albedo and relit images from all methods by a global scale to minimize mean squared error against ground truth.

The quantitative results are detailed in Tab. 2. We observe that when only input images under single far-field lighting are available (rows 1-3), our method gives comparable results as TensoIR [12] and NVDiffRecMC [11]. Adding a flashlight (rows 4, 5) significantly helps in geometry and material estimation (comparing rows 3, 5), where the greatest improvement is made regarding the diffuse albedo. Also, our method clearly surpasses WildLight [6] in both quality and speed in this setting. Leveraging another far-field lighting condition (rows 6, 7) brings similar effects as using a flashlight, but generally with a lower degree of improvement. Further adding a flashlight (row 8) gives the best results among all settings and methods, but the gain is marginal compared to the setting of single far-field lighting with a flashlight (comparing rows 5, 8).

Fig. 3 shows the qualitative comparison on two synthetic scenes: LEGO (left) with detailed geometry, and TROOPER (right) with highly-reflective SV materials. The results show that our method predicts more detailed and faithful shape, diffuse albedo, and specular parameters that could produce accurate and realistic relit results under unseen

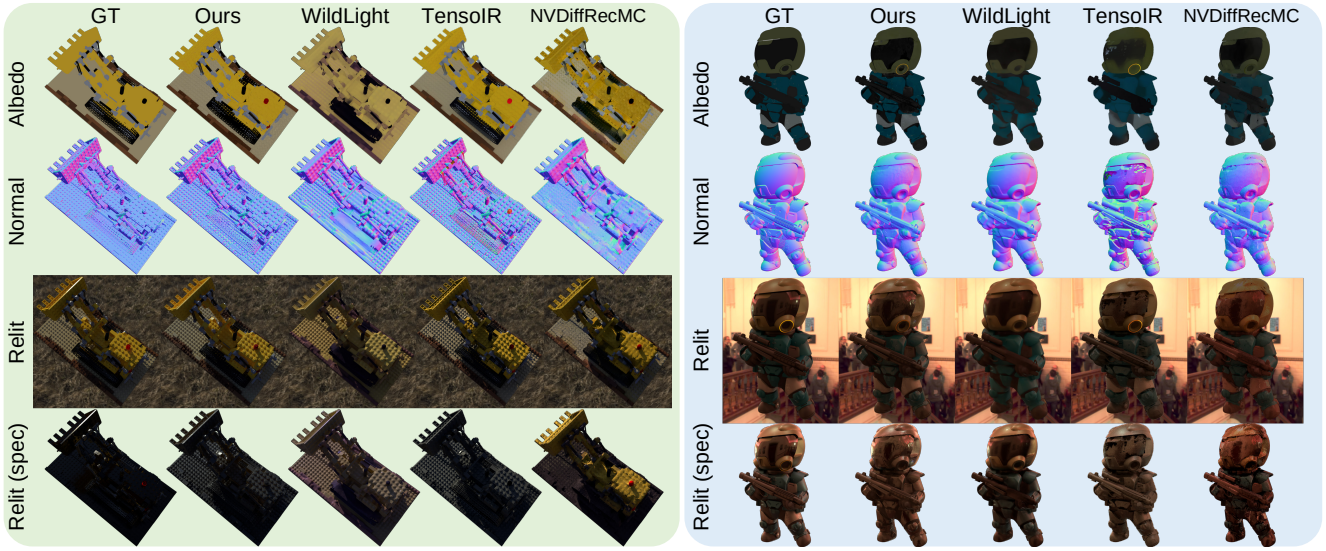

Figure 3. Comparison with state-of-the-art methods. We test NVDiffRecMC [11] (single far-field lighting in input), TensoIR [12] (two far-field lighting in input), and WildLight [6] and our method (single far-field lighting with a flashlight in input) on two synthetic scenes. The intensity of the specular reflection is multiplied by $3\times$ for the visualization purpose. Please zoom in for details.

Table 3. Ablation study on our method using inputs under single far-field lighting with a flashlight.

Model	Normal	Albedo	FV Relit	Relit (spec)
	MAngE↓	PSNR↑	PSNR↑	PSNR↑
(1) w/o $\mathcal{L}_{\text{con}}$	8.452	27.70	29.85	29.08
(2) Mod. $\mathbf{c}_{\text{near}}$	8.572	27.79	30.03	28.87
(3) w/o enc_{mat}	**8.263**	27.79	30.07	28.92
(4) Full model	8.298	**28.05**	**30.19**	**29.36**

lighting. In contrast, WildLight [6] tends to overly smooth geometry and material. TensoIR [12] can not handle highly-reflective surfaces, and lighting is baked into its predicted diffuse albedo for TROOPER.

4.3. Ablation Studies

In Tab. 3, we quantitatively evaluate ablation models on a subset of our synthetic scenes: 1) the model excluding the self-consistency loss $\mathcal{L}_{\text{con}}$, 2) the model where the output of the near-field radiance MLP is not explicitly multiplied by the cosine weakening factor and incident radiance (Eq. (8)), 3) the model with $\text{enc}_{\text{mat}}(\mathbf{x})$ replaced by $\text{enc}_{\text{geo}}(\mathbf{x})$. The results show that all the above techniques enhance the geometry and materials estimated by our method.

4.4. Real-world Results

Fig. 4 shows the results of our method on two real-world scenes captured under single far-field lighting and a flashlight. Our method faithfully recovers the geometry and material of the objects and produces realistic relit results.

5. Conclusion

We introduce a versatile multi-view inverse rendering framework, distinguished by its ability to leverage input images under varied far- and near-field light sources available in capture for better geometry and material estimation.

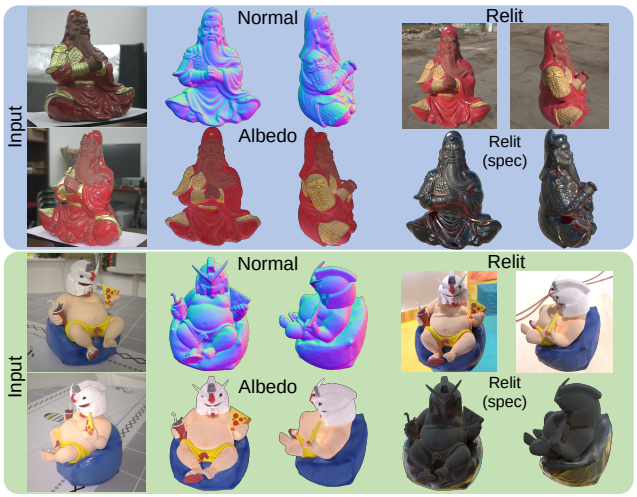

Figure 4. The results of our method on two real-world scenes, using multi-view images under one ambient lighting and a flashlight. The specular reflection intensity is increased for visualization.

