A PRELIMINARIES

Volume rendering. The radiance C of the pixel corresponding to a given ray $\boldsymbol{r}(t) = \boldsymbol{o} + t\boldsymbol{d}$ at the origin $\boldsymbol{o} \in \mathbb{R}^3$ towards direction $\boldsymbol{d} \in \mathbb{S}^2$ is calculated using the volume rendering equation, which involves an integral along the ray with boundaries t_n and t_f (t_n and t_f are parameters to define the near and far clipping plane). This calculation requires the knowledge of the volume density σ and directional color $\boldsymbol{c}$ for each point within the volume.

$$C(\boldsymbol{r}) = \int_{t_n}^{t_f} T(t)\sigma(\boldsymbol{r}(t))\mathbf{c}(\boldsymbol{r}(t), \boldsymbol{d})dt \tag{14}$$

The volume density σ is used to calculate the accumulated transmittance $T(t)$:

$$T(t) = \exp\left(-\int_{t_n}^{t_f} \sigma(\boldsymbol{r}_s)ds\right) \tag{15}$$

It is then used to compute a weighting function $w(t) = T(t)\sigma(\boldsymbol{r}(t))$ to weigh the sampled colors along the ray $\boldsymbol{r}(t)$ to integrate into radiance $C(\boldsymbol{r})$.

Surface rendering. The radiance $L_o(\boldsymbol{x}, \boldsymbol{\omega}_o)$ reflected from a surface point $\boldsymbol{x}$ in direction $\boldsymbol{\omega}_o = -\boldsymbol{d}$ is an integral of bidirectional reflectance distribution function (BRDF) and illumination over half sphere Ω, centered at normal $\boldsymbol{n}$ of the surface point $\boldsymbol{x}$:

$$L_o(\boldsymbol{x}, \boldsymbol{\omega}_o) = \int_\Omega L_i(\boldsymbol{x}, \boldsymbol{\omega}_i) f_r(\boldsymbol{x}, \boldsymbol{\omega}_i, \boldsymbol{\omega}_o)(\boldsymbol{\omega}_i \cdot \boldsymbol{n})\mathrm{d}\boldsymbol{\omega}_i \tag{16}$$

where $L_i(\boldsymbol{x}, \boldsymbol{\omega}_i)$ is the illumination on $\boldsymbol{x}$ from the incoming light direction $\boldsymbol{\omega}_i$, and f_r is BRDF, which is the proportion of light reflected from direction $\boldsymbol{\omega}_i$ towards direction $\boldsymbol{\omega}_o$ at the point $\boldsymbol{x}$.

B IMPLEMENTATION DETAILS

Our full model is composed of several MLP networks, each one of them having a width of 256 hidden units unless otherwise stated. In Stage 1, the SDF network S_θ is composed of 8 layers and includes a skip connection at the 4-th layer, similar to NeuS [Wang et al. 2021]. The input 3D coordinate $\boldsymbol{x}$ is encoded using positional encoding with 6 frequency scales. The diffuse color network M_d utilizes a 4-layer MLP, while the input surface normal $\boldsymbol{n}$ is positional-encoded using 4 scales. For the specular color network M_s, a 4-layer MLP is employed, and the reflection direction $\boldsymbol{\omega}_r$ is also positional-encoded using 4 frequency scales. In the first stage, we exclusively focus on decomposing the highlight (largely white) areas. To reduce the complexity of considering color, we assume that the specular radiance is in grayscale and only consider changes in brightness. We can incorporate color information in later stages to obtain a more detailed specular reflection model. Like NeuS, the background is modeled by NeRF++.

In Stage 2, the light visibility network M_v has 4 layers. To better encode the input 3D coordinate $\boldsymbol{x}$, positional encoding with 10 frequency scales is utilized. The input view direction ω_i is also positional-encoded using 4 scales. The indirect light network M_{ind} in stage 2 comprises 4 layers.

In stage 3, the encoder part of the BRDF network consists of 4 layers, and the input 3D coordinate is positional-encoded using 10 scales. The output latent vector $\boldsymbol{z}$ has 32 dimensions, and we impose a sparsity constraint on the latent code $\boldsymbol{z}$, following IndiSG [Zhang et al. 2022b]. The decoder part of the BRDF network is a 2-layer MLP with a width of 128, and the output has 4 dimensions, including the diffuse albedo $\boldsymbol{d}_a \in \mathbb{R}^3$ and roughness $r \in \mathbb{R}$. Finally, the specular albedo network M_{sa} uses a 4-layer MLP, where the input 3D coordinate $\boldsymbol{x}$ is positional-encoded using 10 scales, and the input reflection direction $\boldsymbol{\omega}_r$ is positional-encoded using 4 scales.

The learning rate for all three stages begins with a linear warm-up from 0 to 5×10^{-4} during the first 5K iterations. It is controlled by the cosine decay schedule until it reaches the minimum learning rate of 2.5×10^{-5}, which is similar to NeuS. The weights λ_{sur} for the surface color loss are set for 0.1, 0.6, 0.6, and 0.01 for DTU, SK3D, Shiny, and the IndiSG dataset, respectively. For all datasets, the Fresnel value f in the rendering equation is set to 0.02. We train our model for 300K iterations in the first stage, which takes 11 hours in total. For the second and third stages, we train for 40K iterations, taking around 1 hour each. The training was performed on a single NVIDIA RTX 4090 GPU.

