Supplementary Material for MAIR: Multi-view Attention Inverse Rendering with 3D Spatially-Varying Lighting Estimation

JunYong Choi[1,2] SeokYeong Lee[1,2] Haesol Park[1] Seung-Won Jung[2] Ig-Jae Kim[1,3,4] Junghyun Cho[1,3,4]

[1]Korea Institute of Science and Technology(KIST) [2]Korea University
[3]AI-Robotics, KIST School, University of Science and Technology
[4]Yonsei-KIST Convergence Research Institute, Yonsei University

{happily,shapin94,haesol,drjay,jhcho}@kist.re.kr swjung83@korea.ac.kr

1. OpenRooms FF dataset

We created a dataset for multi-view inverse rendering called OpenRooms Forward Facing (Openrooms FF) dataset. Openrooms FF is an extension of the existing single-view inverse rendering dataset, OpenRooms [7], and most of resources to build the dataset are provided by the authors of OpenRooms [7], including data sources and creation tools. The materials, however, were unavailable due to the licensing issue, so we had to purchase materials from Adobe Stock [1] except for 200 materials that were not found from Adobe Stock; instead, we replace them with other similar materials. We selected 23,618 images from the OpenRooms dataset by filtering out the images in which the camera looks at a wall or window, lacks textures in the scene, or object is too close to the camera. Then, we rendered forward facing multi-view images of 3 $\times$ 3 arrays by moving camera in eight directions: up, right up, right, right down, down, left down, and left, left up using the OptiX-based renderer [6]. The baseline was set proportionally to the average depth of the scene to observe the change in the specular radiance. See Fig. 1 for a multi-view images sample. As a result, a total of 212,562 (9 $\times$ 23,618) images were created and 27,000 (9 $\times$ 3000) images were separated into test dataset. OpenRooms FF consists of HDR RGB images, diffuse albedo images, roughness images, normal maps, binary masks, depth maps, per-pixel environment maps. We rendered images at 640 $\times$ 480 resolution but resized to 320 $\times$ 240 with bilinear interpolation for the training/test. The OpenRooms FF is summarized in Tab. 1.

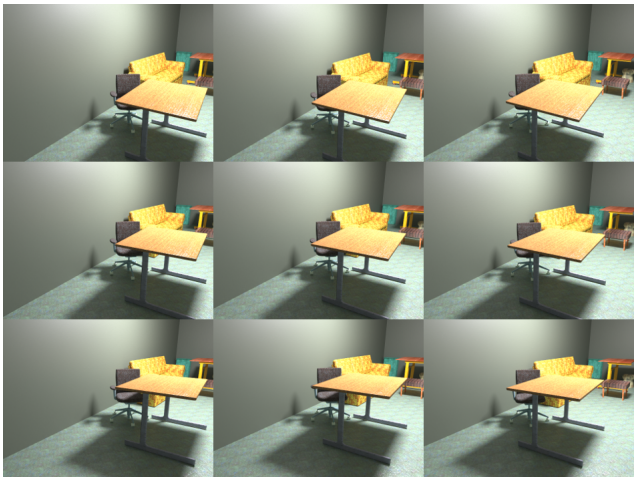

Figure 1. Sample of forward facing multi-view images in OpenRooms FF.

	Dataset	Training / Test
HDR RGB	640 $\times$ 480	320 $\times$ 240
Diffuse Albedo	640 $\times$ 480	320 $\times$ 240
Roughness	640 $\times$ 480	320 $\times$ 240
Normal	640 $\times$ 480	320 $\times$ 240
Mask	640 $\times$ 480	320 $\times$ 240
Depth	640 $\times$ 480	Not used
per-pixel DL	40 $\times$ 30 $\times$ 32 $\times$ 16	40 $\times$ 30 $\times$ 16 $\times$ 8
per-pixel SVL	160 $\times$ 120 $\times$ 32 $\times$ 16	160 $\times$ 120 $\times$ 16 $\times$ 8

Table 1. Data type and resolution of OpenRooms FF. Spatially-varying lighting (SVL) has a spatial resolution of 160 $\times$ 120 and an angular resolution of 32 $\times$ 16.

2. Direct Lighting Details

Since the intensity($\boldsymbol{\eta}_s$) of incident direct lighting is the intensity of the light source, it is unrelated to pixel location. Thus we use global intensities $\boldsymbol{\eta}_s$ rather than per-pixel intensities. Instead, per-pixel visibility $\mu_s \in \mathbb{R}$ was used to account for occlusion. To enhance the dynamic range of the SG lobes, we use the non-linear transformation [5]. The ablation study results for S_D in SVSGs of incident direct lighting are shown in Tab. 2. Please see Eq. (7) for $\mathcal{L}_{\text{reg}}$. Direct lighting performance improved as S_D increased, but

This work was partly supported by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government(MSIT)(No.2020-0-00457, 50%) and KIST Institutional Program(Project No.2E32301, 50%).

GPU Memory also increased. We chose $S_D = 3$ considering its performance and GPU usage. Fig. 2 shows the incident(SVSGs) / exitant($\tilde{\mathrm{V}}_{\mathrm{DL}}$) direct lighting estimation results. SVSGs generally performed better because $\tilde{\mathrm{V}}_{\mathrm{DL}}$ estimates 3D volume, while SVSGs directly estimates 2D per-pixel environment map(E). Also, even though the consistency between them is not considered, since they are trained with the same ground truth(GT), they are consistent enough as shown in the Fig. 2.