Limitations. VMINer does not model the background, necessitating a foreground mask for each image. It also does not consider unknown tone-mapping curves applied during the image signal processing (ISP) stage, using a fixed curve with $\gamma = 2.2$ to the computed linear radiance. It depends on distinguishing the contribution from each lighting, which implies that, for optimal results, near-field light sources should be switched on/off during capture.

Acknowledgments

This work is supported by the National Natural Science Foundation of China (Grand No. 62136001, 62088102).

References

[1] Blender Foundation. The Blender project - free and open 3D creation software. https://www.blender.org. Accessed: 2023-11-07. 2, 6, 7, 13

[2] Mark Boss, Raphael Braun, Varun Jampani, Jonathan T. Barron, Ce Liu, and Hendrik P. A. Lensch. NeRD: Neural reflectance decomposition from image collections. In *Proc. of IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021. 1, 2, 3, 4, 13, 15

[3] Mark Boss, Varun Jampani, Raphael Braun, Ce Liu, Jonathan T. Barron, and Hendrik P. A. Lensch. Neural-PIL: Neural pre-integrated lighting for reflectance decomposition. In *Proc. of Neural Information Processing Systems (NeurIPS)*, 2021.

[4] Mark Boss, Andreas Engelhardt, Abhishek Kar, Yuanzhen Li, Deqing Sun, Jonathan T. Barron, Hendrik P. A. Lensch, and Varun Jampani. SAMURAI: shape and material from unconstrained real-world arbitrary image collections. In *Proc. of Neural Information Processing Systems (NeurIPS)*, 2022. 2

[5] Brent Burley. Physically-based shading at disney. In *SIGGRAPH 2012 Courses*, 2012. 3

[6] Ziang Cheng, Junxuan Li, and Hongdong Li. WildLight: In-the-wild inverse rendering with a flashlight. In *Proc. of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023. 1, 2, 3, 5, 6, 7, 8, 12, 13, 14, 15

[7] Mark Fiala. ARTag, a fiducial marker system using digital techniques. In *Proc. of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2005. 13

[8] Marc-André Gardner, Yannick Hold-Geoffroy, Kalyan Sunkavalli, Christian Gagné, and Jean-François Lalonde. Deep parametric indoor lighting estimation. In *Proc. of IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019. 3

[9] Michael Garland and Paul S. Heckbert. Surface simplification using quadric error metrics. In *Proc. of the ACM SIGGRAPH Conference and Exhibition On Computer Graphics and Interactive Techniques (SIGGRAPH)*, 1997. 13

[10] Amos Gropp, Lior Yariv, Niv Haim, Matan Atzmon, and Yaron Lipman. Implicit geometric regularization for learning shapes. In *Proc. of International Conference on Machine Learning (ICML)*, 2020. 6

[11] Jon Hasselgren, Nikolai Hofmann, and Jacob Munkberg. Shape, light, and material decomposition from images using Monte Carlo rendering and denoising. In *Proc. of Neural Information Processing Systems (NeurIPS)*, 2022. 1, 2, 7, 8, 13, 14

[12] Haian Jin, Isabella Liu, Peijia Xu, Xiaoshuai Zhang, Songfang Han, Sai Bi, Xiaowei Zhou, Zexiang Xu, and Hao Su. TensoIR: Tensorial inverse rendering. In *Proc. of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023. 1, 2, 3, 4, 7, 8, 13, 14

[13] James T. Kajiya. The rendering equation. In *Proc. of the ACM SIGGRAPH Conference and Exhibition On Computer Graphics and Interactive Techniques (SIGGRAPH)*, 1986. 1, 2

[14] Zhengfei Kuang, Kyle Olszewski, Menglei Chai, Zeng Huang, Panos Achlioptas, and Sergey Tulyakov. NeROIC: neural rendering of objects from online image collections. *ACM Transactions on Graphics (TOG)*, 41(4):56:1–56:12, 2022. 1, 2, 3

[15] Eric P Lafortune and Yves D Willems. *Using the modified phong reflectance model for physically based rendering*. Katholieke Universiteit Leuven. Departement Computerwetenschappen, 1994. 6

[16] Bruno Lévy, Sylvain Petitjean, Nicolas Ray, and Jérôme Maillot. Least squares conformal maps for automatic texture atlas generation. *ACM Transactions on Graphics (TOG)*, 21 (3):362–371, 2002. 13

[17] Thomas Lewiner, Hélio Lopes, Antônio Wilson Vieira, and Geovan Tavares. Efficient implementation of marching cubes' cases with topological guarantees. *Journal of Graphics, GPU, & Game Tools*, 8(2):1–15, 2003. 13

[18] Ruilong Li, Hang Gao, Matthew Tancik, and Angjoo Kanazawa. NerfAcc: Efficient sampling accelerates NeRFs. *arXiv preprint arXiv:2305.04966*, 2023. 5

[19] Tzu-Mao Li, Miika Aittala, Frédo Durand, and Jaakko Lehtinen. Differentiable Monte Carlo ray tracing through edge sampling. *ACM Transactions on Graphics (TOG)*, 37(6): 222, 2018. 6

[20] Zhengqin Li, Mohammad Shafiei, Ravi Ramamoorthi, Kalyan Sunkavalli, and Manmohan Chandraker. Inverse rendering for complex indoor scenes: Shape, spatially-varying lighting and SVBRDF from a single image. In *Proc. of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 3

[21] Zhengqin Li, Jia Shi, Sai Bi, Rui Zhu, Kalyan Sunkavalli, Milos Hasan, Zexiang Xu, Ravi Ramamoorthi, and Manmohan Chandraker. Physically-based editing of indoor scene lighting from a single image. In *Proc. of European Conference on Computer Vision (ECCV)*, 2022. 3

[22] Zhaoshuo Li, Thomas Müller, Alex Evans, Russell H. Taylor, Mathias Unberath, Ming-Yu Liu, and Chen-Hsuan Lin. Neuralangelo: High-fidelity neural surface reconstruction. In *Proc. of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023. 1

[23] Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. NeRF: Representing scenes as neural radiance fields for view synthesis. In *Proc. of European Conference on Computer Vision (ECCV)*, 2020. 1, 4

[24] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a multiresolution hash encoding. *ACM Transactions on Graphics (TOG)*, 41(4):102:1–102:15, 2022. 1, 4, 5, 12

[25] Jacob Munkberg, Wenzheng Chen, Jon Hasselgren, Alex Evans, Tianchang Shen, Thomas Müller, Jun Gao, and Sanja Fidler. Extracting triangular 3D models, materials, and lighting from images. In *Proc. of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 1, 2

[26] Vinod Nair and Geoffrey E. Hinton. Rectified linear units improve restricted boltzmann machines. In *Proc. of International Conference on Machine Learning (ICML)*, 2010. 12

