Supplementary Material for NeRD: Neural Reflectance Decomposition from Image Collections

Mark Boss[1], Raphael Braun[1], Varun Jampani[2], Jonathan T. Barron[2], Ce Liu[2], Hendrik P.A. Lensch[1]

[1]University of Tübingen, [2]Google Research

Implementation details. The main network N_{θ_1}/N_{ϕ_1} uses 8 MLP layers with a feature dimension of 256 and ReLU activation. The input coordinate $\boldsymbol{x}$ is transformed by the Fourier output $\gamma(\boldsymbol{x})$ with 10 bands to 63 features. For the *sampling network*, the output from the main network is then transformed to the density σ with a single MLP layer without any activation. The flattened 192 SGs parameters Γ^j are compacted to 16 features using a fully connected layer (N_{θ_2}) without any activation. As the value range can be large in the real-world illuminations, the amplitudes α are normalized to $[0, 1]$. The main network output is then concatenated with the SGs embeddings and passed to the final prediction network. Here, an MLP with ReLU activation is first reducing the joined input to 128 features. The final color prediction c^j is handled in the last layer without activation and an output dimension of 3.

The *decomposition network* uses the output from N_{ϕ_1} and directly predicts the direct color d and the density in a layer without activation and 4 output dimensions (RGB+σ). The main network output is then passed to several ReLU activated layers which handle the BRDF compression N_{θ_2}. The feature outputs are as followed: 32, 16, 2 (no activation), 16, 16, 5 (no activation). The final five output dimensions correspond to the number of parameters of the BRDF model. The compressed embedding with two feature outputs is regularized with a $\mathcal{L}^2$ norm with a scale of 0.1 and further clipped to a value range of -40 to 40 to keep the value ranges in the beginning stable. Per batch, 1024 rays are cast into a single scene.

Loss and learning rate schedule. For adjusting the losses and learning rate during the training, we use exponential decay: $p(i; v, r, s) = vr^{\frac{i}{s}}$. The learning rate then uses $p(i; 0.000375, 0.1, 250000)$, the direct color d loss is faded out using $p(i; 1, 0.75, 1500)$ and the alpha loss is faded in using $p(i; 1, 0.9, 5000)$. During the first 1000 steps, we also do not optimize the SGs parameters and first use the white balancing only to adjust the mean environment strength, as this step also sets the illumination strength per image based on the exposure values.

Mesh extraction. The ability to extract a consistent textured mesh from NeRD after training is one of the key advantages of the decomposition approach and enables realtime rendering and relighting. This is not possible with NeRF-based approaches where the view-dependent appearance is directly baked into the volume.

We use the following four general steps to extract textured meshes:

1. A very dense point cloud representation of the surface is extracted. This step utilizes the same rendering functions used during training, which ensures that the resulting 3D coordinates are consistent with the training. To generate the rays for the rendering step, we sample the *decomposition network* for σ in a regular grid within the view volume determined by the view frustums of the cameras. We construct a discrete PDF from this grid, which is then sampled to generate about 10 million points where σ is high. The rays are constructed by following the normals at those points to get the ray-origins. We use the slightly jittered inverted normals as ray directions. See Fig. A1 a to c for visualizations of this step.

2. For meshing, we use the Open3D [5] implementation of the Poisson surface reconstruction algorithm [2] using the normals from NeRD. Before meshing, we perform two cleanup steps: First, we reject all points where the accumulated opacity along a ray is lower than 0.98. Secondly, we perform statistical outlier removal from Open3D. Those steps are visualized in Fig. A1 d and e

3. We use Blender's [1] *Smart UV Project* to get a simple UV-unwrapping for the mesh. Reducing the mesh resolution beforehand is an optional step that reduces the computational burden for using the mesh later. This is also done using Blender via *Decimate Geometry* or the *Voxel Remesher*.

