Supplementary Document

A. Introduction

This is the supplementary document for NeISF. We discuss some details that are not shown in the main paper due to space limitations. The remainder of this document is structured as follows. We analyze our polarimetric renderer in Sec. B. In Sec. C, we introduce the proposed synthetic and real-world polarimetric datasets. We provide additional experimental results in Sec. D. A detailed physical background of the polarimetric rendering is shown in Sec. E. Finally, We introduce the implementation details of our model in Sec. F.

B. Polarimetric Renderer

Our polarimetric renderer is implemented by PyTorch [7]. To verify the correctness of the constructed renderer, we compare the rendering results of our renderer with Mitsuba 3.0 [4]. The material model is Baek pBRDF [2]. For simplicity, we only render a sphere with diffuse albedo $\boldsymbol{\rho}$ set to $[0.5, 0.5, 0.5]$ and roughness r set to 0.1. The refractive index of air is set to 1.0, and the refractive index of the sphere is set to 1.5. The illumination is a pure white environment map and we only render the direct illumination. As Fig 1 shows, our renderer can produce similar results as Mitsuba 3.0.

C. Datasets

C.1. Synthetic

We rendered our synthetic dataset using Mitsuba 3.0 [4] with polarized mode. For the material, we used the Baek pBRDF [2] model. We set the refractive index of air to 1.0, the refractive index of the object to 1.5, and the specular coefficient to 1.0. The diffuse albedo is textured but the roughness is spatially-constant for each 3D mesh because Mitsuba 3.0 does not support a spatially-varying roughness for Baek pBRDF. We used open-source 3D objects as our geometry. As shown in Fig. 3, we placed the object inside a modified Cornell Box. Specifically, the material of the wall was also set to Baek pBRDF. Besides, we used a very small (less than 0.1) roughness to make sure the specular reflection was strong inside the box. We used two illuminations, one is an area light under the ceiling of the box, and another one is an environment map. Although both light sources are unpolarized, the multiple bounces inside the box can make the light polarized before interacting with the object. For each object, we rendered 110 images. Each image contains a 9-channel Stokes vectors map, a 3-channel surface normal map, a 3-channel albedo map, a 1-channel roughness map, a 3-channel specular intensity image, a 3-channel diffuse intensity image, and a 1-channel object mask. The cameras were uniformly distributed on the hemisphere around the object. We used 100 images for training and 10 images for testing. The resolution for the synthetic dataset is 700 $\times$ 700, and we rendered 4096 samples per pixel.

C.2. Real-world

Images from the real-world dataset were captured by a FLIR BFS-U3-51S5PC-C polarization camera with a Sony IMX250MYR sensor. For each viewpoint, we captured eight images with the exposure time [4, 8, 16, 32, 64, 128, 256, 512] ms. Then, we composite them to obtain one HDR image $\mathbf{I}_{\text{HDR}}$ in the raw image domain. We apply the alpha blending to minimize the noise in the composited image $\mathbf{I}_{\text{HDR}}$ as below.

$$I_{\text{HDR}} = \alpha \cdot g_{\text{short}} \cdot I_{\text{short}} + (1-\alpha) \cdot g_{\text{long}} \cdot I_{\text{long}}, \quad (1)$$

where I_{short} and I_{long} are intensities of a short-exposure and a long-exposure image respectively. g_{short} and g_{long} are gain values to equalize the level of each image. These are calculated from the ratio of the exposure time. We get the optimal weight α by minimizing the noise variance σ^2_{HDR}. This can be represented using noise variances of two images, σ^2_{short}, σ^2_{long},

$$\sigma^2_{\text{HDR}}(\alpha) = \alpha^2 \cdot g^2_{\text{short}} \cdot \sigma^2_{\text{short}} + (1-\alpha)^2 \cdot g^2_{\text{long}} \cdot \sigma^2_{\text{long}}. \quad (2)$$

Note that our main paper uses σ as density in Eq. (9). However, we denote σ^2 as noise variance in this section. The optimal weight $\hat{\alpha}$ that minimizes Eq. 2 can be simply obtained by solving

$$\frac{\partial \sigma^2_{\text{HDR}}(\alpha)}{\partial \alpha} = 0. \quad (3)$$

