

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#}

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

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

³Hong Kong University of Science and Technology ⁴Light 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.

6. Implementation Details

6.1. Network Architecture

VMINer has 6 trainable modules: 4 MLPs including M_{geo} , M_{mat} , M_{far} , and M_{near} , and 2 multi-resolution hash grid encoding including enc_{geo} and enc_{mat} .

The SDF MLP M_{geo} contains 1 hidden layer with 64 neurons with $\text{SoftPlus}(x) = \log(1 + e^x)$ as the activation function. We apply weight normalization reparameterization [12] and initialize it so that its output approximates the SDF field of a sphere.

The material MLP M_{mat} , the far-field radiance MLP M_{far} , and the near-field radiance MLP M_{near} all share the same network architecture: the fully-fused MLP from [10] containing 2 hidden layers with 64 neurons each, using ReLU [11] as in-network activation functions and sigmoid as output activation functions.

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

6.2. Training Scheduling

In addition to the above trainable modules, other trainable parameters include the surface concentration parameter b , the light embedding $\mathbf{F}_{\{\text{far}, \text{near}\}_i}$, the far-field lighting parameters ξ_{ij} , λ_{ij} , and μ_{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 $\mathbf{F}_{\{\text{far}, \text{near}\}_i}$ and the parameters of enc_{geo} , M_{geo} , M_{far} , and M_{near} , are set to 10^{-2} . The base learning rate of parameters of enc_{mat} and M_{mat} is set to $3 \cdot 10^{-2}$. The

base learning rate of lighting parameters ξ_{ij} , λ_{ij} , μ_{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 (M_{geo} , M_{far} , M_{near} , enc_{geo} , b , $\mathbf{F}_{\{\text{far}, \text{near}\}_i}$) 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}_{rf} and \mathcal{L}_{pb} , the Eikonal loss \mathcal{L}_{eik} , the silhouette loss \mathcal{L}_{sil} , the self-consistency loss \mathcal{L}_{con} , and the smoothness loss \mathcal{L}_{smo} . The loss terms are divided into two groups and are used separately in two stages: $\mathcal{L}_{\text{stage1}} = \mathcal{L}_{\text{rf}} + \lambda_{\text{eik}} \mathcal{L}_{\text{eik}} + \lambda_{\text{sil}} \mathcal{L}_{\text{sil}} + \lambda_{\text{ns}} \mathcal{L}_{\text{ns}}$ and $\mathcal{L}_{\text{stage2}} = \mathcal{L}_{\text{pb}} + \lambda_{\text{con}} \mathcal{L}_{\text{con}} + \lambda_{\text{ms}} \mathcal{L}_{\text{ms}}$. We typically set $\lambda_{\text{eik}} = 2 \cdot 10^{-4}$, $\lambda_{\text{sil}} = 10^{-3}$, $\lambda_{\text{ns}} = 10^{-3}$, $\lambda_{\text{con}} = 0.1$, and $\lambda_{\text{ms}} = 10^{-3}$.

6.4. BRDF Model

We use a simplified GGX Microfacet BRDF [17] 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 β 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 Wild-Light [3] to process the SDF and BRDF fields into textured meshes. The zero-level isosurface is extracted using the marching cubes algorithm [9], simplified [5], and stored as

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

a triangle mesh. We then use UV unwrapping [8] 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 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 TensorIR [7] and WildLight [3]. Thus, we have to control the tone mapping curve of the input images. We follow WildLight [3] 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 [15, 16] with HLoc [13, 14] 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 [4] 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 fore-

ground 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 [3] (one far-field lighting with flashlight), TensorIR [7] (two far-field lighting), and NVDiffRecMC [6] (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 [3] (one far-field lighting with flashlight), TensorIR [7] (one far-field lighting), and NVDiffRecMC [6] (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 [3] fails to recover the geometrical details and TensorIR [7] 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 [3] and TensorIR [7] 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 [3]. 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 material to compensate for imperfect estimation of shading. In that context, one key characteristic of WildLight [3] 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 Wild-

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. \uparrow (\downarrow) means bigger (smaller) is better.