S_D	si-MSE	$\mathcal{L}_{\text{reg}}$	GPU Memory(GB).
1	0.106	0.136	10.8
2	0.103	0.127	11.26
3	**0.101**	**0.092**	**12.72**
4	0.101	0.081	13.43
6	0.100	0.061	14.94

Table 2. The ablation study results for S_D in SVSGs.

3. Analysis of Lighting Estimation Results

We have analyzed spatially-varying lighting quality in detail. Since the SVLNet implementation is quite memory-hungry, the resolution of our $\tilde{\mathrm{V}}_{\mathrm{SVL}}$ is 128^3, which is low compared to the image resolution (320×240). Also, because the field-of-view of our camera setup is limited, the lighting of the out-of-view area must rely on context inference about the dataset. Fig. 3 shows the per-pixel lighting estimation results for the OpenRooms FF test scene. In the Fig. 3, our estimation approximates the overall outline of the GT better than Li *et al*. [5] , but fails to mimic the high frequency details of the GT due to limitations in resolution and field-of-view.

4. View Synthesis

While image-based rendering(IBR) can perform view interpolation excellently, the view-dependent effect of highly specular objects, such as chrome spheres, is difficult to reproduce using IBR. Physically-based rendering(PBR) can handle this view-dependent effect realistically, but PBR requires scene material, geometry, and spatially-varying lighting that is difficult to obtain in the real-world. Because MAIR can perform accurate inverse rendering in real-world scenes, and can be easily applied to existing view synthesis methods with multi-view images, we can take advantage of IBR and PBR. The view synthesis result of the scene with chrome sphere inserted is in the accompanied video. This application consists of two steps: (1) background rendering with NeRF [10], and (2) object and mask rendering with our renderer. We render the shadow of an object in all images and we train NeRF with these images. Background including shadow in novel view is rendered with NeRF, and chrome sphere in novel view is rendered with our lighting and renderer. Among the variants of NeRF, we use DirectVoxGO [12] for fast training.

5. Implementation details

Training and architecture details. Our experiments were conducted with 8 NVIDIA RTX A5000 (24GB). In training, we use Adam optimizer, and the binary mask image $(\mathrm{M}_o, \mathrm{M}_l)$. $\mathrm{M}_o \in \mathbb{R}^{H \times W}$ is mask on pixels of valid materials, and $\mathrm{M}_l \in \mathbb{R}^{H \times W}$ is mask on pixels of valid materials and area lighting. The binary mask image is included in the OpenRooms FF and is used only for training. First, we define masked L1 angular error function (g_1), masked MSE function (g_2), masked scale invariant MSE function (g_3), masked scale invariant log space MSE function (g_4), and regularization function (g_5) as follows.

$$g_1(A, B, M) = ||(\cos^{-1}(A \odot B)) \otimes M||_1, \quad (1)$$

$$g_2(A, B, M) = ||(A - B) \otimes M||_2^2, \quad (2)$$

$$g_3(A, B, M) = ||(A - \tau B) \otimes M||_2^2, \quad (3)$$

$$g_4(A, B, M) = ||(\log(A + 1) - \log(\tau B + 1)) \otimes M)||_2^2, \quad (4)$$

$$g_5(A) = -A \log(A), \quad (5)$$

where $\odot$ is element-wise dot product, $\otimes$ is element-wise multiplication, and τ is the scale obtained by least square regression between A and B.

In stage 1, the loss function of NormalNet is as follows:

$$\mathcal{L}_{\text{normal}} = \beta_1 g_1(\mathrm{N}, \tilde{\mathrm{N}}, \mathrm{M}_l) + \beta_2 g_2(\mathrm{N}, \tilde{\mathrm{N}}, \mathrm{M}_l). \quad (6)$$

NormalNet has a U-Net [8] structure with 6 down-up convolution blocks.

Since the light source is not transparent, we use a regularization g_5 so that the visibility μ_s of InDLNet and the opacity α of ExDLNet can be 0 or 1. the loss function of InDLNet and ExDLNet is as follows:

$$\mathcal{L}_{\text{InDL}} = \beta_1 g_4(\mathrm{E}_{DL}, \tilde{\mathrm{E}}_{DL}, \mathrm{M}_o) + \beta_2 g_5(\mu_s), \quad (7)$$

$$\mathcal{L}_{\text{ExDL}} = \beta_1 g_4(\mathrm{E}_{DL}, \tilde{\mathrm{E}}_{DL}, \mathrm{M}_o) + \beta_2 g_5(\alpha), \quad (8)$$

where E_{DL} is the per-pixel direct lighting environment map. InDLNet also has a U-Net structure that encoder is shared, and decoders are separated by λ_s, ξ_s, μ_s. The light source intensity η_s was decoded using MLP. ExDLNet follows structure of OccNet [9] and uses MLP with conditional batch normalization (CBN) [3]. All convolution blocks use batch normalization(BN).

In stage2, the loss function is as follows.

$$\mathcal{L}_{\text{BRDF}} = \beta_1 g_3(\mathrm{A}, \tilde{\mathrm{A}}, \mathrm{M}_o) + \beta_2 g_2(\mathrm{R}, \tilde{\mathrm{R}}, \mathrm{M}_o). \quad (9)$$

ContextNet uses U-Net with ResNet18 [4], SpecNet uses MLP with 3 layers, MVANet uses layer normalization (LN), and RefineNet uses U-Net with group normalization(GN).