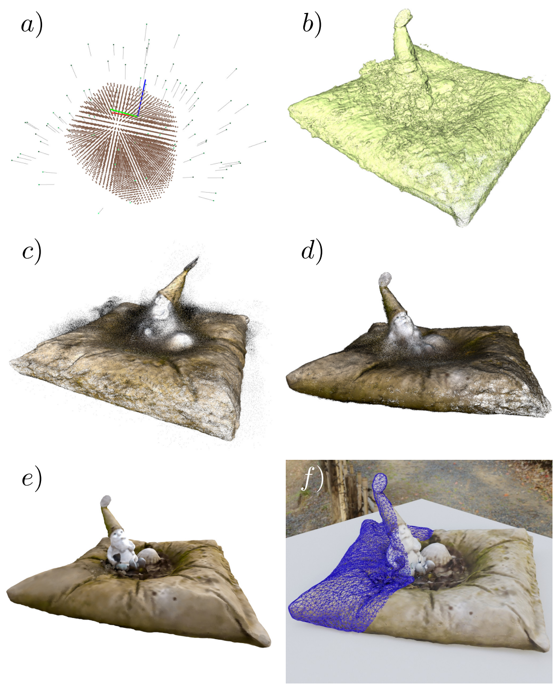


Figure A1: **Mesh Generation.** a) Cameras and view frustum. b) Points sampled where σ was high. c) Rendered point-cloud with basecolor. d) Outlier removal. e) Mesh with vertex-colors. f) Low res mesh with material textures.

Dataset	Resolution (W×H)	#Images	#Train	#Test
Globe	400 × 400	210	200	10
Car Wreck	400 × 400	210	200	10
Chair	400 × 400	210	200	10
Ethiopian Head	500 × 500	66	62	4
Gold Cape	456 × 456	119	111	8
Gnome	752 × 502	103	96	7
MotherChild	864 × 648	104	97	7

Table A1: **Dataset Overivew.** Overview about the resolution and number of images used for training.

4. We bake the surface coordinates and geometry normals into a floating-point texture of the desired resolution. The textures are generated by generating and rendering one ray per texel to look up the BRDF parameters and shading normals with NeRD. A result is show in Fig. A1 f.

Dataset details. In Table A1, we list the trained resolution, the number of total images, and the test train split for each dataset. Exemplary images of the real-world datasets are shown in Fig. A2.

Additional Results. In this section, we show more visual and qualitative results for our training scenes. First, we show the performance on our other real-world datasets (Gold Cape and Ethiopian Head). Samples are shown in

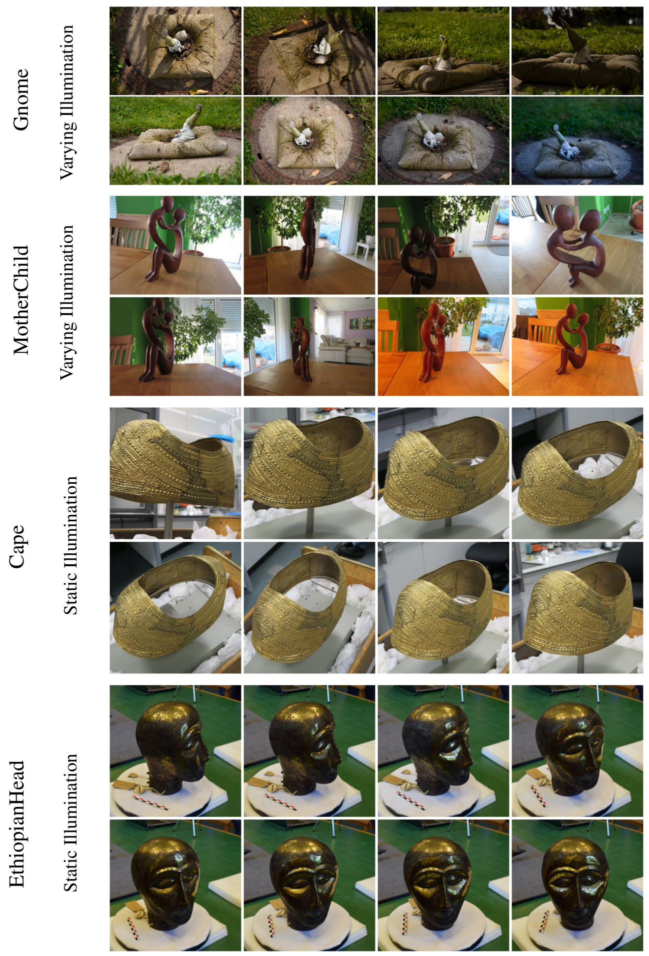


Figure A2: **Datasets.** Exemplary images of our real-world datasets. Notice the challenging environment illumination in the varying illumination scenes. The gnome dataset even features shadows from the environment.