C TRAINING STRATEGIES OF STAGE 1

In our training process, we define three loss functions, namely volume radiance loss $\mathcal{L}_{\text{vol}}$, surface radiance loss $\mathcal{L}_{\text{sur}}$, and regularization loss $\mathcal{L}_{\text{reg}}$. The volume radiance loss $\mathcal{L}_{\text{vol}}$ is measured by calculating the $\mathcal{L}_1$ distance between the ground truth colors C^{gt} and the volume radiances C^{vol} of a subset of rays $\mathcal{R}$, which is defined as follows.

$$\mathcal{L}_{\text{vol}} = \frac{1}{|\mathcal{R}|}\sum_{\boldsymbol{r}\in\mathcal{R}} \|C^{\text{vol}}_{\boldsymbol{r}} - C^{\text{gt}}_{\boldsymbol{r}}\|_1 \tag{17}$$

The surface radiance loss $\mathcal{L}_{\text{sur}}$ is measured by calculating the $\mathcal{L}_1$ distance between the ground truth colors C^{gt} and the surface radiances C^{sur}. During the training process, only a few rays have intersection points with the surface. We only care about the set of selected rays $\mathcal{R}'$, which satisfies the condition that each ray exists point whose SDF value is less than zero and not the first sampled point. The loss is defined as follows.

$$\mathcal{L}_{\text{sur}} = \frac{1}{|\mathcal{R}'|}\sum_{\boldsymbol{r}\in\mathcal{R}'} \|C^{\text{sur}}_{\boldsymbol{r}} - C^{\text{gt}}_{\boldsymbol{r}}\|_1 \tag{18}$$

$\mathcal{L}_{\text{reg}}$ is an Eikonal loss term on the sampled points. Eikonal loss is a regularization loss applied to a set of sampling points X, which is used to constrain the noise in signed distance function (SDF) generation.

$$\mathcal{L}_{\text{reg}} = \frac{1}{|\mathcal{X}|}\sum_{\boldsymbol{x}\in\mathcal{X}} (\|\nabla \mathsf{S}_\theta(\boldsymbol{x})\|_2 - 1)^2 \tag{19}$$

We use weights λ_{sur} and λ_{reg} to balance the impact of these three losses. The overall training weights are as follows.

$$\mathcal{L} = \mathcal{L}_{\text{vol}} + \lambda_{\text{sur}}\mathcal{L}_{\text{sur}} + \lambda_{\text{reg}}\mathcal{L}_{\text{reg}} \tag{20}$$

D DETAILS OF STAGE 2

At this stage, we focus on predicting the lighting visibility and indirect illumination of a surface point $\boldsymbol{x}$ under different incoming light direction $\boldsymbol{\omega}_i$ using the SDF in the first stage. Therefore, we need first to calculate the position of the surface point $\boldsymbol{x}$. In stage one, we have calculated two sampling points $\boldsymbol{r}(t_{i'-1}), \boldsymbol{r}(t'_i)$ near the surface. As Geo-NeuS [Fu et al. 2022], we weigh these two sampling

Table 4. Overview of the capabilities of the recent inverse rendering methods.

Method	Explicit surface extraction	Diffuse/specular color decomposition	Illumination reconstruction	Materials reconstruction	Handles glossy surfaces	No Extra Data?	Code available	Venue
NeRV [Srinivasan et al. 2021]	✗	✓	✓	✓	✗	✓	[to appear]	CVPR 2021
PhySG [Zhang et al. 2021a]	✓	✓	✓	✓	✓	object masks	✓	CVPR 2021
NeuS [Wang et al. 2021]	✓	✗	✗	✗	✗	✓	✓	NeurIPS 2021
NeRFactor [Zhang et al. 2021b]	✗	✓	✓	✓	✗	BRDF dataset	✓	SG Asia 2021
Ref-NeRF [Verbin et al. 2022]	✗	✓	✗	✗	✓	✓	✓	CVPR 2022
NVDiffrec [Munkberg et al. 2022]	✓	✓	✓	✓	✗	object masks	✓	CVPR 2022
IndiSG [Zhang et al. 2022b]	✓	✓	✓	✓	✗	object masks	✓	CVPR 2022
Geo-NeuS [Fu et al. 2022]	✓	✗	✗	✗	✗	point clouds	✓	NeurIPS 2022
BakedSDF [Yariv et al. 2023]	✓	✓	✗	✗	✗	✓	✗	SG 2023
TensoIR [Jin et al. 2023]	✗	✓	✓	✓	✗	✓	✓	CVPR 2023
NeFII [Wu et al. 2023a]	✓	✓	✓	✓	✗	✓	✗	CVPR 2023
Ref-NeuS [Ge et al. 2023]	✓	✗	✗	✗	✓	✓	[to appear]	arXiv 2023/03
αSurf [Wu et al. 2023b]	✓	✗	✗	✗	✗	object masks	[to appear]	arXiv 2023/03
NeILF++ [Zhang et al. 2023b]	✓	✓	✓	✓	✗	✓	[to appear]	arXiv 2023/03
ENVIDR [Liang et al. 2023]	✓	✓	✓	✓	✓	BRDF dataset	[to appear]	arXiv 2023/03
NeMF [Zhang et al. 2023a]	✗	✓	✓	✓	✗	✓	✗	arXiv 2023/04
NeAI [Zhuang et al. 2023]	✗	✓	✓	✓	✓	✓	[to appear]	arXiv 2023/04
NeRO [Liu et al. 2023]	✓	✓	✓	✓	✓	✓	✓	arXiv 2023/05
Factored-NeuS(ours)	✓	✓	✓	✓	✓	✓	[in supp]	-