[27] Ravi Ramamoorthi and Pat Hanrahan. A signal-processing framework for inverse rendering. In *Proc. of the ACM SIGGRAPH Conference and Exhibition On Computer Graphics and Interactive Techniques (SIGGRAPH)*, 2001. 1

[28] Viktor Rudnev, Mohamed Elgharib, William A. P. Smith, Lingjie Liu, Vladislav Golyanik, and Christian Theobalt. NeRF for outdoor scene relighting. In *Proc. of European Conference on Computer Vision (ECCV)*, 2022. 1, 2, 3

[29] Tim Salimans and Diederik P. Kingma. Weight normalization: A simple reparameterization to accelerate training of deep neural networks. In *Proc. of Neural Information Processing Systems (NeurIPS)*, 2016. 12

[30] Paul-Edouard Sarlin, Cesar Cadena, Roland Siegwart, and Marcin Dymczyk. From coarse to fine: Robust hierarchical localization at large scale. In *Proc. of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 13

[31] Paul-Edouard Sarlin, Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. SuperGlue: Learning feature matching with graph neural networks. In *Proc. of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 13

[32] Johannes Lutz Schönberger and Jan-Michael Frahm. Structure-from-motion revisited. In *Proc. of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 13

[33] Johannes Lutz Schönberger, Enliang Zheng, Marc Pollefeys, and Jan-Michael Frahm. Pixelwise view selection for unstructured multi-view stereo. In *Proc. of European Conference on Computer Vision (ECCV)*, 2016. 13

[34] Pratul P. Srinivasan, Boyang Deng, Xiuming Zhang, Matthew Tancik, Ben Mildenhall, and Jonathan T. Barron. NeRV: Neural reflectance and visibility fields for relighting and view synthesis. In *Proc. of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. 1, 2, 3

[35] Marco Toschi, Riccardo De Matteo, Riccardo Spezialetti, Daniele De Gregorio, Luigi Di Stefano, and Samuele Salti. ReLight My NeRF: A dataset for novel view synthesis and relighting of real world objects. In *Proc. of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023. 5

[36] Eric Veach and Leonidas J. Guibas. Optimally combining sampling techniques for Monte Carlo rendering. In *Proc. of the ACM SIGGRAPH Conference and Exhibition On Computer Graphics and Interactive Techniques (SIGGRAPH)*, 1995. 6

[37] Bruce Walter, Stephen R. Marschner, Hongsong Li, and Kenneth E. Torrance. Microfacet models for refraction through rough surfaces. In *Proc. of the Eurographics Symposium on Rendering Techniques*, 2007. 5, 12

[38] Jiaping Wang, Peiran Ren, Minmin Gong, John M. Snyder, and Baining Guo. All-frequency rendering of dynamic, spatially-varying reflectance. *ACM Transactions on Graphics (TOG)*, 28(5):133, 2009. 4

[39] Peng Wang, Lingjie Liu, Yuan Liu, Christian Theobalt, Taku Komura, and Wenping Wang. NeuS: Learning neural implicit surfaces by volume rendering for multi-view reconstruction. In *Proc. of Neural Information Processing Systems (NeurIPS)*, 2021. 4, 5

[40] Zhou Wang, Alan C. Bovik, Hamid R. Sheikh, and Eero P. Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing (TIP)*, 13(4):600–612, 2004. 7

[41] Zian Wang, Jonah Philion, Sanja Fidler, and Jan Kautz. Learning indoor inverse rendering with 3D spatially-varying lighting. In *Proc. of IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021. 3

[42] Zian Wang, Tianchang Shen, Jun Gao, Shengyu Huang, Jacob Munkberg, Jon Hasselgren, Zan Gojcic, Wenzheng Chen, and Sanja Fidler. Neural fields meet explicit geometric representations for inverse rendering of urban scenes. In *Proc. of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023. 1, 2, 3

[43] Haoqian Wu, Zhipeng Hu, Lincheng Li, Yongqiang Zhang, Changjie Fan, and Xin Yu. NeFII: Inverse rendering for reflectance decomposition with near-field indirect illumination. In *Proc. of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023. 4, 6

[44] Wenqi Yang, Guanying Chen, Chaofeng Chen, Zhenfang Chen, and Kwan-Yee K. Wong. PS-NeRF: Neural inverse rendering for multi-view photometric stereo. In *Proc. of European Conference on Computer Vision (ECCV)*, 2022. 2

[45] Yao Yao, Jingyang Zhang, Jingbo Liu, Yihang Qu, Tian Fang, David McKinnon, Yanghai Tsin, and Long Quan. NeILF: Neural incident light field for physically-based material estimation. In *Proc. of European Conference on Computer Vision (ECCV)*, 2022. 1, 2, 3

[46] Bohan Yu, Siqi Yang, Xuanning Cui, Siyan Dong, Baoquan Chen, and Boxin Shi. MILO: multi-bounce inverse rendering for indoor scene with light-emitting objects. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 45(8):10129–10142, 2023. 1, 2, 3

[47] Chong Zeng, Guojun Chen, Yue Dong, Pieter Peers, Hongzhi Wu, and Xin Tong. Relighting neural radiance fields with shadow and highlight hints. In *Proc. of the ACM SIGGRAPH Conference and Exhibition On Computer Graphics and Interactive Techniques (SIGGRAPH)*, 2023. 3

[48] Jingyang Zhang, Yao Yao, Shiwei Li, Jingbo Liu, Tian Fang, David McKinnon, Yanghai Tsin, and Long Quan. NeILF++: Inter-reflectable light fields for geometry and material estimation. In *Proc. of IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023. 2, 3

[49] Kai Zhang, Fujun Luan, Qianqian Wang, Kavita Bala, and Noah Snavely. PhySG: Inverse rendering with spherical gaussians for physics-based material editing and relighting. In *Proc. of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. 1, 2, 3, 4

[50] Kai Zhang, Fujun Luan, Zhengqi Li, and Noah Snavely. IRON: inverse rendering by optimizing neural SDFs and materials from photometric images. In *Proc. of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 3

[51] Xiuming Zhang, Pratul P. Srinivasan, Boyang Deng, Paul E. Debevec, William T. Freeman, and Jonathan T. Barron. NeR-

Factor: neural factorization of shape and reflectance under an unknown illumination. *ACM Transactions on Graphics (TOG)*, 2021. 1, 2, 3, 4, 6, 7

[52] Yuanqing Zhang, Jiaming Sun, Xingyi He, Huan Fu, Rongfei Jia, and Xiaowei Zhou. Modeling indirect illumination for inverse rendering. In *Proc. of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 1, 2, 3, 4