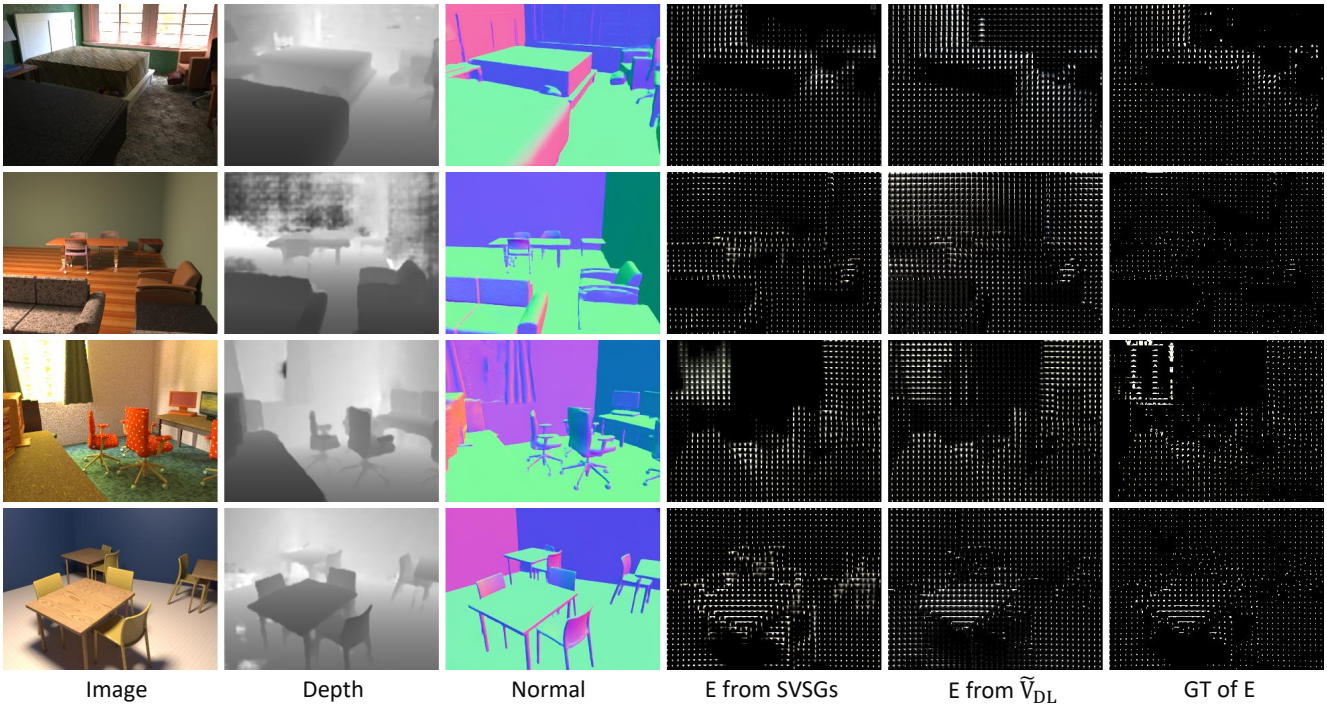

Figure 2. Direct lighting environment map ($16 \times 8 \times 3$) estimation results for OpenRooms FF.