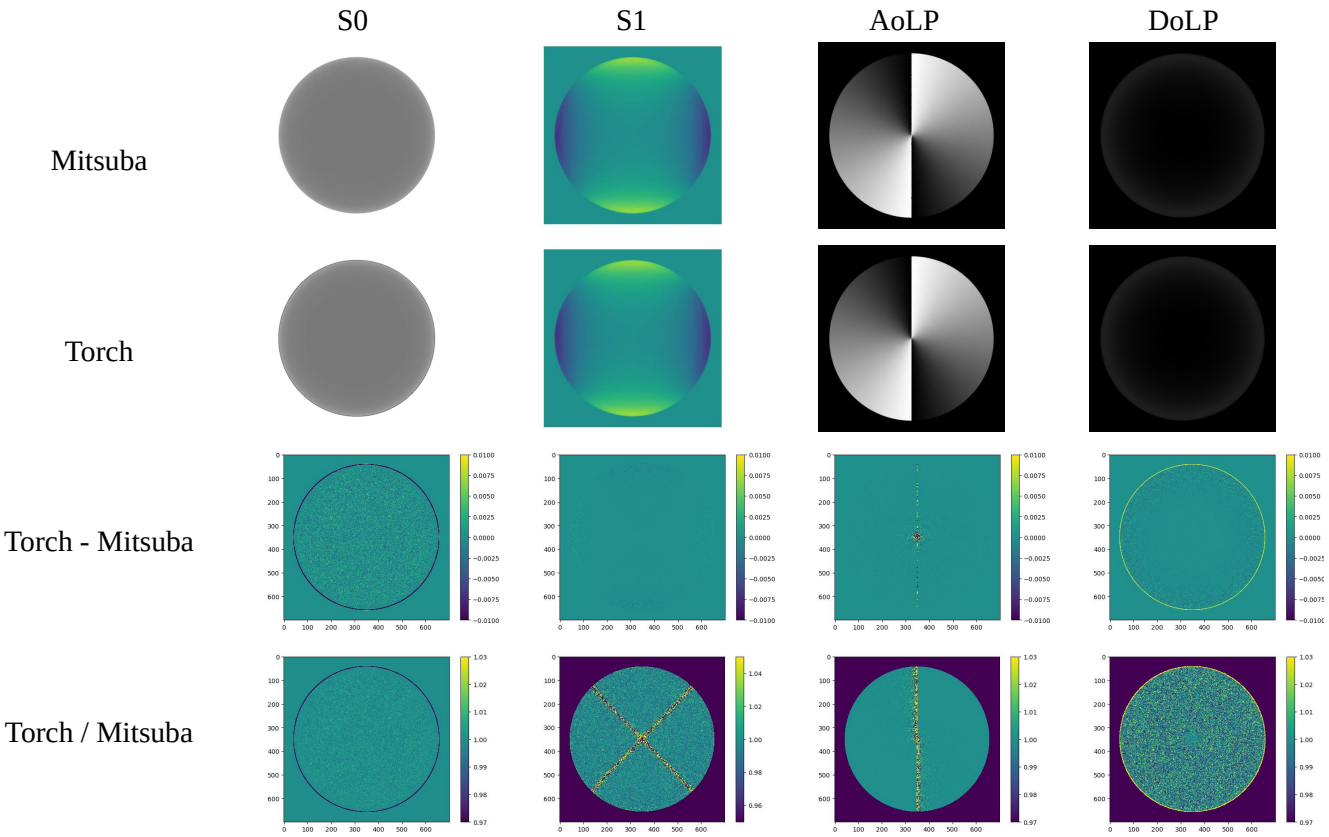

Figure 1. The rendering results between our renderer (denoted as "Torch" in the figure) and Mitsuba 3.0 [4].

Data capturing Camera poses estimation

Figure 2. Scene setup of the real-world dataset. Left: We placed the object on a round table and moved the camera around it. Right: Camera poses were estimated by COLMAP [8, 9].

Figure 3. Scene setup of the synthetic dataset. We placed the object inside a modified Cornell Box. The light bounces multiple times and becomes polarized before hitting the object.

The optimal weight $\hat{\alpha}$ is

$$\hat{\alpha} = \underset{\alpha}{\operatorname{argmin}}\ \sigma^2_{\mathrm{HDR}}(\alpha) \tag{4}$$

$$= \frac{g^2_{\mathrm{long}} \cdot \sigma^2_{\mathrm{long}}}{g^2_{\mathrm{short}} \cdot \sigma^2_{\mathrm{short}} + g^2_{\mathrm{long}} \cdot \sigma^2_{\mathrm{long}}} \tag{5}$$

Here, the variances of noise in the raw image, $\sigma^2_{\mathrm{short}}$, σ^2_{long}, are assumed to follow a shot noise model where the noise variance has a linear relationship with the expected intensity μ [14],

$$\sigma^2 = a \cdot \mu + b. \tag{6}$$

Parameters a and b can be estimated by fitting the mean intensity and variance obtained from a series of raw images of a scene with different brightness to Eq. 6. Thus, we can estimate the noise variance of the raw image by substituting the intensity $\mathbf{I}$ to Eq. 6. We recursively apply Eq. 1 to expand to eight images' composition.