[53] Youjia Zhang, Teng Xu, Junqing Yu, Yuteng Ye, Yanqing Jing, Junle Wang, Jingyi Yu, and Wei Yang. NeMF: Inverse volume rendering with neural microflake field. In *Proc. of IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023. 2

[54] Fuqiang Zhao, Yuheng Jiang, Kaixin Yao, Jiakai Zhang, Liao Wang, Haizhao Dai, Yuhui Zhong, Yingliang Zhang, Minye Wu, Lan Xu, and Jingyi Yu. Human performance modeling and rendering via neural animated mesh. *ACM Transactions on Graphics (TOG)*, 41(6):235:1–235:17, 2022. 4, 5

[55] Jingsen Zhu, Fujun Luan, Yuchi Huo, Zihao Lin, Zhihua Zhong, Dianbing Xi, Rui Wang, Hujun Bao, Jiaxiang Zheng, and Rui Tang. Learning-based inverse rendering of complex indoor scenes with differentiable Monte Carlo raytracing. In *Proc. of the ACM SIGGRAPH Conference and Exhibition on Computer Graphics and Interactive Techniques in Asia (SIGGRAPH Asia)*, 2022. 3

[56] Jingsen Zhu, Yuchi Huo, Qi Ye, Fujun Luan, Jifan Li, Dianbing Xi, Lisha Wang, Rui Tang, Wei Hua, Hujun Bao, and Rui Wang. I^2-SDF: Intrinsic indoor scene reconstruction and editing via raytracing in neural SDFs. In *Proc. of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023. 1, 3

[57] Yongjie Zhu, Yinda Zhang, Si Li, and Boxin Shi. Spatially-varying outdoor lighting estimation from intrinsics. In *Proc. of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. 3

VMINer: Versatile Multi-view Inverse Rendering with Near- and Far-field Light Sources

Supplementary Material

Fan Fei[1,2,4] Jiajun Tang[1,2,4] Ping Tan[3,4] Boxin Shi[1,2#]

[1]National Key Laboratory for Multimedia Information Processing, School of Computer Science, Peking University

[2]National Engineering Research Center of Visual Technology, School of Computer Science, Peking University

[3]Hong Kong University of Science and Technology [4]Light Illusions

In this supplementary material, we provide the implementation details of VMINer (Sec. 6), the creation details of synthetic and real-world data (Sec. 7), and more qualitative and quantitative results (Sec. 8). The supplementary video (will be published later on the project website) shows additional qualitative comparison results against prior methods.

All the cross-reference numbers (including figures, equations, and references) correspond to the main paper.

6. Implementation Details

6.1. Network Architecture

VMINer has 6 trainable modules: 4 MLPs including $\mathrm{M}_{\mathrm{geo}}$, $\mathrm{M}_{\mathrm{mat}}$, $\mathrm{M}_{\mathrm{far}}$, and $\mathrm{M}_{\mathrm{near}}$, and 2 multi-resolution hash grid encoding including $\mathrm{enc}_{\mathrm{geo}}$ and $\mathrm{enc}_{\mathrm{mat}}$.

The SDF MLP $\mathrm{M}_{\mathrm{geo}}$ contains 1 hidden layer with 64 neurons with $\mathrm{SoftPlus}(x) = \log(1 + e^x)$ as the activation function. We apply weight normalization reparameterization [29] and initialize it so that its output approximates the SDF field of a sphere.

The material MLP $\mathrm{M}_{\mathrm{mat}}$, the far-field radiance MLP $\mathrm{M}_{\mathrm{far}}$, and the near-field radiance MLP $\mathrm{M}_{\mathrm{near}}$ all share the same network architecture: the fully-fused MLP from [24] containing 2 hidden layers with 64 neurons each, using ReLU [26] as in-network activation functions and sigmoid as output activation functions.

For the multi-resolutional hash grid encoding $\mathrm{enc}_{\mathrm{geo}}$ and $\mathrm{enc}_{\mathrm{mat}}$, we set the number of levels $L = 16$, the hash table size $T = 2^{19}$, the coarsest resolution $N_{\min} = 16$, and the finest resolution $N_{\max} = 2048$.

6.2. Training Scheduling

In addition to the above trainable modules, other trainable parameters include the surface concentration parameter b, the light embedding $\mathbf{F}_{\{\mathrm{far,near}\}_i}$, the far-field lighting parameters $\boldsymbol{\xi}_{ij}$, λ_{ij}, and $\boldsymbol{\mu}_{ij}$, and the near-field lighting parameters $\mathbf{p}_i$ (trainable if near_i is not collocated with camera) and $\mathbf{h}_i$.

The base learning rate of the light embedding $\mathbf{F}_{\{\mathrm{far,near}\}_i}$ and the parameters of $\mathrm{enc}_{\mathrm{geo}}$, $\mathrm{M}_{\mathrm{geo}}$, $\mathrm{M}_{\mathrm{far}}$, and $\mathrm{M}_{\mathrm{near}}$, are set to 10^{-2}. The base learning rate of parameters of $\mathrm{enc}_{\mathrm{mat}}$ and $\mathrm{M}_{\mathrm{mat}}$ is set to $3 \cdot 10^{-2}$. The base learning rate of lighting parameters $\boldsymbol{\xi}_{ij}$, λ_{ij}, $\boldsymbol{\mu}_{ij}$, $\mathbf{p}_i$, $\mathbf{h}_i$ is set to $2 \cdot 10^{-2}$. The base learning rate of b is set to 10^{-4}. We incorporate a linear warm-up stage at the start of training, then exponentially decay the learning rate to $0.1\times$ eventually. The geometry and radiance fields $(\mathrm{M}_{\mathrm{geo}}, \mathrm{M}_{\mathrm{far}}, \mathrm{M}_{\mathrm{near}}, \mathrm{enc}_{\mathrm{geo}}, b, \mathbf{F}_{\{\mathrm{far,near}\}_i})$ are frozen during the material optimization stage.

6.3. Loss Function Weights

VMINer is trained using 6 loss terms: the direct supervision losses $\mathcal{L}_{\mathrm{rf}}$ and $\mathcal{L}_{\mathrm{pb}}$, the Eikonal loss $\mathcal{L}_{\mathrm{eik}}$, the silhouette loss $\mathcal{L}_{\mathrm{sil}}$, the self-consistency loss $\mathcal{L}_{\mathrm{con}}$, and the smoothness loss $\mathcal{L}_{\mathrm{smo}}$. The loss terms are divided into two groups and are used separately in two stages: $\mathcal{L}_{\mathrm{stage1}} = \mathcal{L}_{\mathrm{rf}} + \lambda_{\mathrm{eik}}\mathcal{L}_{\mathrm{eik}} + \lambda_{\mathrm{sil}}\mathcal{L}_{\mathrm{sil}} + \lambda_{\mathrm{ns}}\mathcal{L}_{\mathrm{ns}}$ and $\mathcal{L}_{\mathrm{stage2}} = \mathcal{L}_{\mathrm{pb}} + \lambda_{\mathrm{con}}\mathcal{L}_{\mathrm{con}} + \lambda_{\mathrm{ms}}\mathcal{L}_{\mathrm{ms}}$. We typically set $\lambda_{\mathrm{eik}} = 2 \cdot 10^{-4}$, $\lambda_{\mathrm{sil}} = 10^{-3}$, $\lambda_{\mathrm{ns}} = 10^{-3}$, $\lambda_{\mathrm{con}} = 0.1$, and $\lambda_{\mathrm{ms}} = 10^{-3}$.