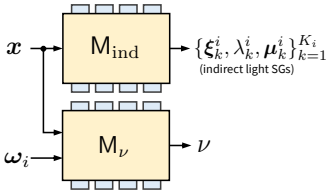

Fig. 11. Overview for Stage 2. $\boldsymbol{x}$ is a point on the surface. ω_i is the view direction. Indirect light and light visibility network M_{ind} and M_v produce their respective indirect light SGs and light visibility v.

points to obtain a surface point $\boldsymbol{x}$ as follows.

$$\boldsymbol{x} = \frac{\mathsf{S}_\theta(\boldsymbol{r}(t_{i'-1}))\boldsymbol{r}(t_i') - \mathsf{S}_\theta(\boldsymbol{r}(t_i'))\boldsymbol{r}(t_{i'-1})}{\mathsf{S}_\theta(\boldsymbol{r}(t_{i'-1})) - \mathsf{S}_\theta(\boldsymbol{r}(t_i'))} \tag{21}$$

Learning lighting visibility. Visibility is an important factor in shadow computation. It calculates the visibility of the current surface point $\boldsymbol{x}$ in the direction of the incoming light $\boldsymbol{\omega}_i$. Path tracing of the SDF is commonly used to obtain a binary visibility (0 or 1) as used in IndiSG [Zhang et al. 2022b], but this kind of visibility is not friendly to network learning. Inspired by NeRFactor [Zhang et al. 2021b], we propose to use an integral representation with the continuous weight function $w(t)$ (from 0 to 1) for the SDF to express light visibility. Specifically, we establish a neural network $\mathsf{M}_\nu : (\boldsymbol{x}, \boldsymbol{\omega}_i) \mapsto \nu$, that maps the surface point $\boldsymbol{x}$ and incoming light direction $\boldsymbol{\omega}_i$ to visibility, and the ground truth value of light visibility is obtained by integrating the weights w_i of the SDF of sampling points along the incoming light direction and can be expressed as follows.

$$\nu^{gt} = 1 - \sum_{i=1}^{n} w_i \tag{22}$$

The weights of the light visibility network are optimized by minimizing the loss between the calculated ground truth values and the predicted values of a set of sampled incoming light directions $\Omega_i \subset \mathbb{S}^2$. This pre-integrated technique can reduce the computational burden caused by the integration for subsequent training.

$$\mathcal{L}_{\text{vis}} = \frac{1}{|\Omega_i|} \sum_{\boldsymbol{\omega} \in \Omega_i} \|\nu_{\boldsymbol{\omega}} - \nu_{\boldsymbol{\omega}}^{\text{gt}}\|_1 \tag{23}$$

Learning indirect illumination. Indirect illumination refers to the light that is reflected or emitted from surfaces in a scene and then illuminates other surfaces, rather than directly coming from a light source, which contributes to the realism of rendered images. Following IndiSG [Zhang et al. 2022b], we parameterize indirect illumination $I(\boldsymbol{x}, \boldsymbol{\omega}_i)$ via $K_i = 24$ Spherical Gaussians (SGs) as follows.

$$I(\boldsymbol{x}, \boldsymbol{\omega}_i) = \sum_{k=1}^{K_i} I_k(\boldsymbol{\omega}_i \mid \boldsymbol{\xi}_k^i(\boldsymbol{x}), \lambda_k^i(\boldsymbol{x}), \boldsymbol{\mu}_k^i(\boldsymbol{x})) \tag{24}$$

where $\boldsymbol{\xi}_k^i(\boldsymbol{x}) \in \mathbb{S}^2$, $\lambda_k^i(\boldsymbol{x}) \in \mathbb{R}_+$, and $\boldsymbol{\mu}_k^i(\boldsymbol{x}) \in \mathbb{R}^3$ are the lobe axis, sharpness, and amplitude of the k-th Spherical Gaussian, respectively. For this, we train a network $\mathsf{M}_{\text{ind}} : \boldsymbol{x} \mapsto \{\boldsymbol{\xi}_k^i, \lambda_k^i, \boldsymbol{\mu}_k^i\}_{k=1}^{K_i}$ that maps the surface point $\boldsymbol{x}$ to the parameters of indirect light SGs. Similar to learning visibility, we randomly sample several directions $\boldsymbol{\omega}_i$ from the surface point $\boldsymbol{x}$ to obtain (pseudo) ground truth $I^{\text{gt}}(\boldsymbol{x}, \boldsymbol{\omega}_i)$. Some of these rays have intersections $\boldsymbol{x}'$ with other surfaces, thus, $\boldsymbol{\omega}_i$ is the direction pointing from $\boldsymbol{x}$ to $\boldsymbol{x}'$. We query our proposed color network M_c to get the (pseudo) ground truth indirect radiance $I^{\text{gt}}(\boldsymbol{x}, \boldsymbol{\omega}_i)$ as follows.