Scene	Method (input lighting)	Normal				Albedo			View synthesis			FV relit			FV relit (spec)		
		MAngE \downarrow	PSNR \uparrow	SSIM \uparrow	LPIPS \downarrow	PSNR \uparrow	SSIM \uparrow	LPIPS \downarrow	PSNR \uparrow	SSIM \uparrow	LPIPS \downarrow	PSNR \uparrow	SSIM \uparrow	LPIPS \downarrow	PSNR \uparrow	SSIM \uparrow	LPIPS \downarrow
 BARRELSOFA	(1) TensoIR [7] (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 [6] (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 [3] (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	<u>31.683</u>	<u>0.945</u>	<u>0.092</u>	32.321	0.956	<u>0.085</u>	31.186	<u>0.946</u>	<u>0.097</u>	28.972	0.881	<u>0.159</u>			
	(6) TensoIR [7] (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)	<u>6.701</u>	32.378	0.948	0.085	<u>31.745</u>	<u>0.955</u>	0.082	<u>31.172</u>	0.948	0.090	<u>28.575</u>	<u>0.866</u>	0.156			
 HANDBAG	(1) TensoIR [7] (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 [6] (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 [3] (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	<u>33.014</u>	0.946	0.107	35.885	0.967	0.084	32.361	0.931	0.115			
	(6) TensoIR [7] (2F)	17.261	27.250	0.928	<u>0.068</u>	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	<u>0.086</u>	28.180	0.937	0.090	31.335	0.923	0.118			
	(8) Ours (2F1N)	<u>6.067</u>	<u>32.760</u>	<u>0.954</u>	0.088	33.259	0.943	0.110	<u>36.795</u>	<u>0.968</u>	<u>0.083</u>	<u>34.329</u>	0.945	<u>0.109</u>			
 HOTDOG	(1) TensoIR [7] (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 [6] (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 [3] (1F1N)	10.254	24.238	<u>0.952</u>	0.076	29.563	0.936	0.087	<u>30.163</u>	<u>0.939</u>	0.080	20.070	0.794	0.154			
	(5) Ours (1F1N)	<u>10.072</u>	<u>24.422</u>	0.940	0.112	<u>29.733</u>	0.949	0.080	29.186	0.937	0.090	<u>29.745</u>	<u>0.905</u>	<u>0.123</u>			
	(6) TensoIR [7] (2F)	11.197	24.103	0.949	<u>0.094</u>	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	<u>0.094</u>	30.520	<u>0.944</u>	<u>0.085</u>	30.449	0.942	<u>0.085</u>	30.557	0.913	0.112			
 LEGO	(1) TensoIR [7] (1F)	20.195	27.988	0.937	<u>0.089</u>	29.982	0.918	0.090	32.029	0.915	0.093	33.296	0.835	0.156			
	(2) NVDiffRecMC [6] (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 [3] (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	<u>30.801</u>	0.941	0.109	<u>30.834</u>	0.929	<u>0.076</u>	<u>33.366</u>	<u>0.940</u>	<u>0.074</u>	<u>34.096</u>	0.856	<u>0.146</u>			
	(6) TensoIR [7] (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)	<u>17.878</u>	26.576	0.861	0.165	29.258	0.913	0.079	31.718	0.928	<u>0.074</u>	32.135	<u>0.859</u>	0.155			
	(8) Ours (2F1N)	17.287	31.201	<u>0.945</u>	0.101	31.150	<u>0.927</u>	0.075	33.889	0.943	0.068	34.966	0.873	0.139			
 SHOES	(1) TensoIR [7] (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 [6] (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 [3] (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)	<u>16.786</u>	<u>31.241</u>	<u>0.965</u>	<u>0.052</u>	29.494	<u>0.959</u>	<u>0.050</u>	<u>26.118</u>	0.949	<u>0.057</u>	<u>27.249</u>	<u>0.914</u>	0.089			
	(6) TensoIR [7] (2F)	22.102	24.923	0.898	0.128	<u>30.165</u>	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	<u>0.095</u>			
	(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 [7] (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 [6] (1F)	12.418	35.243	0.968	<u>0.041</u>	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 [3] (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	<u>0.978</u>	<u>0.041</u>	37.113	0.979	0.038	36.239	0.977	0.045	32.071	<u>0.948</u>	0.069			
	(6) TensoIR [7] (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)	<u>8.349</u>	<u>39.169</u>	0.975	0.042	<u>36.707</u>	<u>0.977</u>	<u>0.039</u>	<u>36.046</u>	<u>0.976</u>	0.045	<u>32.385</u>	0.949	<u>0.068</u>			