6.4. BRDF Model

We use a simplified GGX Microfacet BRDF [37] as our BRDF model when reconstructing the scenes. We presume the reflection to be isotropic and fix the parameters beforehand other than the diffuse albedo $\in [0, 1]^3$ and the specular roughness $\in [0, 1]$. As a consequence, the BRDF parameter $\boldsymbol{\beta}$ predicted by the material MLP belongs to the space $[0, 1]^4$.

6.5. Tone Mapping

Due to the requirement of physically-based rendering, our reconstruction pipeline operates in linear color space. Considering that the gamma-corrected sRGB space is usually used in the input images and is closer to human perception, we apply a fixed gamma correction with $\gamma = 2.2$ to the output linear radiance prior to visualization or the computation of losses and error metrics.

6.6. Mesh Extraction

Post-training, we follow a procedure similar to WildLight [6] to process the SDF and BRDF fields into textured

#Corresponding author. E-mail: shiboxin@pku.edu.cn.

meshes. The zero-level isosurface is extracted using the marching cubes algorithm [17], simplified [9], and stored as a triangle mesh. We then use UV unwrapping [16] to generate UV coordinates for each 3D vertex. We rasterize the 3D meshes onto the 2D texture atlas that store the 3D positions on the mesh surfaces for each pixel in the atlas. Normal and material texture maps are generated by querying the geometry and material fields at the specified 3D locations for each pixel. The resulting textured mesh can be easily integrated into industry-standard rendering software like Blender [1], facilitating fast and high-quality rendering suitable for a variety of applications.

7. Data Creation Details

7.1. Synthetic Data

We collect 6 textured meshes: BARRELSOFA, HANDBAG, HOTDOG, LEGO, SHOES, and TROOPER, manifesting a variety of geometries and materials. We render the scenes in Blender under 4 different setups of lighting conditions, as described in Sec. 4.1 in the main paper. We show two example training images of each scene in the first column in Tab. 4 of this document.

7.2. Real-world Data

VMINer assumes the tone mapping curves of input images to be a gamma correction with $\gamma = 2.2$, *i.e.*, it assumes linear images can be obtained by applying an inverse gamma correction. This assumption is also explicitly or implicitly made in various inverse rendering methods, including TensoIR [12] and WildLight [6]. Thus, we have to control the tone mapping curve of the input images. We follow WildLight [6] and use an iPhone 12 Pro as a hand-held camera and the "ProCam" app to take raw images with a linear camera response. We fix the white balance, focal length, exposure time, and ISO for all images. For each lighting combination of far lights and near lights, we keep an approximately constant distance of 0.5 meters from the object and move the camera in a spiral pattern around the objects. We take about 120 images per object, where half of them are taken with the LED light on the iPhone turned on.

We register the camera poses using COLMAP [32, 33] with HLoc [30, 31] for feature extraction and matching. On images lit by different environments, however, image features from backgrounds may have no relevance with features from other backgrounds for COLMAP to work on. To perform camera registration in such case, one can follow NeRD [2] in relying only on the features of the object itself (if they are rich enough) to guide COLMAP. We instead stick the object to a board covered with ARTag [7] for a fallback when COLMAP fails.

After obtaining the captured raw images of the object, we use a custom image signal processor (ISP) to process the raw image by, *e.g.*, demosaicking, white balancing, transforming color space, and most importantly, applying a tone mapping with $\gamma = 2.2$ to let the processed images satisfy our assumption of availability of linear input images. We crop the images to 1200×1200 and manually mark the foreground region of each image to get RGBA images.

8. More Results

8.1. More Quantitative Comparison

We show quantitative results on each scene in Tab. 4 of this document. We observe that although our method does not beat rival methods on some scenes with smooth geometry and materials (*e.g.*, HANDBAG), it generally outperforms prior methods with the same input lighting conditions by a considerable margin.

8.2. More Qualitative Comparison

For the 6 synthetic scenes, we compare our method (one far-field lighting with flashlight) with prior methods, including WildLight [6] (one far-field lighting with flashlight), TensoIR [12] (two far-field lighting), and NVDiffRecMC [11] (one far-field lighting). For the 2 real scenes, GUANYU and DEBUGUNDAM, we compare our method (one far-field lighting with flashlight) with prior methods, including WildLight [6] (one far-field lighting with flashlight), TensoIR [12] (one far-field lighting), and NVDiffRecMC [11] (one far-field lighting). The results under one viewpoint are shown in Fig. 7, Fig. 8, Fig. 9, and Fig. 10 of this document.

It can be observed that our method typically reconstructs more detailed and faithful shapes and materials, including both diffuse albedo and specular reflection parameters. For example, our method succeeds in reconstructing the complicated shape of LEGO, whereas WildLight [6] fails to recover the geometrical details and TensoIR [12] produces the wrong surface normal of the continuous tracks of the bulldozer. In BARRELSOFA, our method satisfactorily reproduces the text and graphics on the cap of the barrel while WildLight [6] and TensoIR [12] can only produce blurry results. Our method also reconstructs the spatially-varying material parameters that can correctly reproduce highlights with varied sharpness on different parts of the scene (*e.g.*, the metal and the leather in BARRELSOFA, the plate and the hotdog in HOTDOG, the clay and the polished porcelain surface in GUANYU, and the rough and smooth PVC in DEBUGUNDAM).

However, on some scenes with spatially-uniform materials (*e.g.*, HANDBAG and the plate of HOTDOG), our method do not outperform WildLight [6]. The strong cast shadow in HANDBAG and inter-reflection in HOTDOG somewhat remain in the result of our method. However, it is known that in reconstruction-based inverse rendering methods (including ours), hard cast shadows can undesirably alter the ma-

Table 4. Quantitative comparison results with state-of-the-art methods on each of the 6 synthetic scenes. Notations of input lighting conditions: "1F" means single far-field lighting, "1F1N" means single far-field lighting with single near-field lighting, "2F" means two far-field lighting, and "2F1N" means two far-field lighting with single near-field lighting. On the first column, for each scene we show two example training images, the upper one under the far-field lighting in "1F" and the lower one under another far-field lighting in "2F". We show results of surface normal, diffuse albedo, view synthesis RGB, free-viewpoint (FV) relit RGB, the specular reflection part of FV relit RGB. We mark the **best** and the second best results in each column. ↑ (↓) means bigger (smaller) is better.