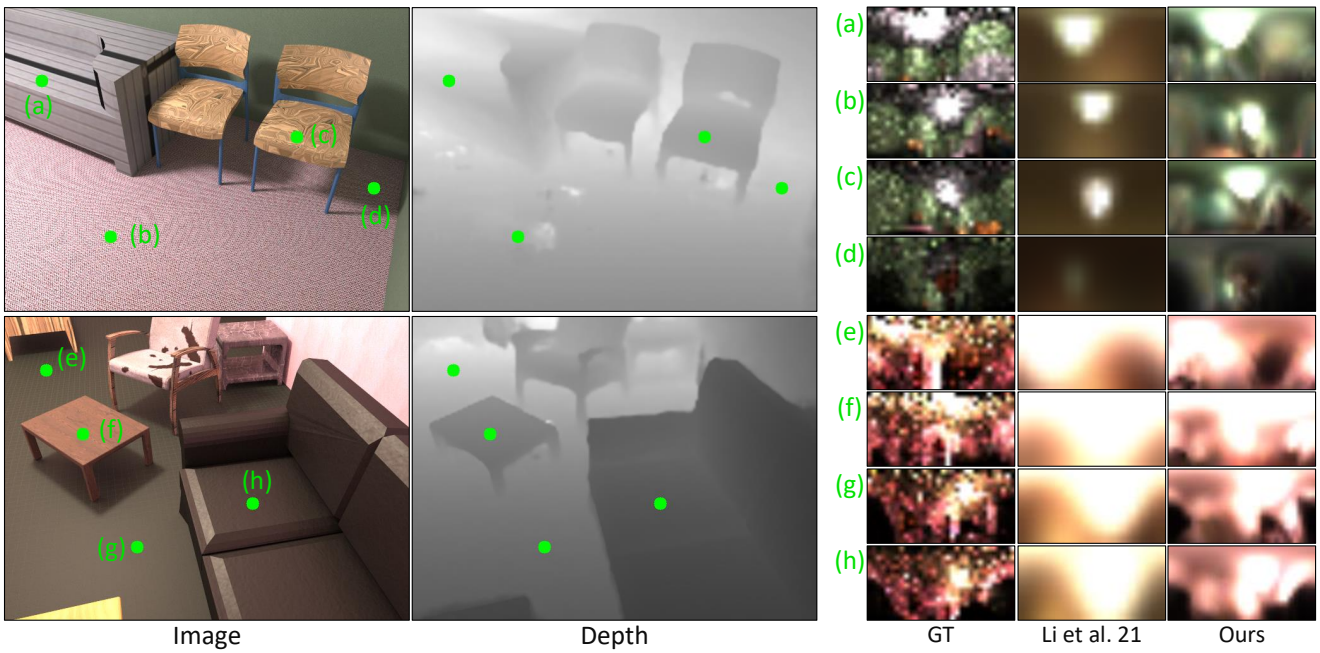

Figure 3. Per-pixel environment map ($32 \times 16 \times 3$) estimation results for OpenRooms FF.

In stage3, the loss function is as follows.

$$\mathcal{L}_{\text{SVL}} = \beta_1 g_4(\mathrm{E}_{SVL}, \tilde{\mathrm{E}}_{SVL}, \mathrm{M}_o) + \beta_2 g_5(\alpha) + \beta_3 \sum_{k=1}^{K} ||w_k(\mathrm{I}^k - \tau_{diff}\tilde{\mathrm{I}}_{diff} - \tau_{spec}\tilde{\mathrm{I}}^k_{spec}) \otimes \mathrm{M}_o||_2^2, \quad (10)$$

where E_{SVL} is the per-pixel lighting environment map, τ_{diff} and τ_{spec} are the scale obtained by least square regression with target image. $\mathrm{I}^k, \tilde{\mathrm{I}}_{diff}, \tilde{\mathrm{I}}^k_{spec}$ are k-view image, diffuse image, k-view specular image, respectively, and w_k is multi-view weight. In SVLNet, visible surface volume

(T) is concatenated with $\tilde{\mathrm{V}}_{\mathrm{DL}}$ after 2 downsampling and processed with 3D U-Net. The resolution of the $\tilde{\mathrm{V}}_{\mathrm{DL}}$ is 32^3, and the resolution of the T and $\tilde{\mathrm{V}}_{\mathrm{SVL}}$ is 128^3. SVLNet uses instance normalization(IN). SVLNet needs a lot of memory when training, so we render environment map with a spatial resolution of 60×80. A summary of training, number of GPUs, hyperparameter and network architecture is provided in Tab. 3. Rendering includes the time to obtain a $60\times80\times8\times16$ environment map from VSG and the time to re-render the input image.

Test details. Li *et al*. [5] and we both used an environment map with an angular resolution of 16×8 during training, but we created an environment map with 32×16 during testing because our VSG was not restricted by resolution. In training, all views are rendered for re-rendering loss, but in testing, only the target view was rendered.

6. Additional Experimental Results

6.1. Indoor Synthetic Scenes

We provide additional inverse rendering results for OpenRooms FF test scene in Fig. 4. Our method leverage multi-view and incident direct lighting to provide more accurate material estimation results for highly specular regions. (*e.g*. table in sample 2, chair in sample 3) Furthermore, the proposed method yields better normal estimation results especially for more complicated structures by utilizing MVS depth. As a result, our lighting is more realistic and we can re-render input image more accurately.

6.2. Real-World Scenes

The performance gaps between MAIR and the single-view-based methods are more distinct in the unseen real-world scene. Fig. 5 shows that our method robustly produces reasonable normal maps even for complex scene structures, and this naturally affects the subsequent material, lighting estimation. MAIR shows better material estimation results for shadowed regions(*e.g*. table, wall in sample 2, floor in sample 3) or specular regions(*e.g*. drawer in sample 4). Although there are no ground truths for materials, from our experience, we know that the stones, bushes in sample 1, and the dolls in sample 5 should show high roughness, which are consistent with our high roughness estimation results.

6.3. Object Insertion

Inverse rendering performance of three competing methods, lighthouse [11], Li *et al*. [5], and MAIR, are tested by comparing the quality of object insertion. We implemented a simple renderer for object insertion by referring to Wang *et al*. [14] and used it for rendering results of MAIR and lighthouse [11]. As the public implementation of Li *et al*. [5] includes a renderer of their own, results of Li *et al*. [5] were rendered using this renderer, except for the results of the chrome sphere insertion; the renderer from Li *et al*. [5] does not support the chrome sphere rendering directly, so we used our renderer for this case. It should be also noted that all results of lighthouse [11] were produced by using our scene geometries because scene geometry results from lighthouse [11] were not accurate enough to render.

We conducted a user study to evaluate the quality of object insertion from the three methods. Given a background image and an object of a particular material, users selected the most natural image among the three different results in a random order. 100 users evaluated 25 different scenes. Fig. 6, 7, 8, 9, and 10 show all the scenes used in our user study. Our 3D lighting not only clearly expresses HDR lighting, but also fully reflects real-world scene geometry and material. This allowed the object to be realistically inserted into the scene, acquiring the highest score among the competing methods.

We also provide additional object insertion results. In the accompanied video, the object can be located not only on the plane but also on any geometry, and the shadow of the object realistically appears to match the scene illumination.