$$I^{\text{gt}}(\boldsymbol{x}, \boldsymbol{\omega}_i) = \mathsf{M}_c(\boldsymbol{x}', \boldsymbol{n}', \boldsymbol{\omega}_i, \boldsymbol{v}_f) \tag{25}$$

where $\boldsymbol{n}'$ is the normal on the point $\boldsymbol{x}'$. We also use $\mathcal{L}_1$ loss to train the network.

$$\mathcal{L}_{\text{ind}} = \frac{1}{|M|} \sum_{m \in M} \|I(\boldsymbol{x}, \boldsymbol{\omega}_m) - I_m^{\text{gt}}(\boldsymbol{x}, \boldsymbol{\omega}_m)\|_1 \tag{26}$$

E DETAILS OF STAGE 3

The combination of light visibility and illumination SG is achieved by applying a ratio to the lobe amplitude of the output SG, while preserving the center position of the SG. We randomly sample $K_s = 32$ directions within the SG lobe and compute a weighted average of the visibility with different directions.

$$\begin{aligned} &v(\boldsymbol{x}, \boldsymbol{\omega}_i) \otimes E_k(\boldsymbol{\omega}_i \mid \boldsymbol{\xi}_k^e, \lambda_k^e, \boldsymbol{\mu}_k^e) \\ \approx &E_k(\boldsymbol{\omega}_i \mid \boldsymbol{\xi}_k^e, \lambda_k^e, \frac{\sum_{s=1}^{K_s} E_k(\boldsymbol{\omega}_s) v(\boldsymbol{x}, \boldsymbol{\omega}_s)}{\sum_{s=1}^{K_s} E_k(\boldsymbol{\omega}_s)} \boldsymbol{\mu}_k^e) \end{aligned} \tag{27}$$

Here, we offer intuitive explanations for why the incorporation of specular albedo in the model results in a decrease in lighting prediction. The increase in the model's complexity is the primary reason. Specular albedo introduces a more detailed modeling of surface reflection characteristics, requiring additional parameters and learning capacity. This raises the difficulty of training the model, potentially resulting in overfitting or training instability, thereby affecting the accurate prediction of lighting.

F ADDITIONAL RESULTS

In order to have a fair comparison with Geo-NeuS on the DTU dataset, we incorporate the components of Geo-NeuS based on the additional data (the point clouds from SfM and image pairs) used in Geo-NeuS into our method. As shown in Tab. 5, our approach can further enhance the surface reconstruction quality on datasets where highlights are less pronounced.

Table 5. Quantitative results in terms of Chamfer distance on DTU [Jensen et al. 2014].

	DTU 63	DTU 97	DTU 110	**Mean**
Geo-NeuS [Fu et al. 2022]	0.96	0.91	0.70	0.86
Factored-NeuS (ours)	0.99	1.15	0.89	1.01
Factored-NeuS (ours w/ Geo)	**0.95**	**0.89**	**0.69**	**0.84**

We conduct another experiment to compare our modeling approach with Ref-NeRF and S^3-NeRF [Yang et al. 2022]. The experimental quantitative and qualitative results are shown in Fig. 13. Ref-NeRF utilizes volume rendering colors for diffuse and specular components. If we directly combine SDF and the architecture of Ref-NeRF, it is challenging to eliminate the influence of highlights. Furthermore, if we applied the construction method of S^3-NeRF, which involves integrating surface rendering colors into volume rendering, to modify our model structure, we found that this modeling approach cannot address the issue of geometric concavity caused by highlights.

We conducted a more in-depth comparison of our method with the already published work NeRO. For DTU datasets, our findings demonstrate that NeRO performs less effectively than our approach on real datasets DTU as shown in Fig. 14. NeRO not only struggles to accurately restore detailed information but also fails to address the negative impact of partial highlights on the geometry. Moreover, the presence of shadows causes NeRO to mistakenly reconstruct shadowed areas as real objects and fill them in (bricks and skull models). In addition, our quantitative evaluation of Chamfer distance for the DTU dataset performs better as presented in Tab. 6. Furthermore, we extended our comparison to include new glossy datasets, Glossy-Blender dataset and Glossy-Real dataset in Fig. 15 and Tab. 7, where although NeRO performs better, but our method is also capable of mitigating the impact of highlights on geometry. Our method demonstrated comparable results to NeRO. Moreover, compared to NeuS, the results show a significant improvement. Note that the Glossy real datasets include bear, bunny, coral, vase. The rest of the others are in the glossy synthetic dataset.