Fig. A3. The details are preserved and apparent in our reconstructions. The reflective properties also match closely. We also want to highlight the prediction quality compared to NeRF [4] in Fig. A4. Here, especially in the scenes with varying illumination (Chair and Gnome), NeRF fails as expected. Our method decomposes the information, and after rendering the view, synthesis is close to ground truth. In the scenes with fixed illumination (Head and Cape), the performance between NeRF and our method is on par in most parts. The main difference in MSE is due to the baked-in highlights of NeRF. Our physically grounded design using rendering reduces these artifacts drastically. We also want to point out that relighting a scene is not possible in NeRF. Lastly, in Fig. A6 an additional example for the accuracy of the BRDF prediction is shown. In the scene, the BRDFs do not reproduce the input perfectly. However, the re-rendering shows a small error and is visually also close. As the opti-

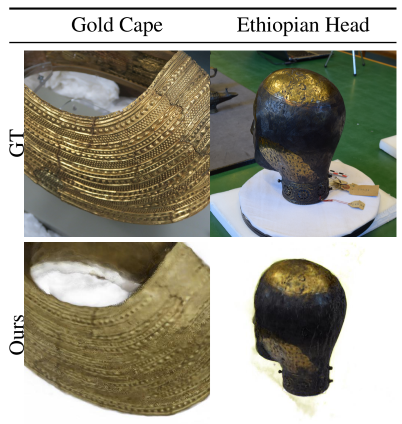


Figure A3: **Real World Novel View Synthesis.** Comparison on real-world samples with novel view synthesis on test dataset views.

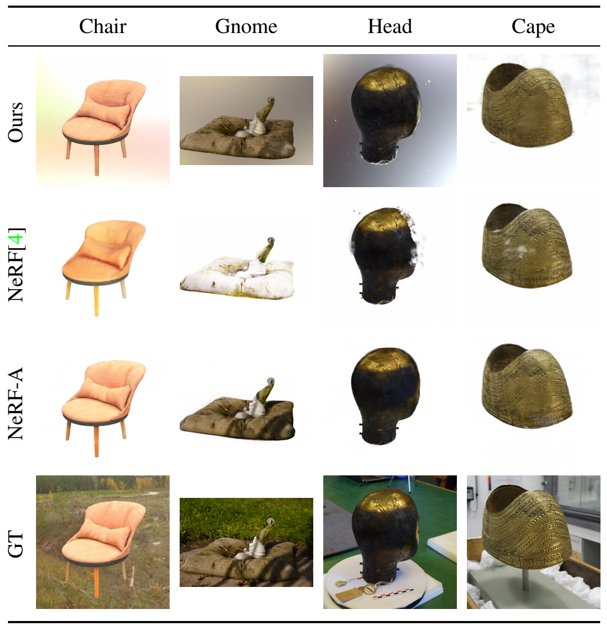
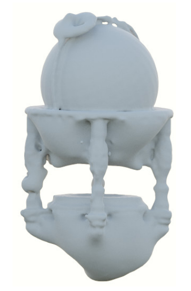


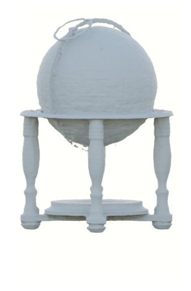
Figure A4: **Comparison with NeRF and NeRF-A.** Comparison with NeRF and NeRF-A on various scenes. Here, it is evident that NeRF fails as expected on scenes with varying illumination (Chair, Gnome).

mization is fully unconstrained, the decomposition found a solution that perfectly explains the input images.

Results from partial estimation techniques. In Fig. A5 we show a failure case of running COLMAP on data with varying illumination. An accurate surface normal is required for a correct BRDF estimation. If a pipeline is constructed where the first step is an independent geometry reconstruction, the method will fail. Another approach in a partial estimation is to decompose the BRDF for each im-



(a) **COLMAP Reconstruction.** COLMAP fails to recreate plausible geometry.



(b) **Ours.** Our reconstruction can handle this complex scene.

Figure A5: **Geometry reconstruction.** Comparison of a COLMAP reconstruction in our globe scene with varying illuminations.

age independently. Li *et al*. [3] is a method that decomposes objects illuminated by environment illumination only into diffuse and specular roughness. However, no novel views cannot be synthesized, and single image decomposition is highly ill-posed. In Fig. A7 results are shown. The diffuse parameter in our method is more consistent compared to Li *et al*., and it is apparent that Li *et al*. failed to factor out the illumination from the diffuse. However, the roughness is slightly better for Li *et al*. but is not as consistent, and the roughness is highly correlated with the texture of the globe, which is not correct. Our method is biased towards the extreme roughness value range but is more consistent. It is also worth noting that the roughness parameter plays a minor role compared to the diffuse color during re-rendering. If the color of an object is off, it is more visible than slight alterations in how reflective it is. Additionally, our method can estimate the specular color, which is a challenging task and allows our method to render metals correctly.