After demosaicing, we can get four polarized images with the polarization angle $[0°, 45°, 90°, 135°]$, and we denote them as $\mathbf{I}_0, \mathbf{I}_{45}, \mathbf{I}_{90}, \mathbf{I}_{135}$. The Stokes vectors can be calculated as follows:

$$\mathbf{s}[0] = (\mathbf{I}_0 + \mathbf{I}_{45} + \mathbf{I}_{90} + \mathbf{I}_{135})/2, \tag{7}$$

$$\mathbf{s}[1] = \mathbf{I}_0 - \mathbf{I}_{90}, \tag{8}$$

$$\mathbf{s}[2] = \mathbf{I}_{45} - \mathbf{I}_{135}. \tag{9}$$

We selected 3 real-world objects which are "Dinosaur", "Sumo", and "Sakura pot". For each object, we took 96 viewpoints for training and 3-7 viewpoints for evaluation. The resolution of the captured Stokes vectors is 1224 $\times$ 1024. Due to the limited computational resources, we re-scaled the resolution to 612 $\times$ 512 before training. In addition, we manually created a binary mask using Photoshop [1] for each viewpoint. The camera poses were calculated by COLMAP [8, 9]. Fig. 2 shows the capture settings and the reconstructed camera poses.

D. Additional Results

We provide additional results here due to the page limitation of the main paper. In Fig. 4, we show that our method is also capable of handling objects that have complex geometry. We can reconstruct a high-fidelity geometry while the other methods lose some details. In Fig. 5, we show that with the aid of polarization cues, the reconstructed roughness map is much cleaner due to the better disentanglement of geometry and material. Finally, in Fig. 6, we show the contribution of the joint optimization.

E. Physical Background

This section is an extension of the physical background of the main paper and contains more details. Some contents are reused from the main paper.

E.1. Polarimetric rendering

In Beak pBRDF [2], the rendered Stokes vectors can be obtained by the combination of the diffuse and specular components:

$$\mathbf{s}^{\mathrm{cam}} = \mathbf{s}^{\mathrm{cam}}_{\mathrm{dif}} + \mathbf{s}^{\mathrm{cam}}_{\mathrm{spec}}, \tag{10}$$

and the diffuse and specular components should be treated separately:

$$\mathbf{s}^{\mathrm{cam}}_{\mathrm{dif}} = \mathbf{R}^{\mathrm{cam}}_{\mathrm{dif}} \cdot \int_{\Omega} \mathbf{M}_{\mathrm{dif}} \cdot \mathbf{s}^{\mathrm{r}}_{\mathrm{dif}}\, d\boldsymbol{\omega_i}, \tag{11}$$

$$\mathbf{s}^{\mathrm{cam}}_{\mathrm{spec}} = \int_{\Omega} \mathbf{R}^{\mathrm{cam}}_{\mathrm{spec}} \cdot \mathbf{M}_{\mathrm{spec}} \cdot \mathbf{s}^{\mathrm{r}}_{\mathrm{spec}}\, d\boldsymbol{\omega_i}. \tag{12}$$

For the diffuse component, the rotation matrix $\mathbf{R}^{\mathrm{cam}}_{\mathrm{dif}}$ is the same for all incident light directions. For convenience, we can also put the rotation matrix into the integral, then we can rewrite Eq. 11 as:

$$\mathbf{s}^{\mathrm{cam}}_{\mathrm{dif}} = \int_{\Omega} \mathbf{R}^{\mathrm{cam}}_{\mathrm{dif}} \cdot \mathbf{M}_{\mathrm{dif}} \cdot \mathbf{s}^{\mathrm{r}}_{\mathrm{dif}}\, d\boldsymbol{\omega_i}. \tag{13}$$

Where $\mathbf{s}^{\mathrm{r}}_{\mathrm{dif}}$ and $\mathbf{s}^{\mathrm{r}}_{\mathrm{spec}}$ are the already rotated incident Stokes vectors of diffuse and specular components and they are estimated by MLPs. In contrary, $\mathbf{R}^{\mathrm{cam}}_{\mathrm{dif}} \cdot \mathbf{M}_{\mathrm{dif}}$ and $\mathbf{R}^{\mathrm{cam}}_{\mathrm{spec}} \cdot \mathbf{M}_{\mathrm{spec}}$ are modeled explicitly.