For completeness, we visualize the decomposition of diffuse and specular in the first stage in Fig. 16. In the first stage, the decomposition of diffuse and specular is not a true BRDF model. This is because the MLP in the first stage is used solely for predicting the components of diffuse and specular reflection, rather than predicting material properties such as albedo and roughness. The decision to directly predict colors instead of material properties in the first stage serves two purposes: reducing model complexity by focusing on the direct prediction of specular reflection color, and optimizing geometry for better reconstruction. By decomposing highlights through the network in the first stage, surfaces with specular reflections can be reconstructed more effectively, demonstrated by the presence of flower pot ablation, and without encountering the concavity issues observed in other methods.

Additionally, in Fig. 17, Fig. 18, and Fig. 19, we presented all components, the rendering, albedo, roughness, diffuse color, specular color, light visibility, indirect light, and environment light results for the IndiSG, DTU and SK3D datasets, respectively. An interesting observation is that our reconstructed environment maps have the capability to represent multiple direct light illuminants, as demonstrated in the DTU dataset.

In Fig. 20, we additionally showcase the visualization results of relighting compared with the IndiSG method. IndiSG and ours yield different predictions for material, resulting in variations in the relighting results, but the relighting results generated by our method exhibit richer details. Our method demonstrates the practical utility employed in the relighting scenarios.

For chrome-like materials, We increase the Fresnel value to 0.75 in the rendering formula of stage 3 to test the impact of this operation. We show the results and their PSNR value in Fig. 21, we observed that increasing the Fresnel value indeed leads to better reconstruction of objects with chrome-like materials. For the Toaster model, we observed a significant improvement in PSNR with an increased Fresnel value. However, we also noticed that solely increasing the Fresnel value can result in the degradation of texture details. For instance, in the Coffee model, although the highlights on the spoon are better reconstructed, the text on the cup deteriorates. One of our future directions is to address this issue more effectively.

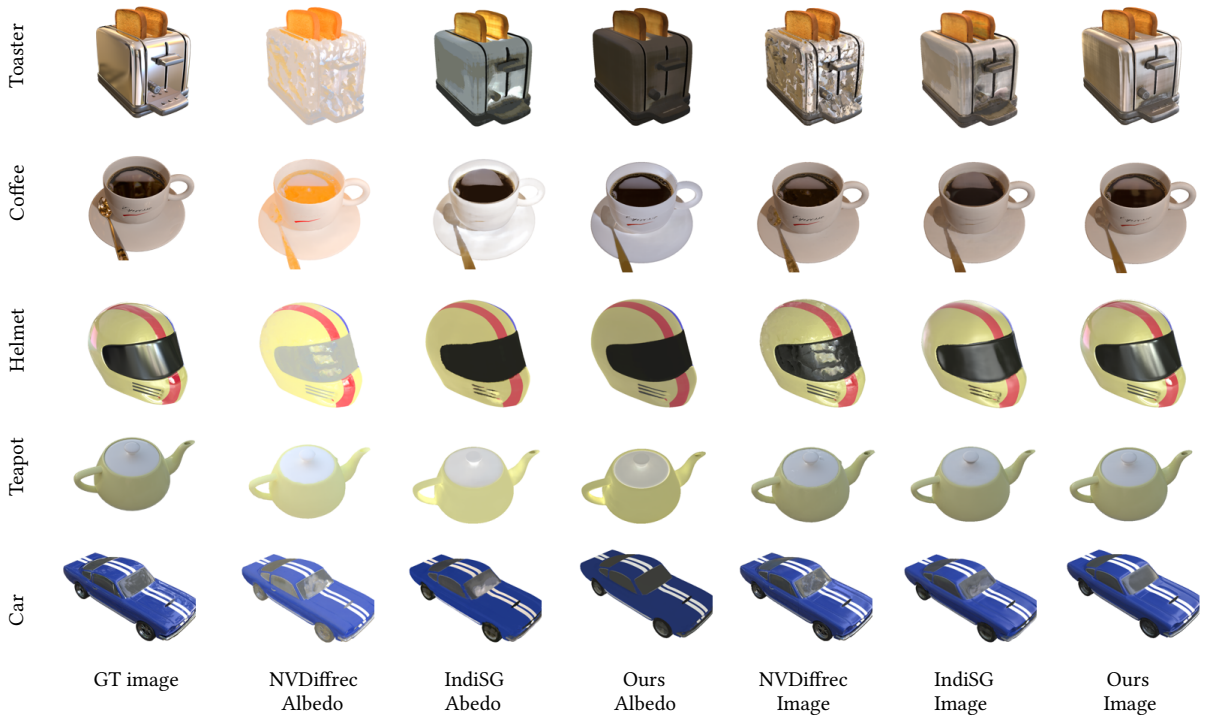

Fig. 12. Qualitative results for the Shiny dataset. Albedo refers to the diffuse albedo.