As Li *et al*. does not allow for drastic novel view synthesis – except slightly based on the estimated depth map – one approach to solve this is to use NeRF on top. By running Li *et al*. on the train set and then constraining NeRF to not use view-dependent effects and extending the RGB space to RGB + roughness, we can try to join the distinct images in a volumetric model. In Fig. A7 it is clearly visible that this method fails, as each image is quite different from the other, and NeRF cannot place the varying information at the correct locations.

Notations. All notations in this work are listed in Table A2.

References

[1] Blender Online Community. *Blender - a 3D modelling and rendering package*. Blender Foundation, Stichting Blender Foundation, Amsterdam, 2018. 1

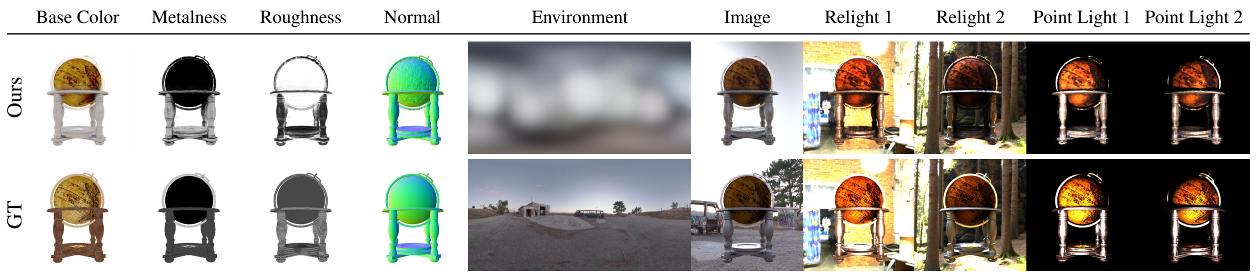


Figure A6: **Globe Decomposition.** Results on the decomposition of the synthetic globe scene. Even if the BRDF parameters do not capture the ground truth perfectly, the visual and quantitative error in re-rendering is extremely low. For most images, an alternative decomposition, which explains the input images, is found. As no other constraints exist, the solution is also plausible.