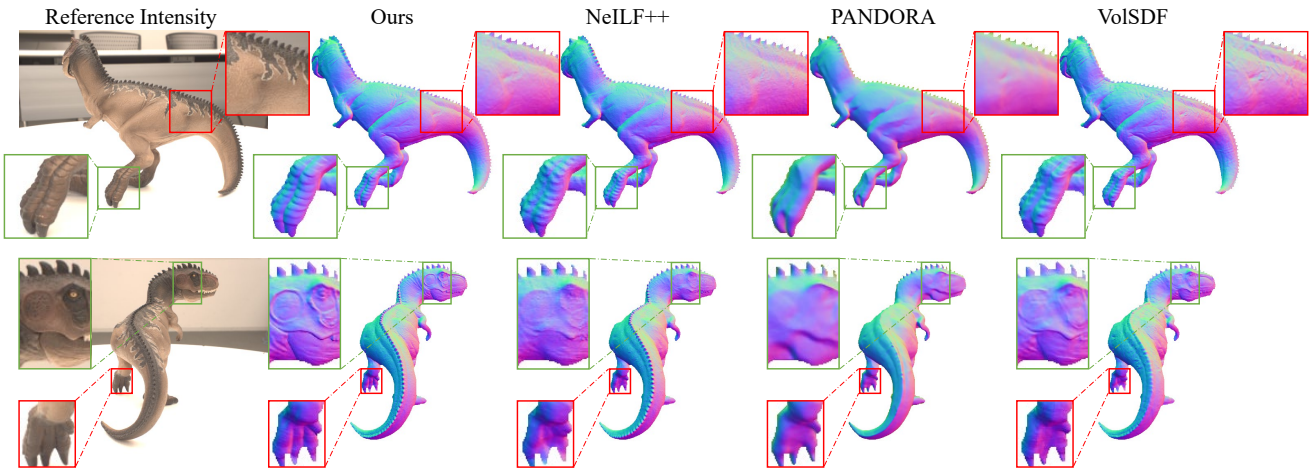

Figure 4. Qualitative comparison of the reconstructed surface normal of the real dataset.

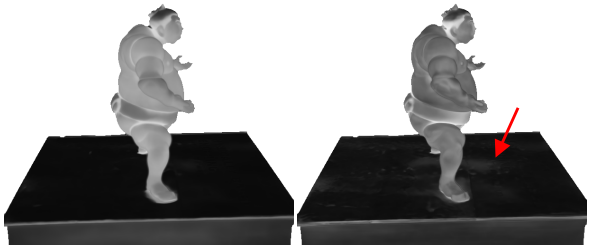

Figure 5. Roughness comparison. Without polarization cues, the reconstructed roughness is easily affected by geometry and shadows.

$\mathbf{R}_{\text{dif}}^{\text{cam}}$ with the rotation angle ϕ_{dif} is as follows:

$$\mathbf{R}_{\text{dif}}^{\text{cam}} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(2\phi_{\text{dif}}) & \sin(2\phi_{\text{dif}}) \\ 0 & -\sin(2\phi_{\text{dif}}) & \cos(2\phi_{\text{dif}}) \end{bmatrix}. \tag{14}$$

$\mathbf{M}_{\text{dif}}$ can be formulated as follows:

$$\mathbf{M}_{\text{dif}} = (\frac{\rho}{\pi}\cos\theta_i)\mathbf{F}_o^T \cdot \mathbf{D} \cdot \mathbf{F}_i^T. \tag{15}$$

ρ is the diffuse albedo, $\theta_{i,o}$ denotes the incident/outgoing angle, $\mathbf{D} \in \mathbb{R}^{3\times3}$ is a depolarizer:

$$\mathbf{D} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \tag{16}$$

the Fresnel transmission term $\mathbf{F}_{i,o}^T \in \mathbb{R}^{3\times3}$ is defined as:

$$\mathbf{F}_{i,o}^T = \begin{bmatrix} T_{i,o}^+ & T_{i,o}^- & 0 \\ T_{i,o}^- & T_{i,o}^+ & 0 \\ 0 & 0 & T_{i,o}^\times \end{bmatrix}, \tag{17}$$

where $T_{i,o}^+ = (T_{i,o}^\perp + T_{i,o}^\parallel)/2$, $T_{i,o}^- = (T_{i,o}^\perp - T_{i,o}^\parallel)/2$, and $T_{i,o}^\times = \sqrt{T_{i,o}^\perp T_{i,o}^\parallel}$. Where $T_{i,o}^\perp$ is the perpendicular term of the transmission coefficient:

$$T_{i,o}^\perp = \frac{4\cos\theta_{i,o}\sqrt{\eta^2 - \sin^2\theta_{i,o}}}{(\cos\theta_{i,o} + \sqrt{\eta^2 - \sin^2\theta_{i,o}})^2}, \tag{18}$$

and $T_{i,o}^\parallel$ is the parallel term of the transmission coefficient:

$$T_{i,o}^\parallel = \frac{4\eta^2\cos\theta_{i,o}\sqrt{\eta^2 - \sin^2\theta_{i,o}}}{(\eta^2\cos\theta_{i,o} + \sqrt{\eta^2 - \sin^2\theta_{i,o}})^2}, \tag{19}$$

where η is the refractive index of the object. Then, we can rewrite Eq. 13 as follows:

$$\mathbf{s}_{\text{dif}}^{\text{cam}} = \int_\Omega \frac{\rho}{\pi}\cos\theta_i \begin{bmatrix} T_o^+T_i^+ & T_o^+T_i^- & 0 \\ T_o^-T_i^+\cos(2\phi_{\text{dif}}) & T_o^-T_i^-\cos(2\phi_{\text{dif}}) & 0 \\ -T_o^-T_i^+\sin(2\phi_{\text{dif}}) & -T_o^-T_i^-\sin(2\phi_{\text{dif}}) & 0 \end{bmatrix} \cdot \mathbf{s}_{\text{dif}}^{\text{r}}\, d\boldsymbol{\omega_i}, \tag{20}$$

where ϕ_{dif} is the rotation angle from the reference frame of $\mathbf{M}_{\text{dif}}$ to the camera axis. Note that elements in the third column of the matrix are all zero. That is the reason we do not estimate $\mathbf{s}_{\text{dif}}^{\text{r}}[2]$ in the main paper. In addition, when rendering RGB images, Eq. 20 should be repeated three times with separate diffuse albedos.

Similarly, $\mathbf{R}_{\text{spec}}^{\text{cam}}$ with the rotation angle ϕ_{spec} is as follows:

$$\mathbf{R}_{\text{spec}}^{\text{cam}} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(2\phi_{\text{spec}}) & \sin(2\phi_{\text{spec}}) \\ 0 & -\sin(2\phi_{\text{spec}}) & \cos(2\phi_{\text{spec}}) \end{bmatrix}. \tag{21}$$

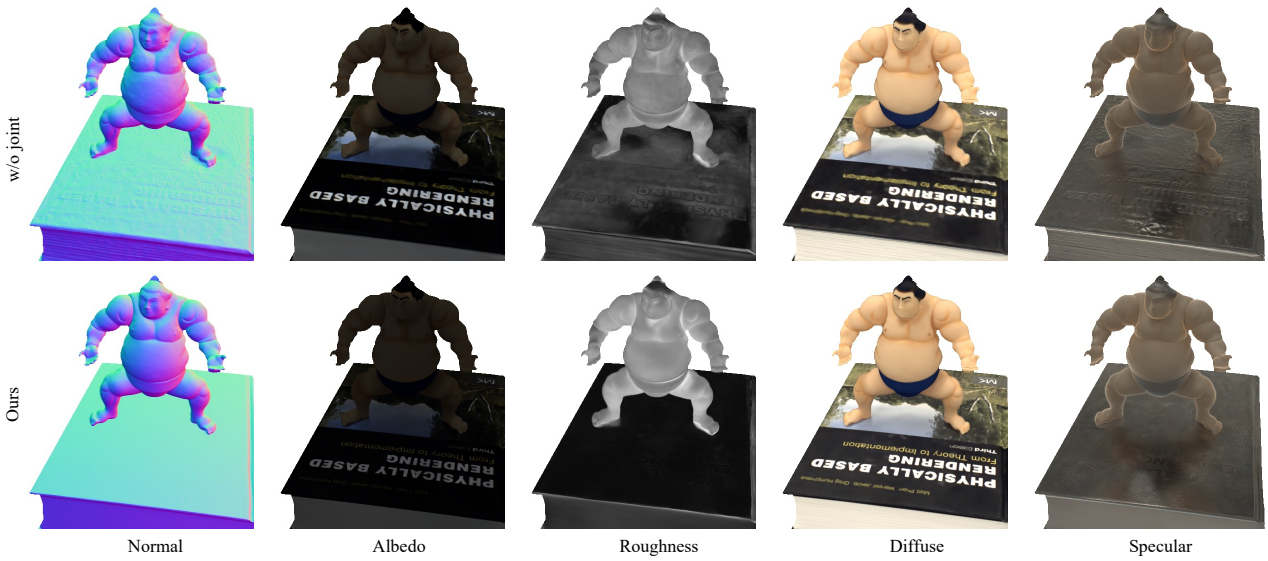

Figure 6. Comparison of with or without joint optimization. Because of the inaccurate geometry initialization in the first training stage, the results of roughness and specular intensity map are noisy.