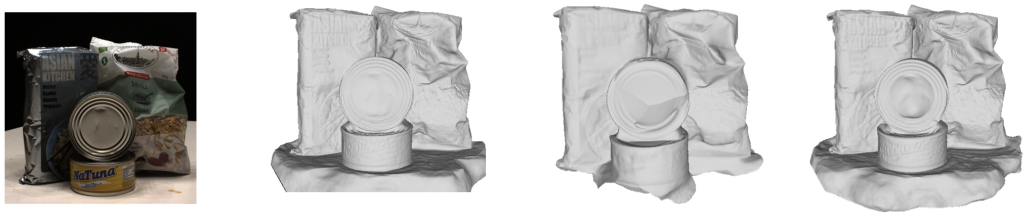

Fig. 13. Comparison with Ref-NeRF and S^3-NeRF.

Table 6. Comparison with NeRO and NeuS on DTU dataset.

DTU	24	37	40	55	63	65	69	83	97	105	106	110	114	118	122	mean
NeuS	1.00	1.37	0.93	0.43	1.01	0.65	**0.57**	1.48	1.21	0.83	**0.52**	1.14	**0.35**	0.49	0.54	0.83
NeRO	1.10	1.13	1.26	0.46	1.32	1.93	0.71	1.61	1.47	1.10	0.70	1.14	0.39	0.52	0.57	1.03
Ours	**0.82**	**1.05**	**0.85**	**0.40**	**0.99**	**0.59**	0.60	**1.44**	**1.15**	**0.81**	0.58	**0.89**	0.36	**0.44**	**0.46**	**0.76**

Table 7. Comparison with NeRO and NeuS on Glossy dataset.

Glossy	bear	bunny	coral	maneki	vase	angel	bell	cat	horse	luyu	potion	tbell	teapot	mean
NeuS	0.0074	0.0022	0.0016	0.0091	0.0101	0.0035	0.0146	0.0278	0.0053	0.0066	0.0393	0.0348	0.0546	0.0167
NeRO	**0.0033**	**0.0012**	**0.0014**	**0.0024**	**0.0011**	**0.0034**	**0.0032**	**0.0044**	**0.0049**	**0.0054**	**0.0053**	**0.0035**	**0.0037**	**0.0033**
Ours	0.0034	0.0017	**0.0014**	0.0027	0.0023	**0.0034**	0.0054	0.0059	0.0052	0.0060	0.0058	**0.0035**	0.0105	0.0044

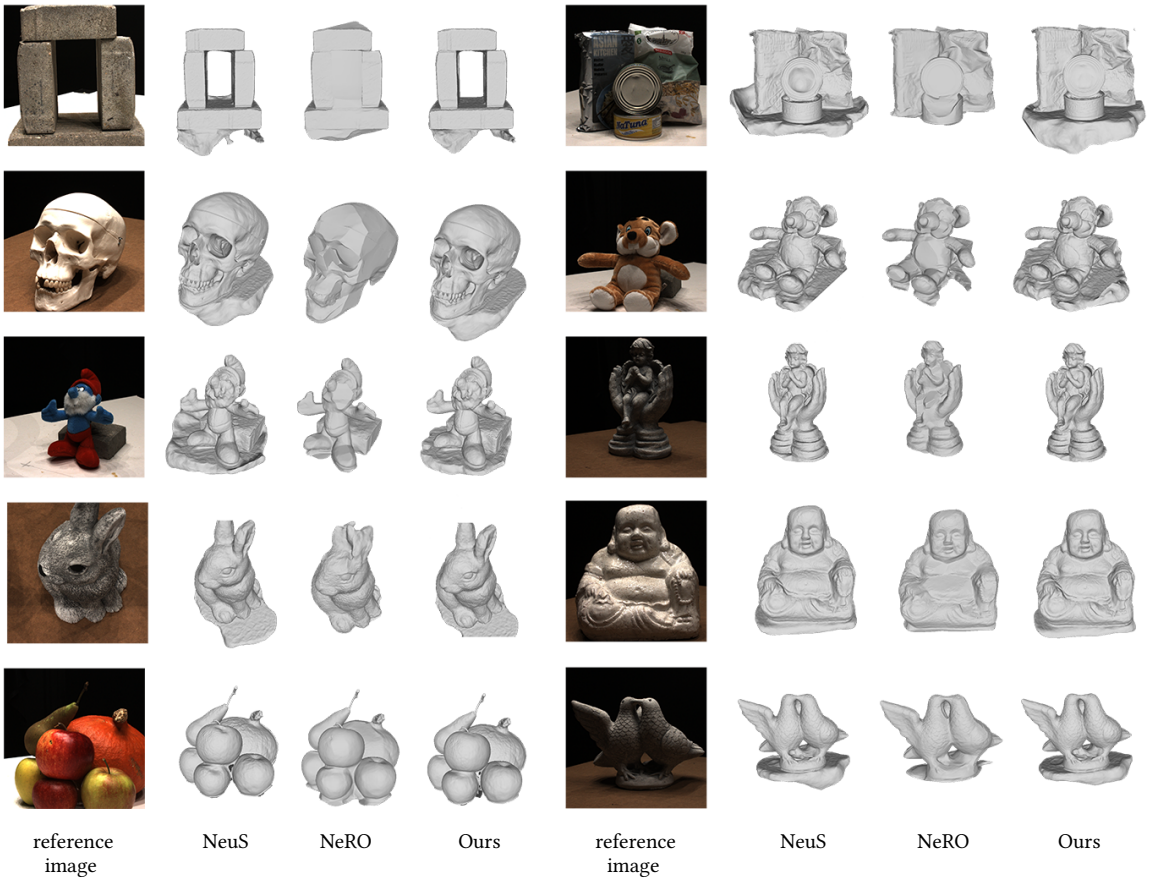

Fig. 14. Comparison with NeRO and NeuS on DTU dataset.