References

[1] Adobe Stock. https://stock.adobe.com/3d-assets. 1

[2] The stanford 3d scanning repository. https://graphics.stanford.edu/data/3Dscanrep. 10, 11

[3] Harm De Vries, Florian Strub, Jérémie Mary, Hugo Larochelle, Olivier Pietquin, and Aaron C Courville. Modulating early visual processing by language. *Advances in Neural Information Processing Systems*, 30, 2017. 2

[4] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 2

[5] 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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2475–2484, 2020. 1, 2, 4

[6] Zhengqin Li, Mohammad Shafiei, Ravi Ramamoorthi, Kalyan Sunkavalli, Manmohan Chandraker, and Yu-Ying Yeh. Optixrenderer. https://github.com/lzqsd/OptixRenderer. 1

[7] Zhengqin Li, Ting-Wei Yu, Shen Sang, Sarah Wang, Meng Song, Yuhan Liu, Yu-Ying Yeh, Rui Zhu, Nitesh Gundavarapu, Jia Shi, et al. Openrooms: An open framework for photorealistic indoor scene datasets. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7190–7199, 2021. 1

[8] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *Pro-*

Stage	Network	input	Arch	norm	batch	epoch	β_1	β_2	β_3	lr	training / GPUs	inference	output(channels)
1	NormalNet	$\mathrm{I}, \tilde{\mathrm{D}}, \nabla\tilde{\mathrm{D}}, \tilde{\mathrm{C}},$	U-Net	BN	96	60	1.0	1.0	-	2e-3	7h / 4	3ms	$\tilde{\mathrm{N}}(3)$
	InDLNet	$\mathrm{I}, \tilde{\mathrm{N}}, \tilde{\mathrm{D}}, \tilde{\mathrm{C}}$	U-Net, MLP	BN	384	80	1.0	1e-3	-	2e-4	10h / 4	5ms	$\boldsymbol{\xi}_s, \lambda_s, \mu_s, \boldsymbol{\eta}_s(8)$
	ExDLNet	$\mathrm{I}, \tilde{\mathrm{N}}, \tilde{\mathrm{D}}, \tilde{\mathrm{C}}$	U-Net, MLP	BN	96	80	1.0	1e-4	-	1e-4	1d / 8	6ms	$\tilde{\mathrm{V}}_{\mathrm{DL}}(8)$
2	ContextNet	$\mathrm{I}, \tilde{\mathrm{N}}, \tilde{\mathrm{D}}, \tilde{\mathrm{C}}$	Res U-Net	BN	64	40	3.0	1.0	-	1e-4	1d 20h / 8	54ms	$\boldsymbol{f}_{\mathrm{context}}(32)$
	SpecNet	$\boldsymbol{\xi}_s, \lambda_s, \mu_s, \boldsymbol{\eta}_s, \boldsymbol{v}, \tilde{\mathrm{N}}$	MLP	-									$\boldsymbol{f}_{\mathrm{spec}}(8)$
	MVANet	$\mathrm{I}, \boldsymbol{f}_{\mathrm{context}}, \boldsymbol{f}_{\mathrm{spec}}, \boldsymbol{w}$	-	LN									$\boldsymbol{f}_{\mathrm{BRDF}}(16)$
	RefineNet	$\mathrm{I}, \tilde{\mathrm{N}}, \tilde{\mathrm{D}}, \tilde{\mathrm{C}}, \boldsymbol{f}_{\mathrm{context}}, \boldsymbol{f}_{\mathrm{BRDF}}$	U-Net	GN									$\tilde{\mathrm{A}}(3), \tilde{\mathrm{R}}(1)$
3	SVLNet	$\mathrm{I}, \tilde{\mathrm{N}}, \tilde{\mathrm{D}}, \tilde{\mathrm{C}}, \tilde{\mathrm{A}}, \tilde{\mathrm{R}}, \tilde{\mathrm{V}}_{\mathrm{DL}}$	3D U-net	IN	8	10	10.0	1e-2	1.0	1e-4	3d 8h / 8	11ms	$\tilde{\mathrm{V}}_{\mathrm{SVL}}(8)$
-	Rendering	$\tilde{\mathrm{N}}, \tilde{\mathrm{D}}, \tilde{\mathrm{C}}, \tilde{\mathrm{A}}, \tilde{\mathrm{R}}, \tilde{\mathrm{V}}_{\mathrm{SVL}},$	-	-	-	-	-	-	-	-	-	834ms	$\tilde{\mathrm{I}}(3)$

Table 3. The details of the network architecture, and training. Please refer to the main paper for the architecture of MVANet.

ceedings of the IEEE conference on computer vision and pattern recognition, pages 3431–3440, 2015. 2

[9] Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy networks: Learning 3d reconstruction in function space. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4460–4470, 2019. 2

[10] 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. *Communications of the ACM*, 65(1):99–106, 2021. 2

[11] Pratul P Srinivasan, Ben Mildenhall, Matthew Tancik, Jonathan T Barron, Richard Tucker, and Noah Snavely. Lighthouse: Predicting lighting volumes for spatially-coherent illumination. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8080–8089, 2020. 4

[12] Cheng Sun, Min Sun, and Hwann-Tzong Chen. Direct voxel grid optimization: Super-fast convergence for radiance fields reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5459–5469, 2022. 2

[13] Qianqian Wang, Zhicheng Wang, Kyle Genova, Pratul P Srinivasan, Howard Zhou, Jonathan T Barron, Ricardo Martin-Brualla, Noah Snavely, and Thomas Funkhouser. Ibrnet: Learning multi-view image-based rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4690–4699, 2021. 7, 8, 9, 10, 11