$\mathbf{M}_{\text{spec}}$ can be written as follows:

$$\mathbf{M}_{\text{spec}} = k_s \frac{DG}{4\cos\theta_o}\mathbf{F}^R, \tag{22}$$

where k_s is the specular coefficient. GGX distribution function D [11] is defined as follows:

$$D = \frac{r^2}{\pi \cos^4\theta_h (r^2 + \tan^2\theta_h)^2}, \tag{23}$$

where r is the roughness, θ_h is the angle between the halfway vector and surface normal. Smith G function [3] is as follows:

$$G = \left(\frac{2}{1+\sqrt{1+r^2\tan^2\theta_i}}\right)\left(\frac{2}{1+\sqrt{1+r^2\tan^2\theta_o}}\right). \tag{24}$$

Fresnel reflection term $\mathbf{F}^R \in \mathbb{R}^{3\times3}$ is as follows:

$$\mathbf{F}^R = \begin{bmatrix} R^+ & R^- & 0 \\ R^- & R^+ & 0 \\ 0 & 0 & R^\times \cos\psi \end{bmatrix}, \tag{25}$$

where ψ is the phase shift, $\cos\psi = -1$ when the incident angle is less than the Brewster angle; otherwise, $\cos\psi = 1$. $R^+ = (R^\perp + R^\parallel)/2$, $R^- = (R^\perp - R^\parallel)/2$, and $R^\times = \sqrt{R^\perp R^\parallel}$. $R^\perp$ is the perpendicular term of the reflection coefficient:

$$R^\perp = \left(\frac{\cos\theta_o - \sqrt{\eta^2 - \sin^2\theta_o}}{\cos\theta_o + \sqrt{\eta^2 - \sin^2\theta_o}}\right)^2, \tag{26}$$

and $R^\parallel$ is the parallel term of the reflection coefficient:

$$R^\parallel = \left(\frac{\eta^2\cos\theta_o - \sqrt{\eta^2 - \sin^2\theta_o}}{\eta^2\cos\theta_o + \sqrt{\eta^2 - \sin^2\theta_o}}\right)^2. \tag{27}$$

Then Eq. 12 can be rewritten as follows:

$$\mathbf{s}^{\text{cam}}_{\text{spec}} = \int_\Omega k_s \frac{DG}{4\cos\theta_o} \begin{bmatrix} R^\times & R^- & 0 \\ R^-\cos 2\phi_{\text{spec}} & R^+\cos 2\phi_{\text{spec}} & R^\times \sin 2\phi_{\text{spec}}\cos\psi \\ -R^-\sin 2\phi_{\text{spec}} & -R^+\sin 2\phi_{\text{spec}} & R^\times\cos 2\phi_{\text{spec}}\cos\psi \end{bmatrix} \cdot \mathbf{s}^{\text{r}}_{\text{spec}}\, d\boldsymbol{\omega_i}, \tag{28}$$

where ϕ_{spec} is the rotation angle from the reference frame of $\mathbf{M}_{\text{spec}}$ to the camera axis. Similar to the diffuse term, Eq. 28 also needs to be repeated three times with separate specular coefficients when rendering RGB images.

E.2. Unpolarized version

We introduce the unpolarized version of baek pBRDF [2] used in the ablation study. The diffuse $\mathbf{i}^{\text{cam}}_{\text{diff}}$ and specular $\mathbf{i}^{\text{cam}}_{\text{spec}}$ components are as follows:

$$\mathbf{i}^{\text{cam}}_{\text{diff}} = \int_\Omega \frac{\boldsymbol{\rho}}{\pi} T_o^+ T_i^+ \mathbf{i}^{\text{in}} \cos\theta_i \, d\boldsymbol{\omega_i}, \tag{29}$$

$$\mathbf{i}^{\text{cam}}_{\text{spec}} = \int_\Omega \mathbf{k_s} \frac{DGR^\times}{4\cos\theta_o} \mathbf{i}^{\text{in}} \, d\boldsymbol{\omega_i}, \tag{30}$$

where $\mathbf{i}^{\text{in}}$ is the unpolarized incident light intensity.