Scene	Method (input lighting)	Normal	Albedo			View synthesis			FV relit			FV relit (spec)		
		MAngE↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
BARRELSOFA	(1) TensoIR [12] (1F)	8.469	25.554	0.892	0.183	29.824	0.898	0.224	27.888	0.873	0.226	26.724	0.813	0.218
	(2) NVDiffRecMC [11] (1F)	12.141	22.231	0.867	0.140	24.990	0.911	0.125	24.513	0.871	0.148	24.832	0.796	0.203
	(3) Ours (1F)	10.751	26.055	0.876	0.142	30.151	0.943	0.112	27.087	0.905	0.140	25.914	0.834	0.188
	(4) WildLight [6] (1F1N)	6.905	24.749	0.884	0.171	26.578	0.917	0.128	26.660	0.903	0.135	26.847	0.844	0.160
	(5) Ours (1F1N)	**6.525**	31.683	0.945	0.092	**32.321**	**0.956**	0.085	**31.186**	0.946	0.097	**28.972**	**0.881**	0.159
	(6) TensoIR [12] (2F)	9.528	25.851	0.892	0.187	28.886	0.892	0.228	28.290	0.877	0.222	26.883	0.822	0.208
	(7) Ours (2F)	8.261	26.435	0.876	0.131	29.361	0.941	0.110	27.975	0.920	0.122	27.085	0.852	0.171
	(8) Ours (2F1N)	6.701	**32.378**	**0.948**	**0.085**	31.745	0.955	**0.082**	31.172	**0.948**	**0.090**	28.575	0.866	**0.156**
HANDBAG	(1) TensoIR [12] (1F)	18.277	25.258	0.920	0.069	28.652	0.917	0.110	30.220	0.922	0.116	30.098	0.900	0.134
	(2) NVDiffRecMC [11] (1F)	12.949	27.473	0.921	0.126	26.993	0.912	0.130	30.037	0.937	0.108	28.537	0.890	0.125
	(3) Ours (1F)	6.771	22.460	0.892	0.149	26.348	0.937	**0.074**	28.529	0.932	0.092	31.134	0.920	0.120
	(4) WildLight [6] (1F1N)	**5.978**	**33.687**	**0.981**	**0.025**	32.338	**0.946**	0.108	**37.125**	**0.972**	**0.075**	**35.255**	**0.945**	**0.098**
	(5) Ours (1F1N)	6.097	31.855	0.950	0.087	33.014	**0.946**	0.107	35.885	0.967	0.084	32.361	0.931	0.115
	(6) TensoIR [12] (2F)	17.261	27.250	0.928	0.068	28.682	0.915	0.104	31.304	0.927	0.112	30.828	0.910	0.128
	(7) Ours (2F)	6.482	22.342	0.887	0.152	25.651	0.930	0.086	28.180	0.937	0.090	31.335	0.923	0.118
	(8) Ours (2F1N)	6.067	32.760	0.954	0.088	**33.259**	0.943	0.110	36.795	0.968	0.083	34.329	**0.945**	0.109
HOTDOG	(1) TensoIR [12] (1F)	14.951	23.341	0.933	0.120	26.916	0.874	0.177	24.734	0.879	0.177	23.506	0.799	0.194
	(2) NVDiffRecMC [11] (1F)	13.397	23.880	0.944	0.111	21.504	0.869	0.157	20.325	0.874	0.166	23.291	0.794	0.204
	(3) Ours (1F)	11.273	19.732	0.857	0.171	23.533	0.913	0.108	22.493	0.891	0.134	22.483	0.818	0.161
	(4) WildLight [6] (1F1N)	10.254	24.238	0.952	**0.076**	29.563	0.936	0.087	30.163	0.939	**0.080**	20.070	0.794	0.154
	(5) Ours (1F1N)	10.072	24.422	0.940	0.112	29.733	**0.949**	**0.080**	29.186	0.937	0.090	29.745	0.905	0.123
	(6) TensoIR [12] (2F)	11.197	24.103	0.949	0.094	27.544	0.878	0.177	24.912	0.892	0.167	25.020	0.817	0.197
	(7) Ours (2F)	10.541	22.501	0.913	0.130	26.988	0.928	0.104	27.332	0.920	0.107	25.838	0.864	0.149
	(8) Ours (2F1N)	**9.326**	**24.926**	**0.954**	0.094	**30.520**	0.944	0.085	**30.449**	**0.942**	0.085	**30.557**	**0.913**	**0.112**
LEGO	(1) TensoIR [12] (1F)	20.195	27.988	0.937	0.089	29.982	0.918	0.090	32.029	0.915	0.093	33.296	0.835	0.156
	(2) NVDiffRecMC [11] (1F)	27.239	25.729	0.889	0.146	30.724	0.916	0.104	32.438	0.919	0.103	25.751	0.825	0.177
	(3) Ours (1F)	18.952	24.875	0.830	0.178	28.778	0.906	0.081	31.058	0.912	0.082	31.607	0.840	0.152
	(4) WildLight [6] (1F1N)	23.565	23.870	0.909	0.129	26.441	0.861	0.135	29.035	0.882	0.123	24.604	0.744	0.175
	(5) Ours (1F1N)	18.076	30.801	0.941	0.109	30.834	**0.929**	0.076	33.366	0.940	0.074	34.096	0.856	0.146
	(6) TensoIR [12] (2F)	19.691	28.434	**0.947**	**0.083**	29.659	0.915	0.092	32.039	0.920	0.089	32.187	0.822	0.164
	(7) Ours (2F)	17.878	26.576	0.861	0.165	29.258	0.913	0.079	31.718	0.928	0.074	32.135	0.859	0.155
	(8) Ours (2F1N)	**17.287**	**31.201**	0.945	0.101	**31.150**	0.927	**0.075**	**33.889**	**0.943**	**0.068**	**34.966**	**0.873**	**0.139**
SHOES	(1) TensoIR [12] (1F)	22.918	24.067	0.881	0.150	28.876	0.915	0.111	23.329	0.875	0.117	26.199	0.893	0.126
	(2) NVDiffRecMC [11] (1F)	19.268	24.577	0.902	0.119	25.027	0.900	0.125	20.591	0.845	0.169	22.772	0.842	0.155
	(3) Ours (1F)	17.267	20.508	0.894	0.075	25.133	0.934	0.061	21.428	0.920	0.079	23.130	0.877	0.119
	(4) WildLight [6] (1F1N)	**13.843**	29.927	0.932	0.116	29.928	0.937	0.095	25.440	0.912	0.111	25.655	0.903	0.102
	(5) Ours (1F1N)	16.786	31.241	0.965	0.052	29.494	0.959	0.050	26.118	**0.949**	0.057	27.249	0.914	**0.089**
	(6) TensoIR [12] (2F)	22.102	24.923	0.898	0.128	30.165	0.924	0.105	23.816	0.858	0.116	24.939	0.886	0.112
	(7) Ours (2F)	17.182	25.004	0.926	0.061	**30.857**	**0.964**	0.052	**26.142**	0.935	0.068	26.838	0.909	0.095
	(8) Ours (2F1N)	17.006	**31.815**	**0.967**	**0.051**	29.245	0.958	**0.048**	25.936	**0.949**	**0.055**	**27.578**	**0.915**	0.098
TROOPER	(1) TensoIR [12] (1F)	21.151	32.665	0.961	0.062	32.632	0.953	0.111	30.876	0.942	0.115	29.969	0.928	0.102
	(2) NVDiffRecMC [11] (1F)	12.418	35.243	0.968	0.041	33.511	0.970	0.060	31.732	0.960	0.073	28.219	0.869	0.119
	(3) Ours (1F)	9.321	33.363	0.941	0.085	35.241	0.972	0.058	34.184	0.962	0.068	31.108	0.935	0.087
	(4) WildLight [6] (1F1N)	8.367	36.658	**0.979**	**0.030**	34.386	0.974	0.042	34.205	0.974	0.047	**33.834**	0.946	**0.060**
	(5) Ours (1F1N)	**7.755**	**39.743**	0.978	0.041	**37.113**	**0.979**	**0.038**	**36.239**	**0.977**	**0.045**	32.071	0.948	0.069
	(6) TensoIR [12] (2F)	17.660	32.541	0.961	0.063	33.160	0.949	0.114	31.576	0.940	0.117	30.928	0.924	0.103
	(7) Ours (2F)	9.862	33.579	0.947	0.064	35.420	0.974	0.050	34.252	0.964	0.064	31.303	0.935	0.088
	(8) Ours (2F1N)	8.349	39.169	0.975	0.042	36.707	0.977	0.039	36.046	0.976	**0.045**	32.385	**0.949**	0.068