[14] Zian Wang, Jonah Philion, Sanja Fidler, and Jan Kautz. Learning indoor inverse rendering with 3d spatially-varying lighting. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 12538–12547, 2021. 4

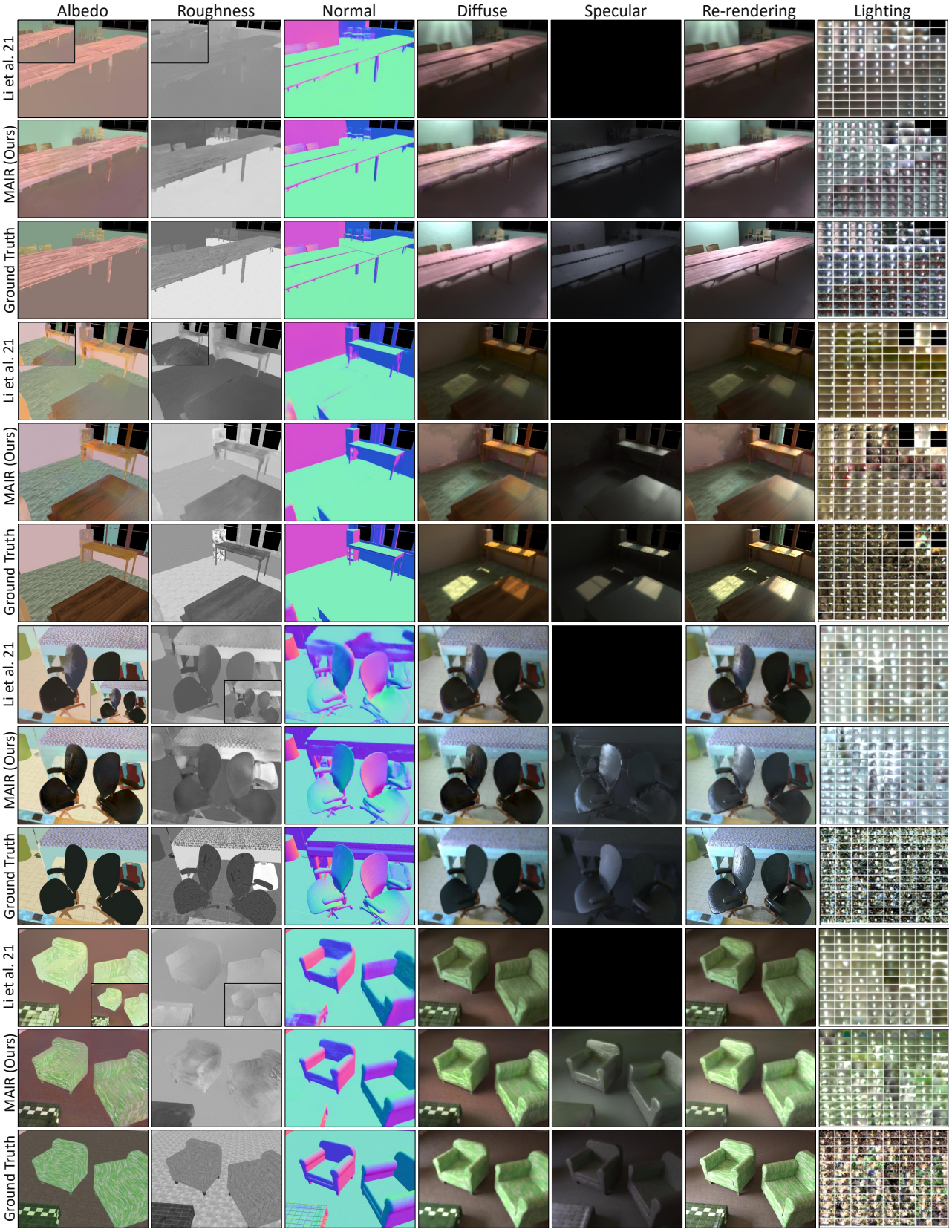

Figure 4. Additional inverse rendering results on OpenRooms FF. Small insets are the estimations without bilateral solver (BS).