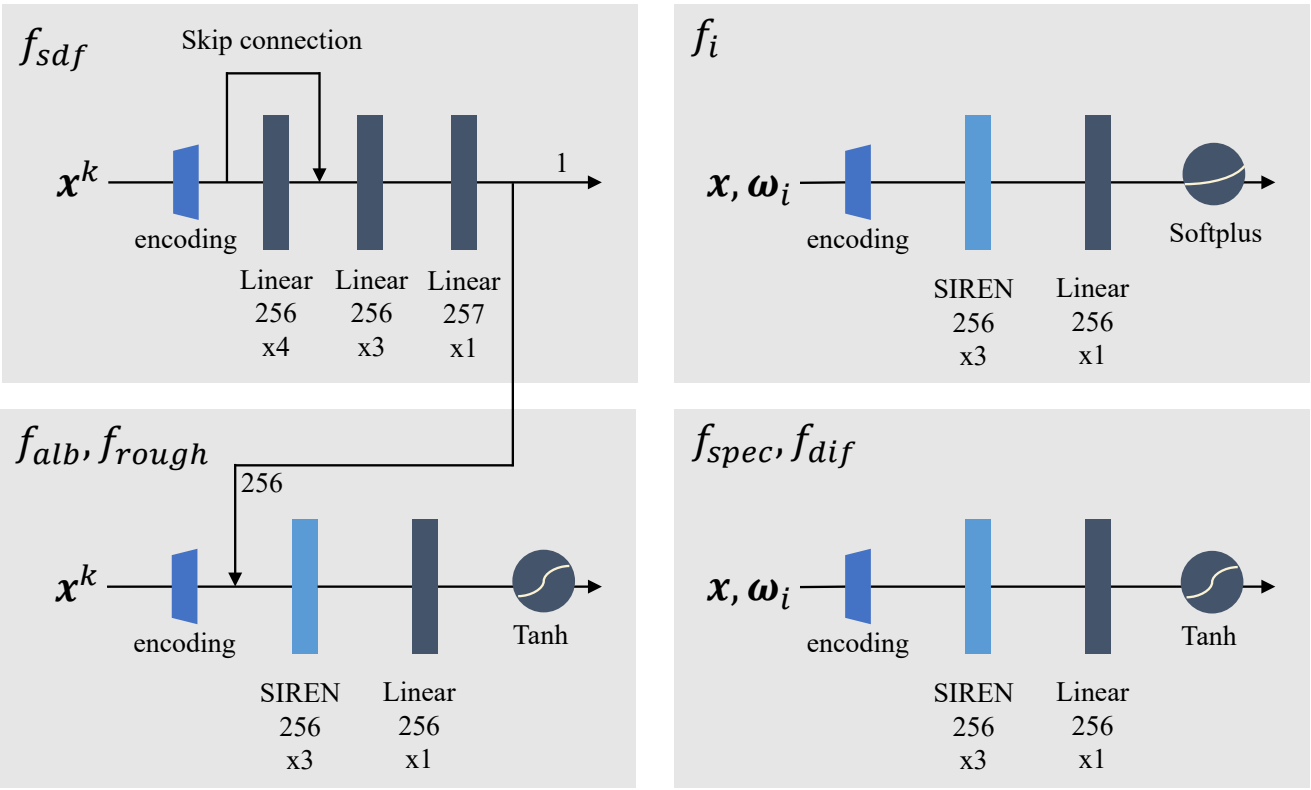

Figure 7. Network architecture. The SIREN layer is directly taken from [10].

F. Implementation Details

F.1. Ray marching algorithm

As briefly mentioned in the main paper, the position of the interaction point is calculated using a simplified ray marching algorithm. Given the ray origin r_o and the direction r_d, we iteratively march the ray to compute the interaction point $\mathbf{x}$ using Algo. 1. The number of steps is hard coded as 100 for both training and inference.

Algorithm 1 Ray Marching Algorithm

$t \leftarrow 0$
for step $= 1$ **to** max_step **do**
  $\mathbf{x} \leftarrow r_o + t \cdot r_d$
  $t \leftarrow t + f_{\text{sdf}}(\mathbf{x})$
end for
$\mathbf{x} \leftarrow r_o + t \cdot r_d$
return $\mathbf{x}$

F.2. Network architecture

Please refer to Fig. 7 for the architecture of our networks. We set the dimension of all positional encoding [6] to 6. We have in total six neural networks. The signed distance network f_{sdf} is directly taken from VolSDF [12]. The left networks are modified from NeILF++ [15]. The output dimensions for f_{alb}, f_{rough}, f_i, f_{spec}, and f_{dif} are 3, 1, 3, 6, 3, separately.