Symbol	$\in$	Description
q	$\mathbb{N}$	Number of images
s	$\mathbb{N}$	Number of pixel in each image
I_j	$\mathbb{R}^{s\times 3}; j \in 1, ..., q$	A specific image
$\boldsymbol{x}$	$\mathbb{R}^3$	The 3D coordinate in (x,y,z)
$\boldsymbol{b}$	$\mathbb{R}^5$	The BRDF parameters for the analytical cook torrance model
$\boldsymbol{n}$	$\mathbb{R}^3$	The surface normal
σ	$\mathbb{R}$	The density in the volume
$\boldsymbol{\Gamma}$	$\mathbb{R}^{24\times 7}$	The parameters for the spherical Gaussians environment illumination
t	$\mathbb{R}$	Used to query a position in distance t on a ray
$\boldsymbol{o}$	$\mathbb{R}^3$	The ray origin
$\boldsymbol{d}$	$\mathbb{R}^3$	The ray direction
$\boldsymbol{p}$	$\mathbb{R}^z$	A placeholder for a output of an object. Can be either BRDF parameters $\boldsymbol{b}$ or color $\boldsymbol{c}$. The dimensions z are dependent on the output type.
$\boldsymbol{c}^j$	$\mathbb{R}^3$	The potentially illumination dependent optimized color for the image j
$\boldsymbol{c}^j_{\omega_o r}$	$\mathbb{R}^3$	The illumination and view dependent optimized color for the image j
$\hat{\boldsymbol{c}}^j$	$\mathbb{R}^3$	The actual color for the image j
t_n	$\mathbb{R}$	The near clipping distance for the view frustum
t_f	$\mathbb{R}$	The far clipping distance of the view frustum
$\boldsymbol{\omega_i}$	$\mathbb{R}^3$	The incoming light direction (Pointing away from the surface)
$\boldsymbol{\omega_o}$	$\mathbb{R}^3$	The outgoing reflected light direction (Pointing away from the surface)
Ω		Defines the hemisphere at a point in normal direction $\boldsymbol{n}$
Function	$\in$	Description
$L_o(\boldsymbol{x}, \boldsymbol{\omega_o})$	$f(\mathbb{R}^3 \times \mathbb{R}^3) \mapsto \mathbb{R}^3$	The amount of outgoing light in the specified direction
$L_i(\boldsymbol{x}, \boldsymbol{\omega_i})$	$f(\mathbb{R}^3 \times \mathbb{R}^3) \mapsto \mathbb{R}^3$	The amount of incoming light from a specified direction
$f_r(\boldsymbol{x}, \boldsymbol{\omega_i}, \boldsymbol{\omega_o})$	$f(\mathbb{R}^3 \times \mathbb{R}^3 \times \mathbb{R}^3) \mapsto \mathbb{R}^3$	The BRDF, which is dependent on the position on the surface and the incoming and outgoing light directions
$\rho_d(\boldsymbol{\omega_o}, \boldsymbol{\Gamma}, \boldsymbol{n}, \boldsymbol{b})$	$f((\mathbb{R}^3 \times \mathbb{R}^7 \times \mathbb{R}^3 \times \mathbb{R}^5) \mapsto \mathbb{R}^3$	The diffuse lobe evaluation using spherical Gaussian representations
$\rho_s(\boldsymbol{\omega_o}, \boldsymbol{\Gamma}, \boldsymbol{n}, \boldsymbol{b})$	$f((\mathbb{R}^3 \times \mathbb{R}^7 \times \mathbb{R}^3 \times \mathbb{R}^5) \mapsto \mathbb{R}^3$	The specular lobe evaluation using spherical Gaussian representations
$\gamma(\boldsymbol{x})$	$f(\mathbb{R}^3) \mapsto \mathbb{R}^{3z}$	Maps the input point coordinate to a higher dimensional embedding using z Fourier embedding

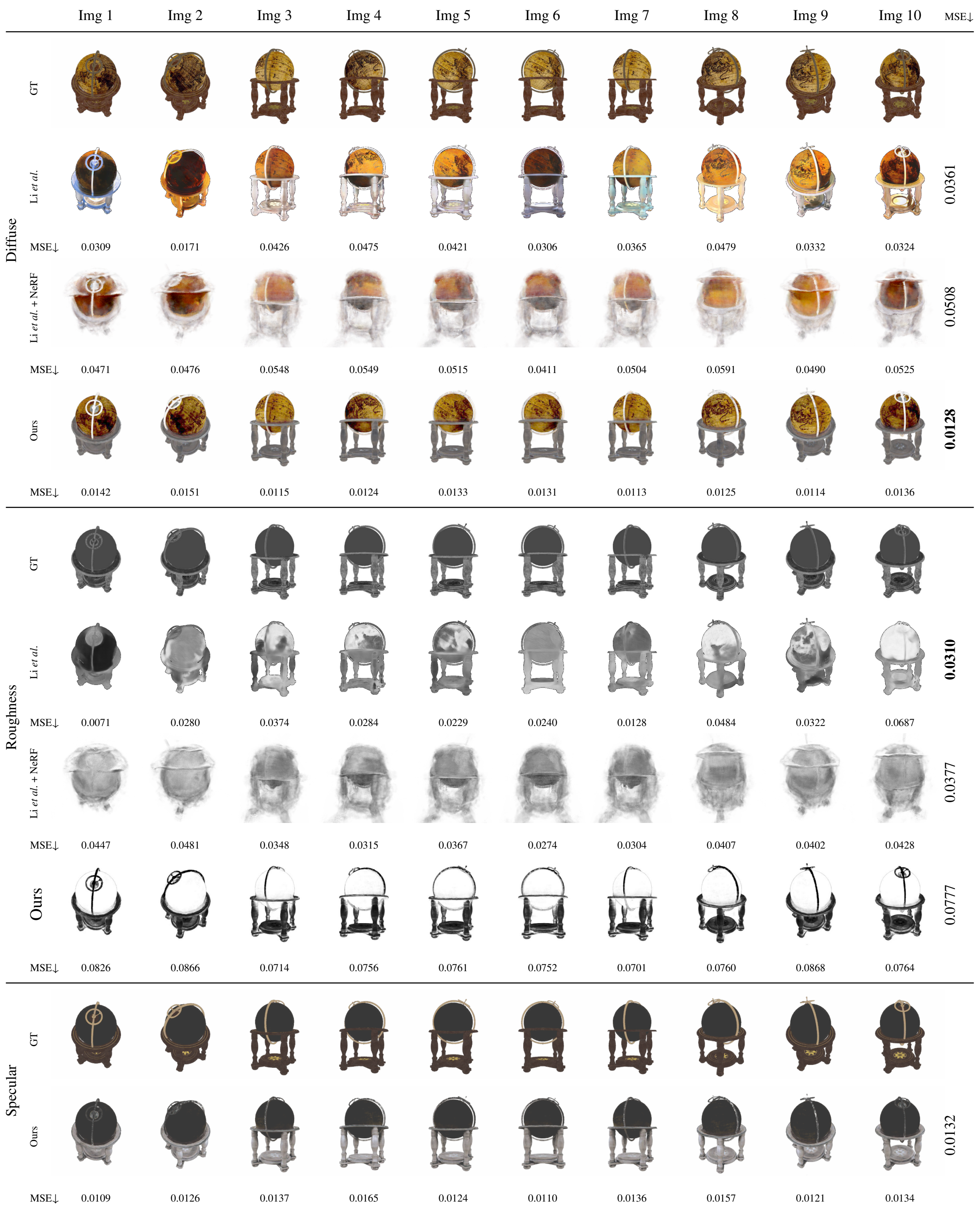
Table A2: **Notations.** Overview of all notations used in this work.