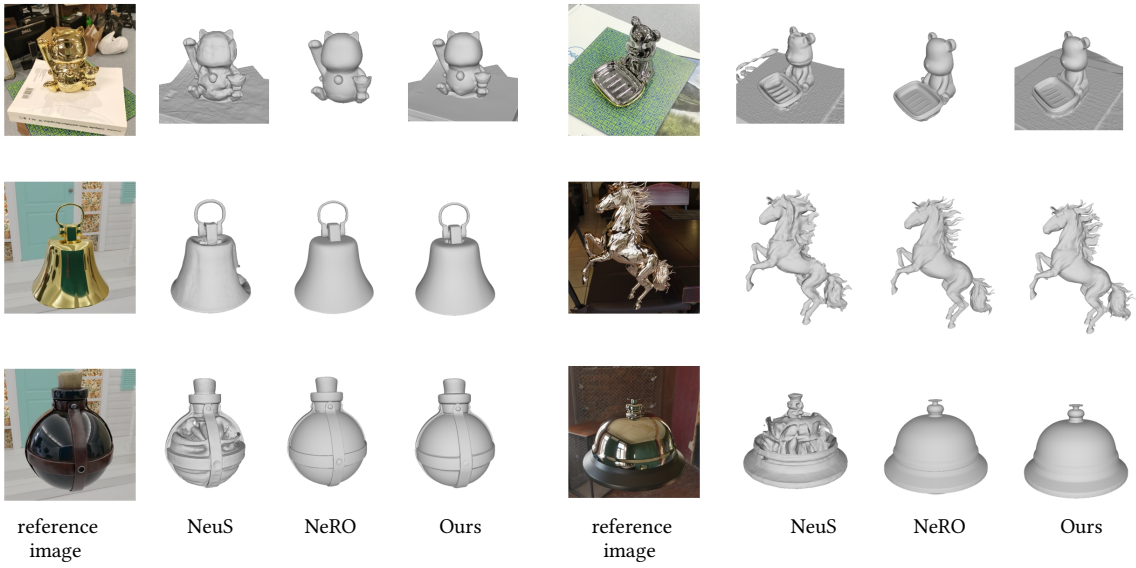

Fig. 15. Comparison with NeRO and NeuS on Glossy dataset.

Fig. 16. Diffuse and specular decomposition results in the first stage.