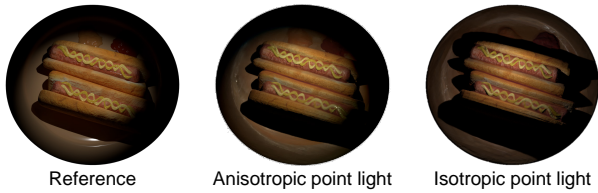


Figure 5. Comparison between the reconstruction under an anisotropic point light model (adopted by our method) and under an isotropic point light model.

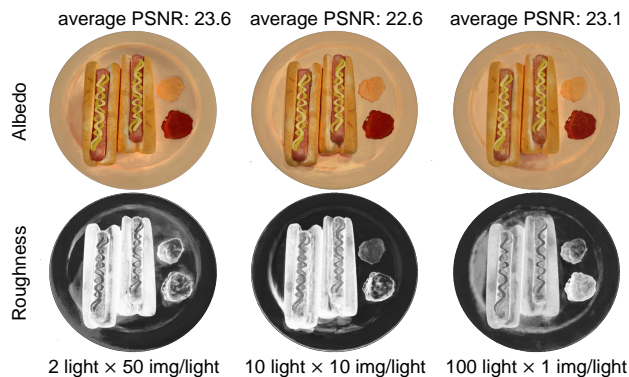


Figure 6. Results on synthetic scenes with varying amounts of far-field lighting.

Light [3] to ascribe spatial variation of radiance to the neural radiance instead of material. This helps WildLight [3] 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 [3] 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 collec-

tions. 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.