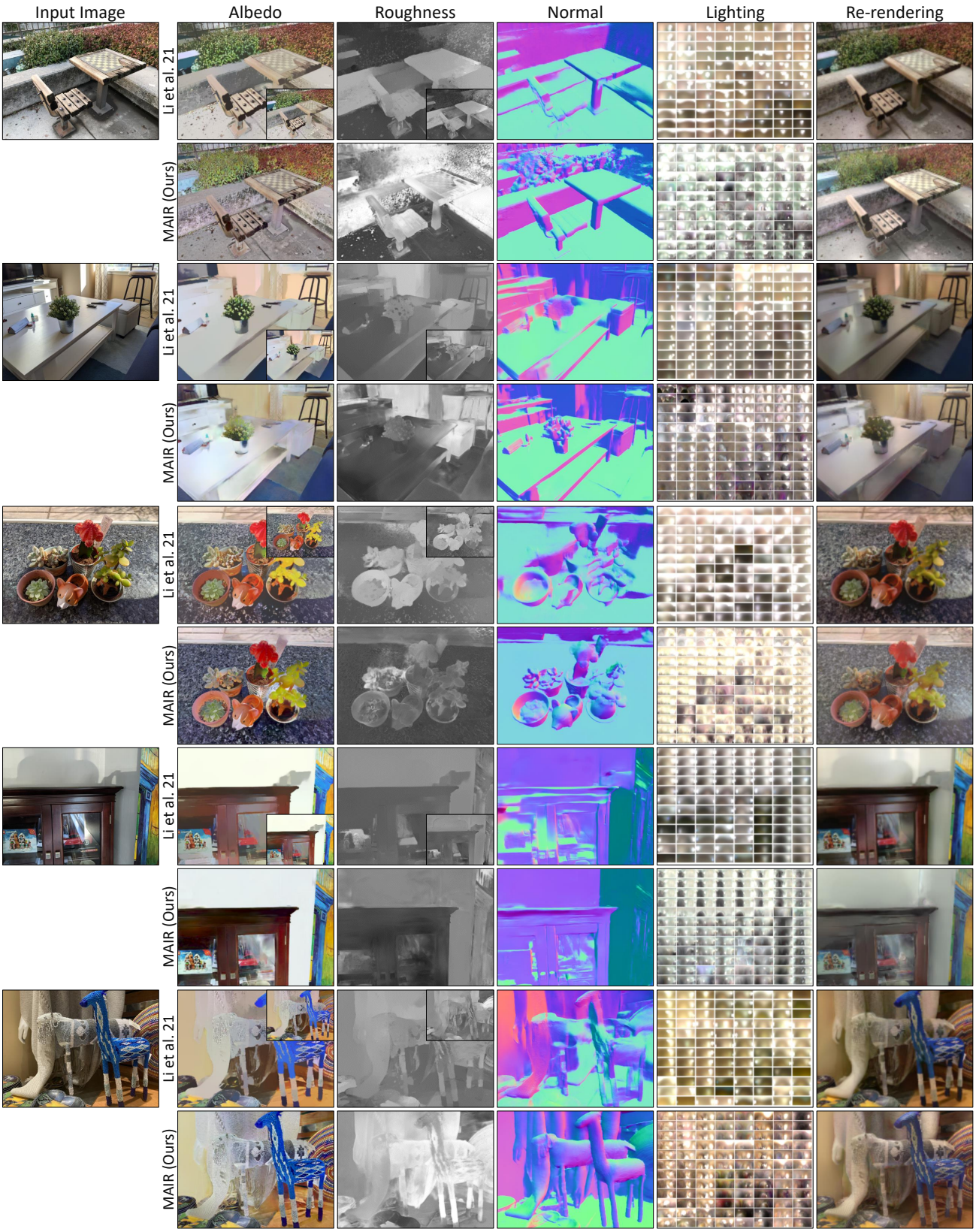

Figure 5. Additional inverse rendering results on IBRNet dataset [13]. Small insets are the estimations without BS.