[2] Michael Kazhdan, Matthew Bolitho, and Hugues Hoppe. Poisson surface reconstruction. In *Proceedings of the fourth Eurographics symposium on Geometry processing*, volume 7, 2006. 1

[3] Zhengqin Li, Zexiang Xu, Ravi Ramamoorthi, Kalyan Sunkavalli, and Manmohan Chandraker. Learning to reconstruct shape and spatially-varying reflectance from a single image. In *ACM Transactions on Graphics (SIGGRAPH ASIA)*, 2018. 3, 5

[4] Ben Mildenhall, Pratul Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *European Conference on Computer Vision (ECCV)*, 2020. 2, 3

[5] Qian-Yi Zhou, Jaesik Park, and Vladlen Koltun. Open3D: A modern library for 3D data processing. *ArXiv e-prints*, 2018. 1



		Img 1	Img 2	Img 3	Img 4	Img 5	Img 6	Img 7	Img 8	Img 9	Img 10	MSE↓
Diffuse	Li *et al.* MSE↓	0.0309	0.0171	0.0426	0.0475	0.0421	0.0306	0.0365	0.0479	0.0332	0.0324	0.0361
	Li *et al.* + NeRF MSE↓	0.0471	0.0476	0.0548	0.0549	0.0515	0.0411	0.0504	0.0591	0.0490	0.0525	0.0508
	Ours MSE↓	0.0142	0.0151	0.0115	0.0124	0.0133	0.0131	0.0113	0.0125	0.0114	0.0136	**0.0128**
Roughness	Li *et al.* MSE↓	0.0071	0.0280	0.0374	0.0284	0.0229	0.0240	0.0128	0.0484	0.0322	0.0687	**0.0310**
	Li *et al.* + NeRF MSE↓	0.0447	0.0481	0.0348	0.0315	0.0367	0.0274	0.0304	0.0407	0.0402	0.0428	0.0377
	Ours MSE↓	0.0826	0.0866	0.0714	0.0756	0.0761	0.0752	0.0701	0.0760	0.0868	0.0764	0.0777
Specular	Ours MSE↓	0.0109	0.0126	0.0137	0.0165	0.0124	0.0110	0.0136	0.0157	0.0121	0.0134	0.0132

Figure A7: **Partial Estimation Globe Comparison.** Comparison with partial estimation methods. Here, we compare our method with Li *et al.* [3] and Li *et al.* + NeRF. Li *et al.* is a method that decomposes a single image of an object illuminated by environment light into diffuse and roughness parameters. For the Li *et al.* + NeRF baseline, we first translated the training dataset and then trained a NeRF with disabled view conditioning on top. We then generate the novel test set views. For Li *et al.*, no view synthesis takes place, and the method is run on the test set directly. Notice how our method generates consistent results on all test views.