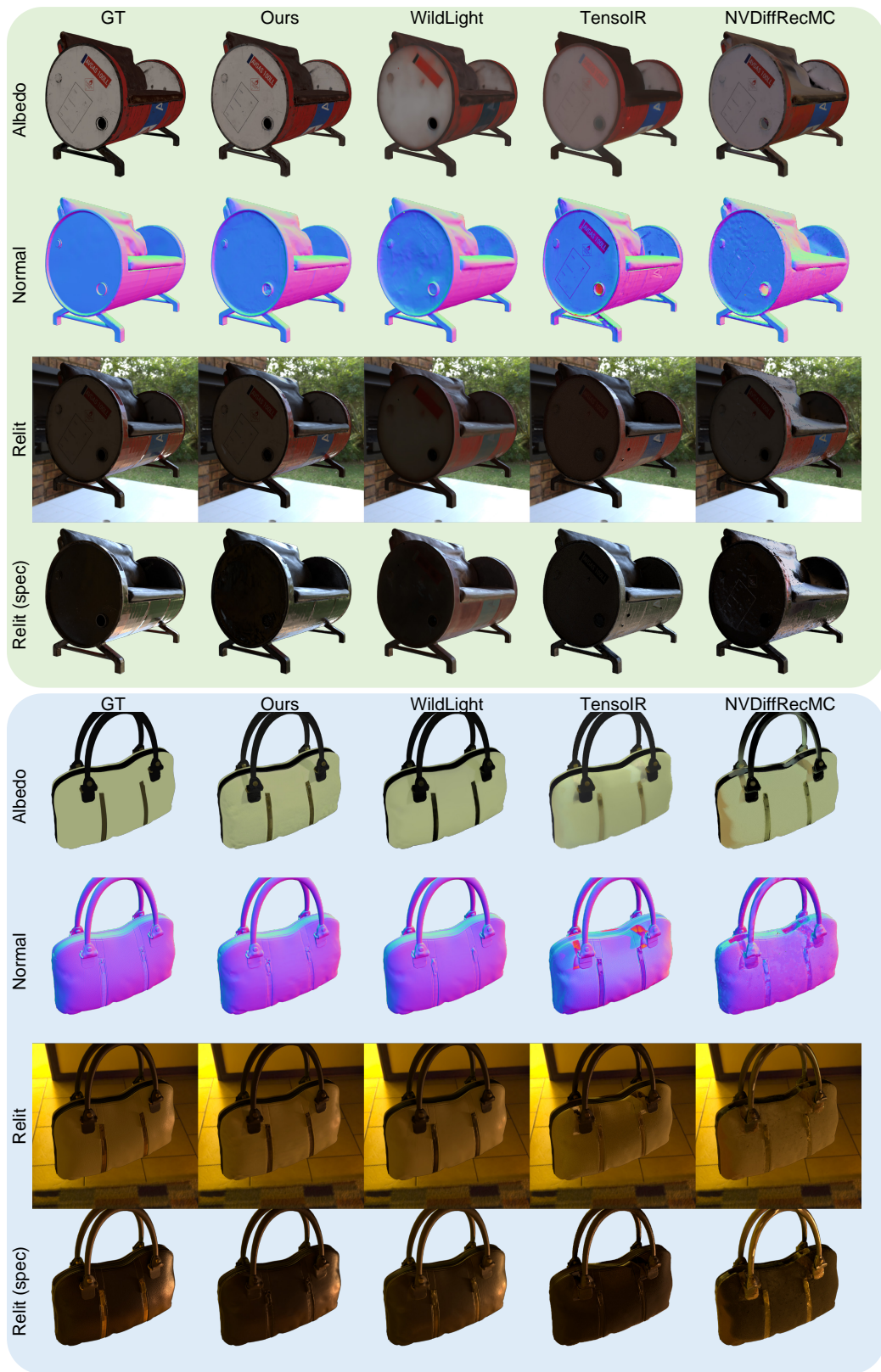


Figure 7. Comparison with state-of-the-art methods on two synthetic scenes: BARRELSOFA and HANDBAG.