F.3. Loss function

Let $\mathbf{s}^{\hat{\text{cam}}}$ be the GT Stokes vectors, we compute the L_1 loss for the re-rendered Stokes vectors $\mathbf{s}^{\text{cam}}$:

$$L_1 = \frac{1}{B} \sum_B |\mathbf{s}^{\hat{\text{cam}}} - \mathbf{s}^{\text{cam}}|, \tag{31}$$

where B is the batch size. Following IDR [13], we also compute a Eikonal regularization:

$$L_{\text{Eik}} = \mathbb{E}_{\mathbf{x}}(||\nabla_{\mathbf{x}} f_{\text{sdf}}(\mathbf{x})|| - 1)^2. \tag{32}$$

The final loss is a linear combination of the above two losses:

$$L_{\text{total}} = \lambda_1 L_1 + \lambda_{\text{Eik}} L_{\text{Eik}}. \tag{33}$$

In practice, we set $\lambda_1 = 1.0$ and $\lambda_{\text{Eik}} = 0.1$.

F.4. Training details

Considering that all the pixels from the foreground of all the training images are trained as one epoch, we trained

the first stage (geometry initialization) for 20 epochs, the second stage (material and lighting initialization) for 20 epochs, and the third stage (joint optimization) for 60 epochs. The batch size was set to 2,048 and we used Adam [5] optimizer with a learning rate set to 5e-4 and decayed exponentially to 5e-5. For the joint optimization stage, We clipped the gradient norm of the signed distance field net f_{sdf} with the maximum norm set to 0.1. The number of samples along a ray is set to 64. The number of sampled incident rays of each interaction point is 128.

The time cost of training is strongly affected by the ratio between foreground and background. The average training time of our dataset is roughly 1 day for the geometry initialization, 1 day for the material and lighting initialization stage, and 3 days for the joint optimization stage on a single Nvidia V-100 GPU.

References

[1] Adobe Inc. Adobe photoshop. 3

[2] Seung-Hwan Baek, Daniel S Jeon, Xin Tong, and Min H Kim. Simultaneous acquisition of polarimetric svbrdf and normals. *ACM TOG*, 37(6):268–1, 2018. 1, 3, 5

[3] Eric Heitz. Understanding the masking-shadowing function in microfacet-based brdfs. *Journal of Computer Graphics Techniques*, 3(2):32–91, 2014. 5

[4] Wenzel Jakob, Sébastien Speierer, Nicolas Roussel, and Delio Vicini. Dr.jit: A just-in-time compiler for differentiable rendering. *ACM TOG*, 41(4), 2022. 1, 2

[5] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 7

[6] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *ECCV*, pages 405–421, 2020. 6

[7] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In *NeurIPS*, pages 8024–8035. 2019. 1

[8] Johannes Lutz Schönberger and Jan-Michael Frahm. Structure-from-motion revisited. In *CVPR*, pages 4104–4113, 2016. 2, 3

[9] Johannes Lutz Schönberger, Enliang Zheng, Marc Pollefeys, and Jan-Michael Frahm. Pixelwise view selection for unstructured multi-view stereo. In *ECCV*, pages 501–518, 2016. 2, 3

[10] Vincent Sitzmann, Julien Martel, Alexander Bergman, David Lindell, and Gordon Wetzstein. Implicit neural representations with periodic activation functions. *NeurIPS*, 33:7462–7473, 2020. 6

[11] Bruce Walter, Stephen R Marschner, Hongsong Li, and Kenneth E Torrance. Microfacet models for refraction through rough surfaces. In *Proceedings of the 18th Eurographics conference on Rendering Techniques*, pages 195–206, 2007. 5

[12] Lior Yariv, Jiatao Gu, Yoni Kasten, and Yaron Lipman. Volume rendering of neural implicit surfaces. *NeurIPS*, 34:4805–4815, 2021. 6

[13] Lior Yariv, Yoni Kasten, Dror Moran, Meirav Galun, Matan Atzmon, Basri Ronen, and Yaron Lipman. Multiview neural surface reconstruction by disentangling geometry and appearance. *NeurIPS*, 33:2492–2502, 2020. 6

[14] Masakazu Yoshimura, Junji Otsuka, Atsushi Irie, and Takeshi Ohashi. Rawgment: noise-accounted raw augmentation enables recognition in a wide variety of environments. In *CVPR*, pages 14007–14017, 2023. 3

[15] 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 *ICCV*, pages 3601–3610, 2023. 6