Figure 5. Comparison between the reconstruction under an anisotropic point light model (adopted by our method) and under an isotropic point light model.

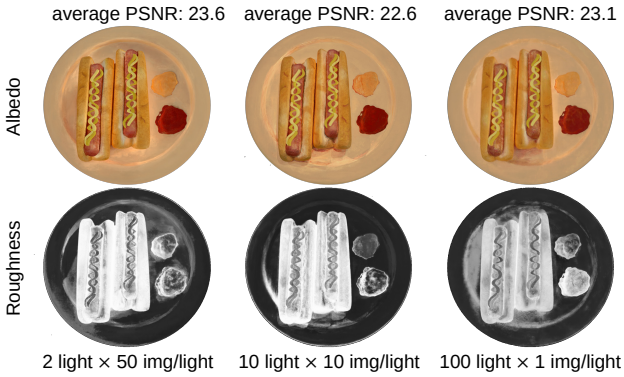

Figure 6. Results on synthetic scenes with varying amounts of far-field lighting.

terial to compensate for imperfect estimation of shading. In that context, one key characteristic of WildLight [6] is that its reconstructed radiance is the summation of the neural radiance from the ambient lighting and the PBR radiance from the flashlight. This additive ambiguity enables WildLight [6] to ascribe spatial variation of radiance to the neural radiance instead of material. This helps WildLight [6] to generate diffuse albedo maps that are free of high-frequency shading effects such as the strong cast shadow on HANDBAG from which other methods suffer, but also let it predict textureless materials, which can be easily observed on BARRELSOFA. We believe that this shade-baking problem can be alleviated by introducing more priors on the material (more likely in a data-driven manner) in future works.

For results under all viewpoints, please see the attached supplementary video.

8.3. The Anisotropic Point Light Model

Fig. 5 shows that our anisotropic point light model (Eq. (3)) can accurately reconstruct the radiance under a fixed and anisotropic spotlight while an isotropic one the same as that used by WildLight [6] fails.

8.4. Results on Few-images-per-lighting Data

Our work does not focus on Internet photo collections (*e.g.*, there are hundreds of photos of an object, each under a unique and unknown lighting) like NeRD [2] because: 1) Internet photo collections cannot leverage appearance variation under near-field lights, 2) Internet photo collections aims at different application scenarios (things on the Internet, usually landmarks with millions of photos) instead of self-captured photos (things at hand, for which capturing a few environments is more practical). Nevertheless, our method works well on simulated Internet photo collections. Fig. 6 shows our results on synthetic scenes with varying amounts of far-field lighting: 1) 2 lighting, 50 images/lighting, 2) 10 lighting, 10 images/lighting, and 3) 100 lighting, 1 image/lighting (simulating Internet photo collections). Due to the absence of near-field lights in the above setting, there is no more demand to separate radiance under different light sources. As a consequence, the reduced view interpolation ability of the radiance cache (due to fewer images per lighting) does not prevent our method from estimating reasonable material.

Figure 7. Comparison with state-of-the-art methods on two synthetic scenes: BARRELSOFA and HANDBAG.