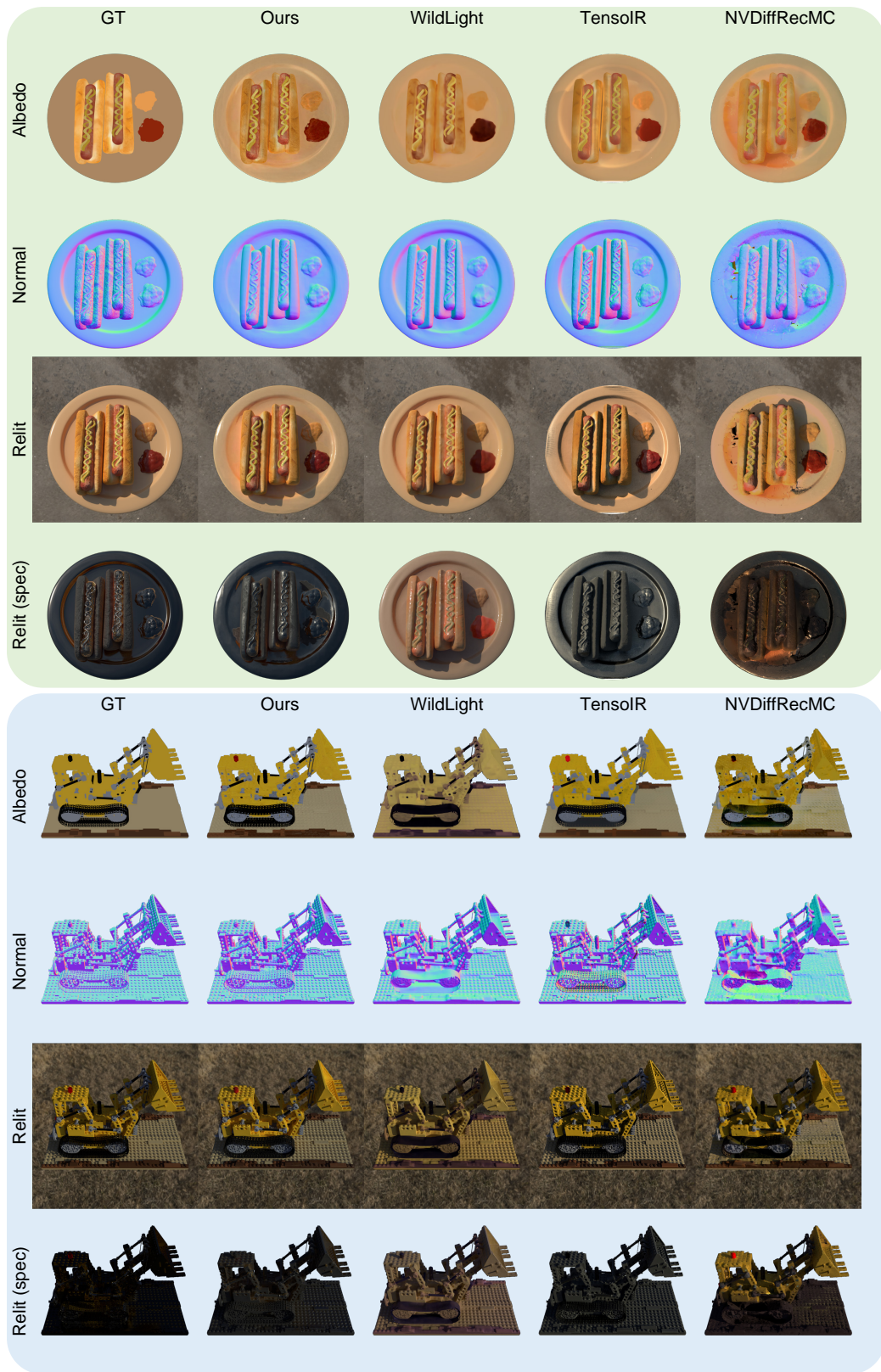


Figure 8. Comparison with state-of-the-art methods on two synthetic scenes: HOTDOG and LEGO.