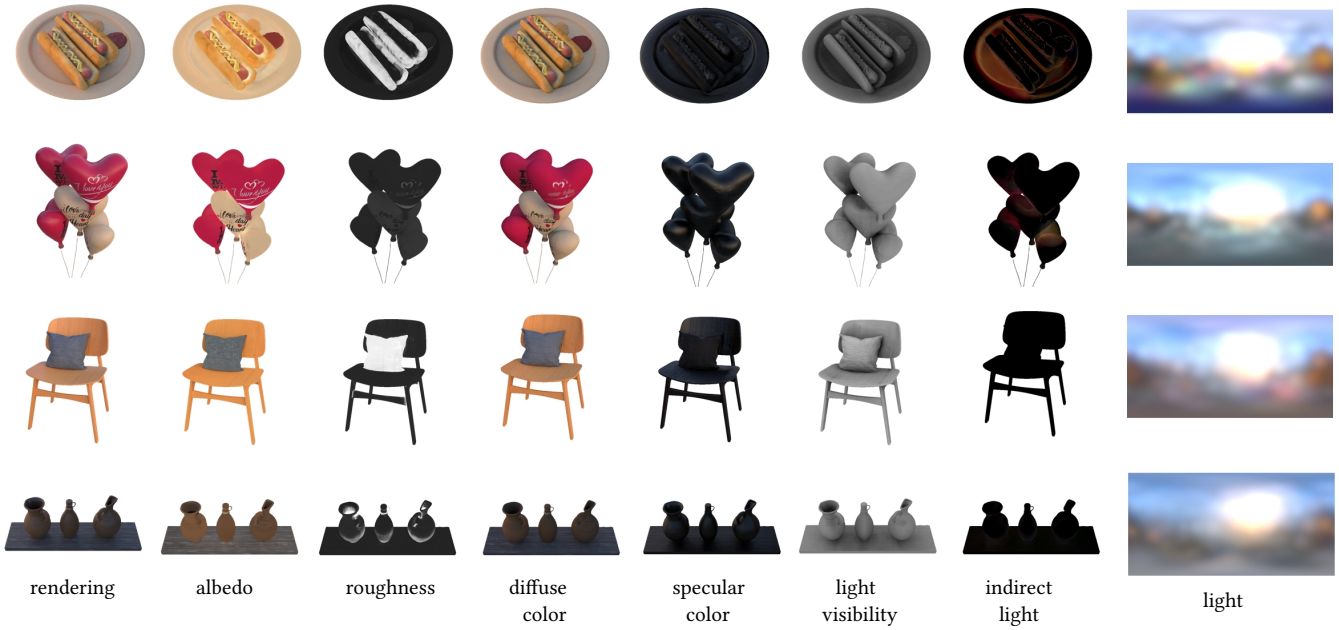

Fig. 17. Visualization of all components on IndiSG dataset.

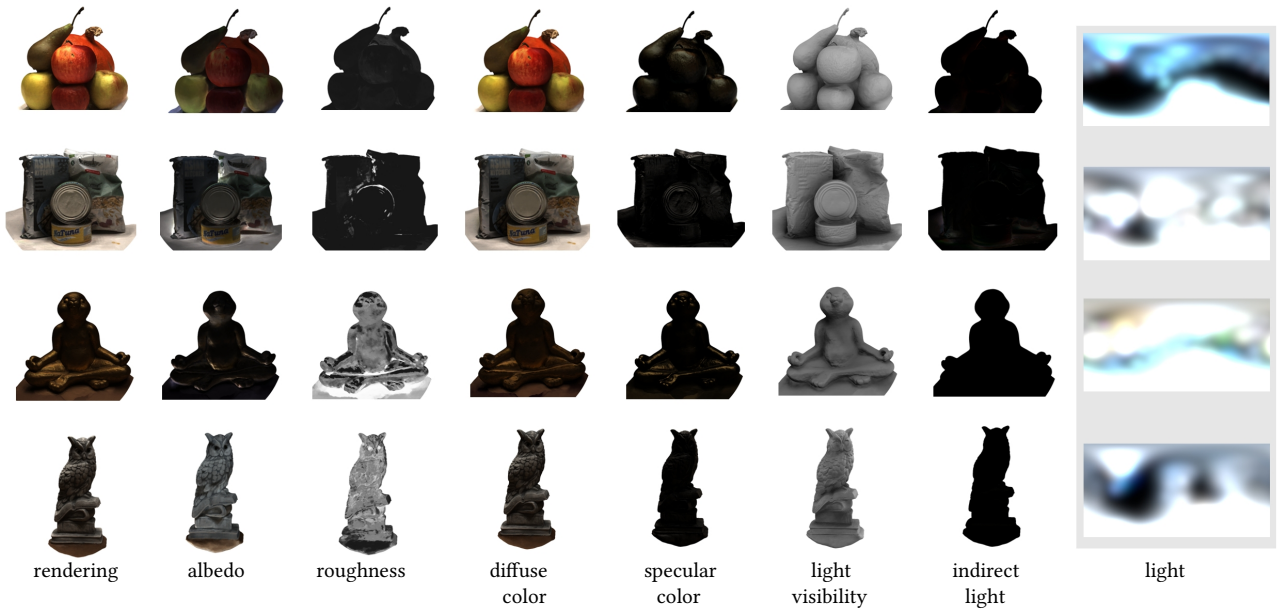

Fig. 18. Visualization of all components on DTU dataset.

Fig. 19. Visualization of all components on SK3D dataset.

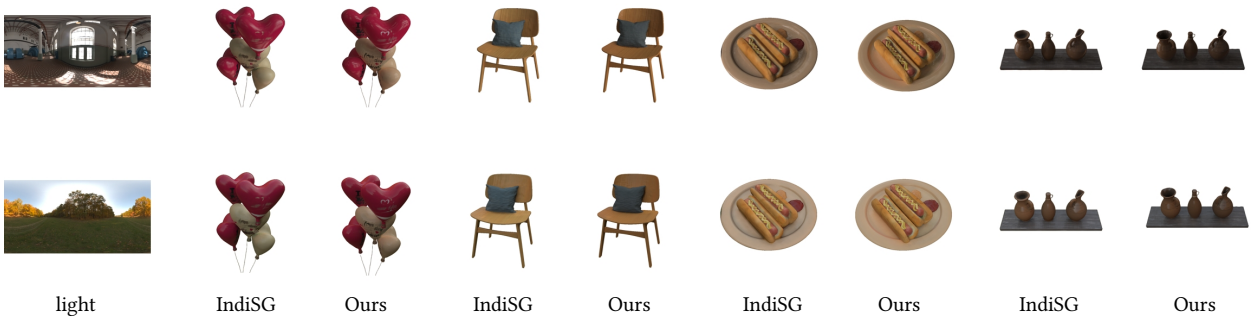

Fig. 20. Relighting comparison with IndiSG.

GT image | f=0.02 22.33 | f=0.75 24.31 | GT image | f=0.02 30.43 | f=0.75 30.18

Fig. 21. Adjusting Fresnel value to model chrome-like appearance.