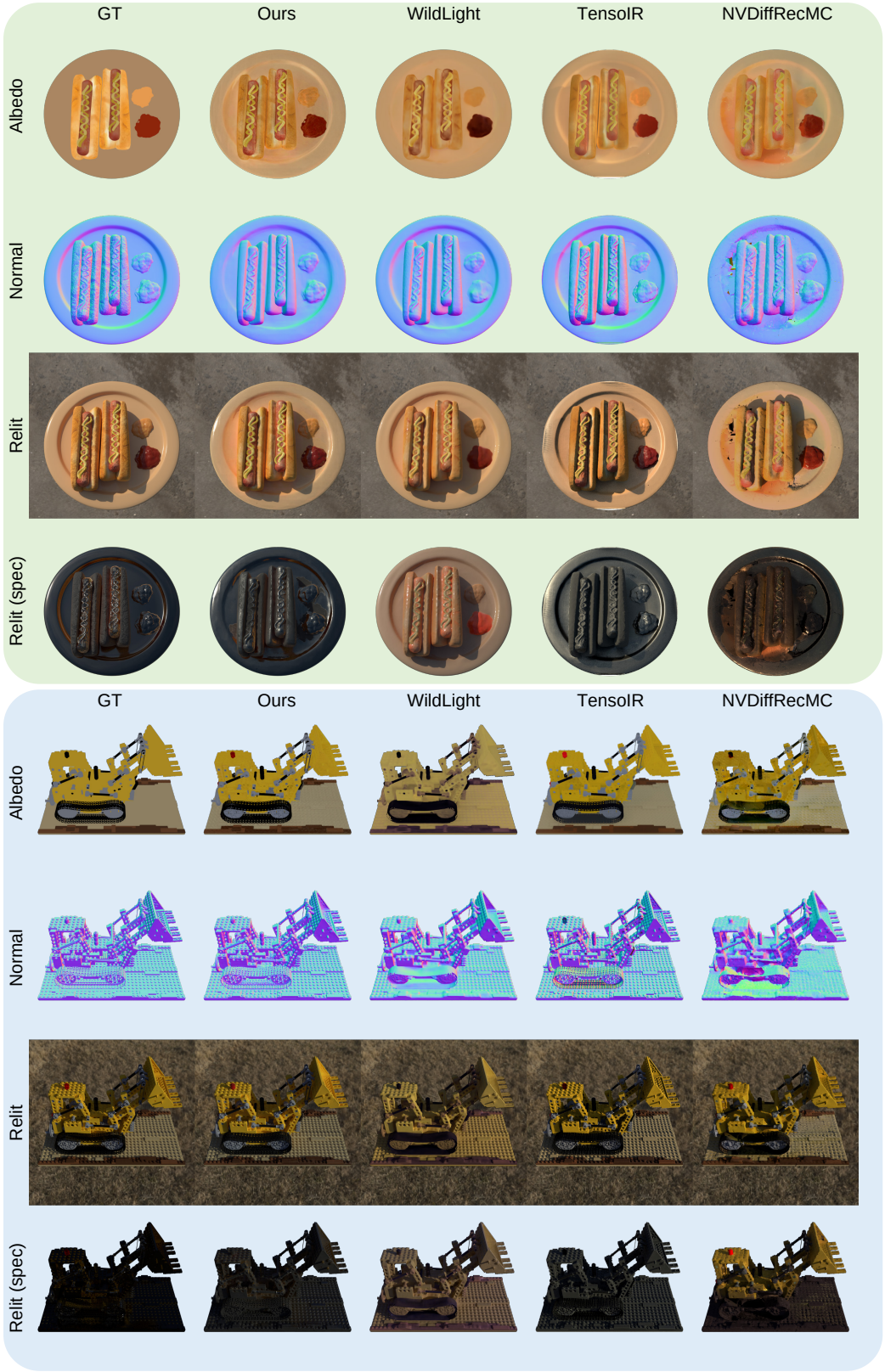

Figure 8. Comparison with state-of-the-art methods on two synthetic scenes: HOTDOG and LEGO.

Figure 9. Comparison with state-of-the-art methods on two synthetic scenes: SHOES and TROOPER.

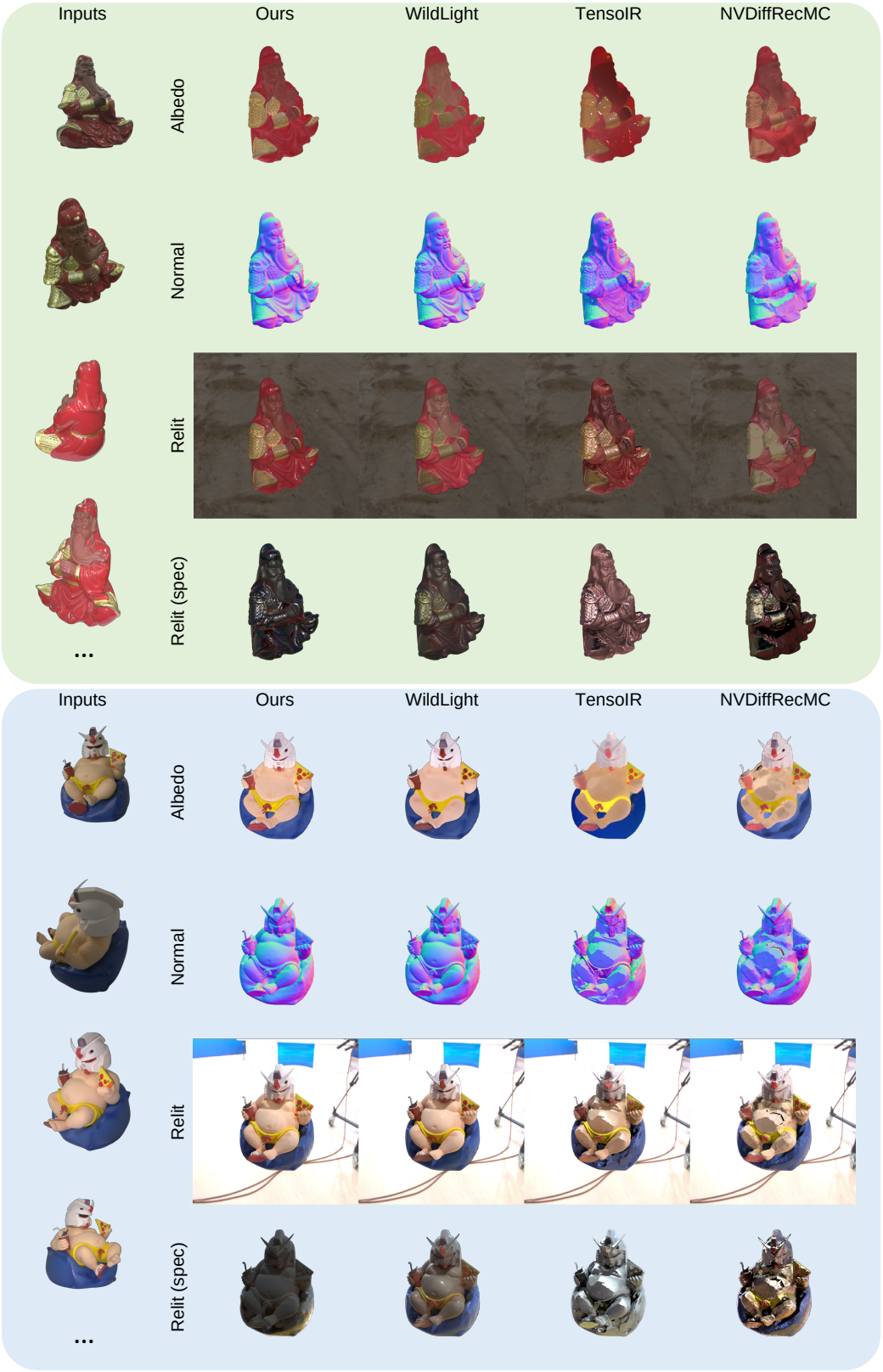

Figure 10. Comparison with state-of-the-art methods on two real-world scenes: GUANYU and DEBUGUNDAM.