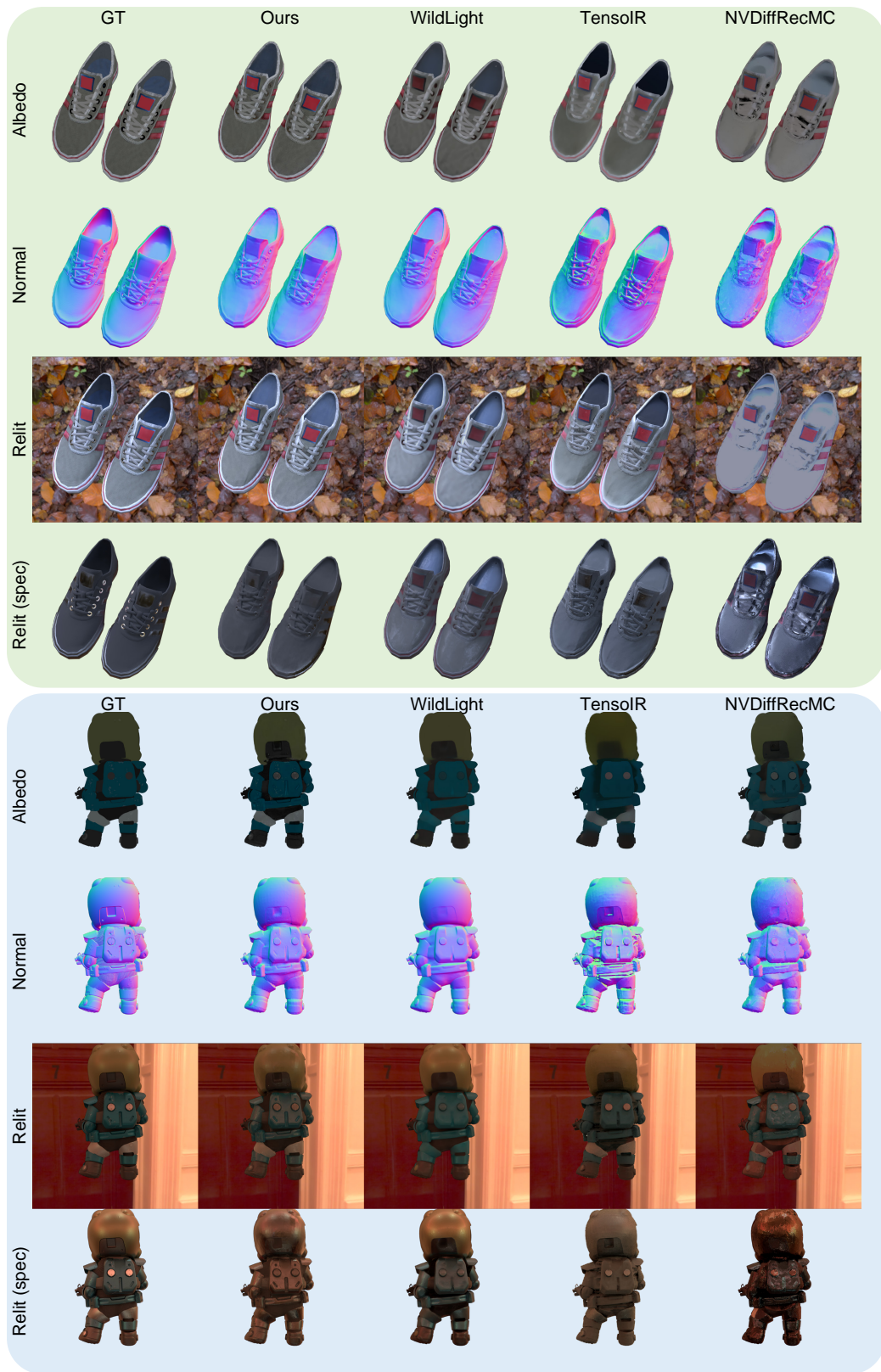


Figure 9. Comparison with state-of-the-art methods on two synthetic scenes: SHOES and TROOPER.

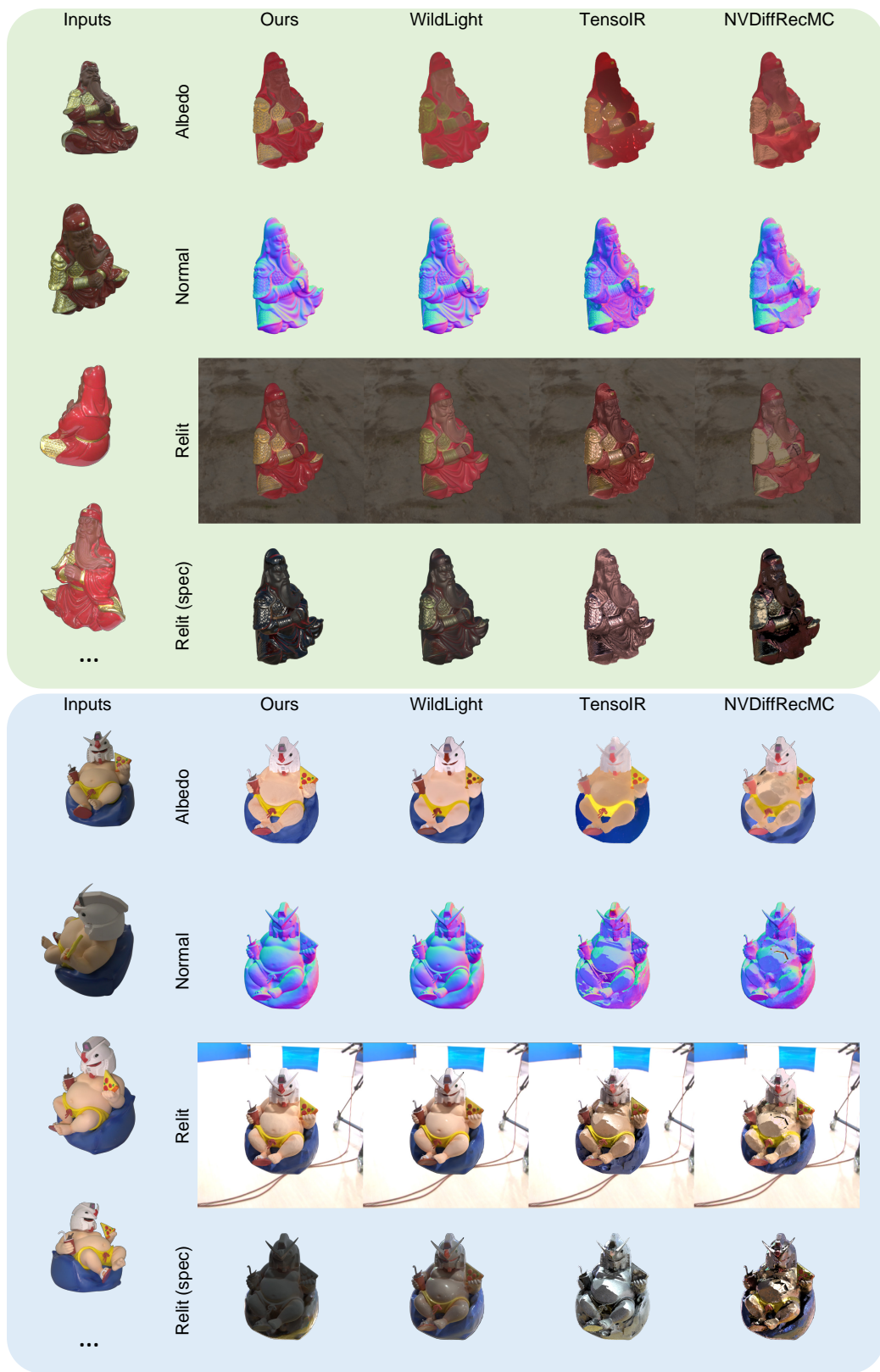


Figure 10. Comparison with state-of-the-art methods on two real-world scenes: GUANYU and DEBUGUNDAM.

References

- [1] Blender Foundation. The Blender project - free and open 3D creation software. <https://www.blender.org>. Accessed: 2023-11-07. **2**
- [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. **2, 4**
- [3] 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, 4**
- [4] 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. **2**
- [5] 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. **1**
- [6] 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. **2, 3**
- [7] Haiyan Jin, Isabella Liu, Peijia Xu, Xiaoshuai Zhang, Songfang Han, Sai Bi, Xiaowei Zhou, Zexiang Xu, and Hao Su. TensorIR: Tensorial inverse rendering. In *Proc. of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023. **2, 3**
- [8] 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. **2**
- [9] 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. **1**
- [10] 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**
- [11] Vinod Nair and Geoffrey E. Hinton. Rectified linear units improve restricted boltzmann machines. In *Proc. of International Conference on Machine Learning (ICML)*, 2010. **1**
- [12] 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. **1**
- [13] 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. **2**
- [14] 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. **2**
- [15] 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. **2**
- [16] 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. **2**
- [17] 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. **1**