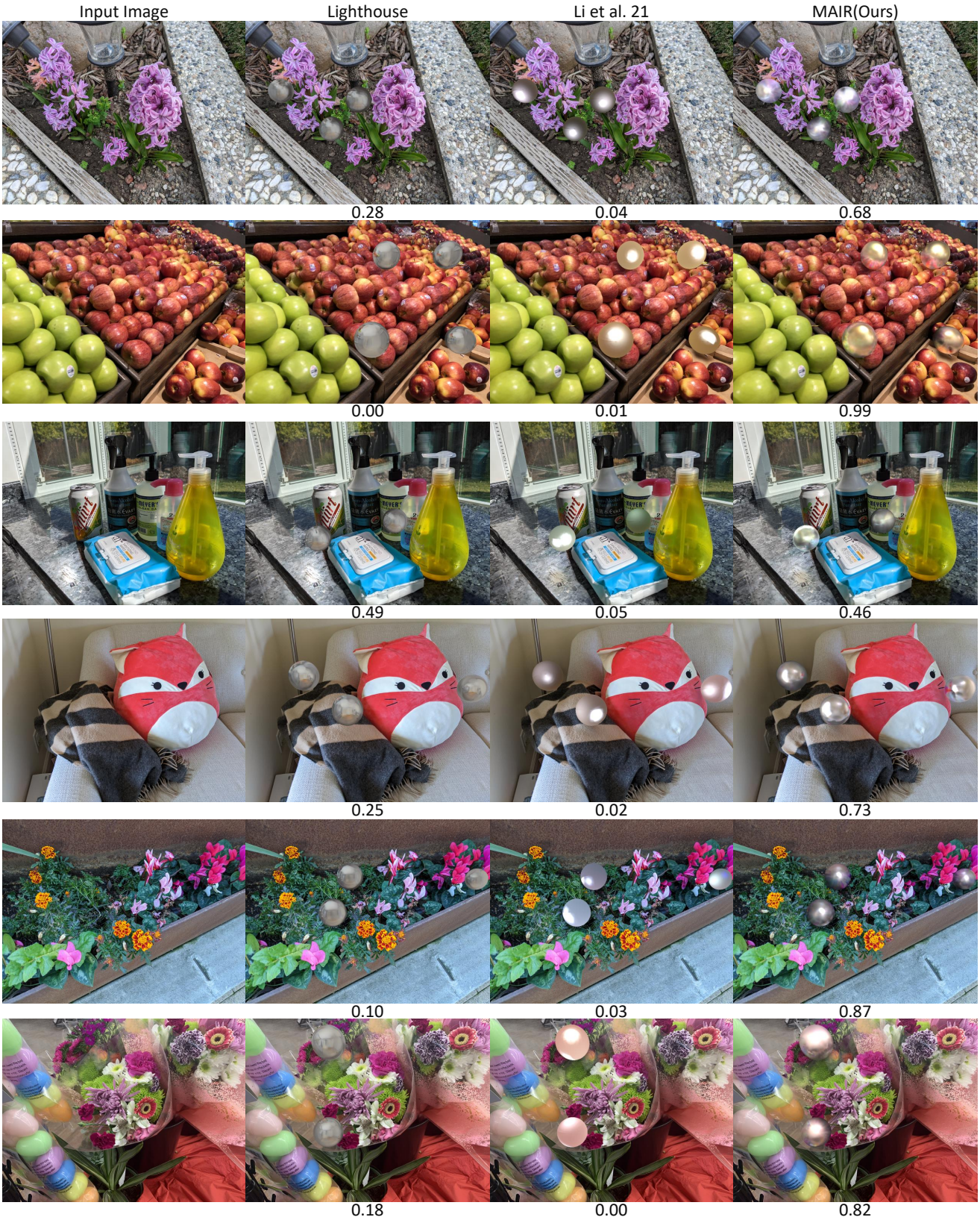

Figure 6. Additional chrome sphere insertion results on IBRNet dataset [13]. The number under the image is the result of user study.

Figure 7. Additional chrome sphere insertion results on IBRNet dataset [13]. The number under the image is the result of user study.

Figure 8. Additional white sphere insertion results on OpenRooms FF. The number under the image is the result of user study.

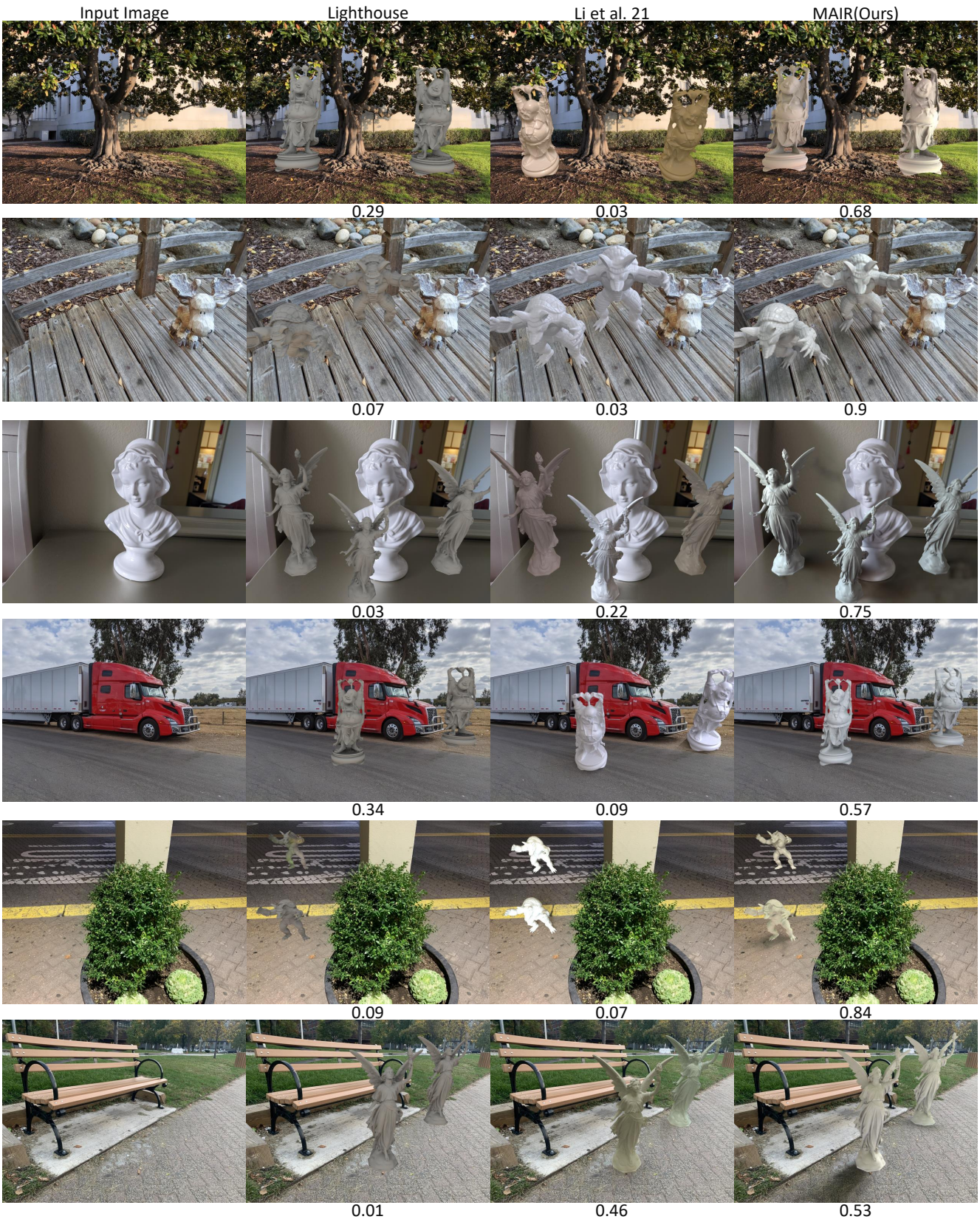

Figure 9. Additional virtual object [2] insertion results on IBRNet dataset [13]. The number under the image is the result of user study.

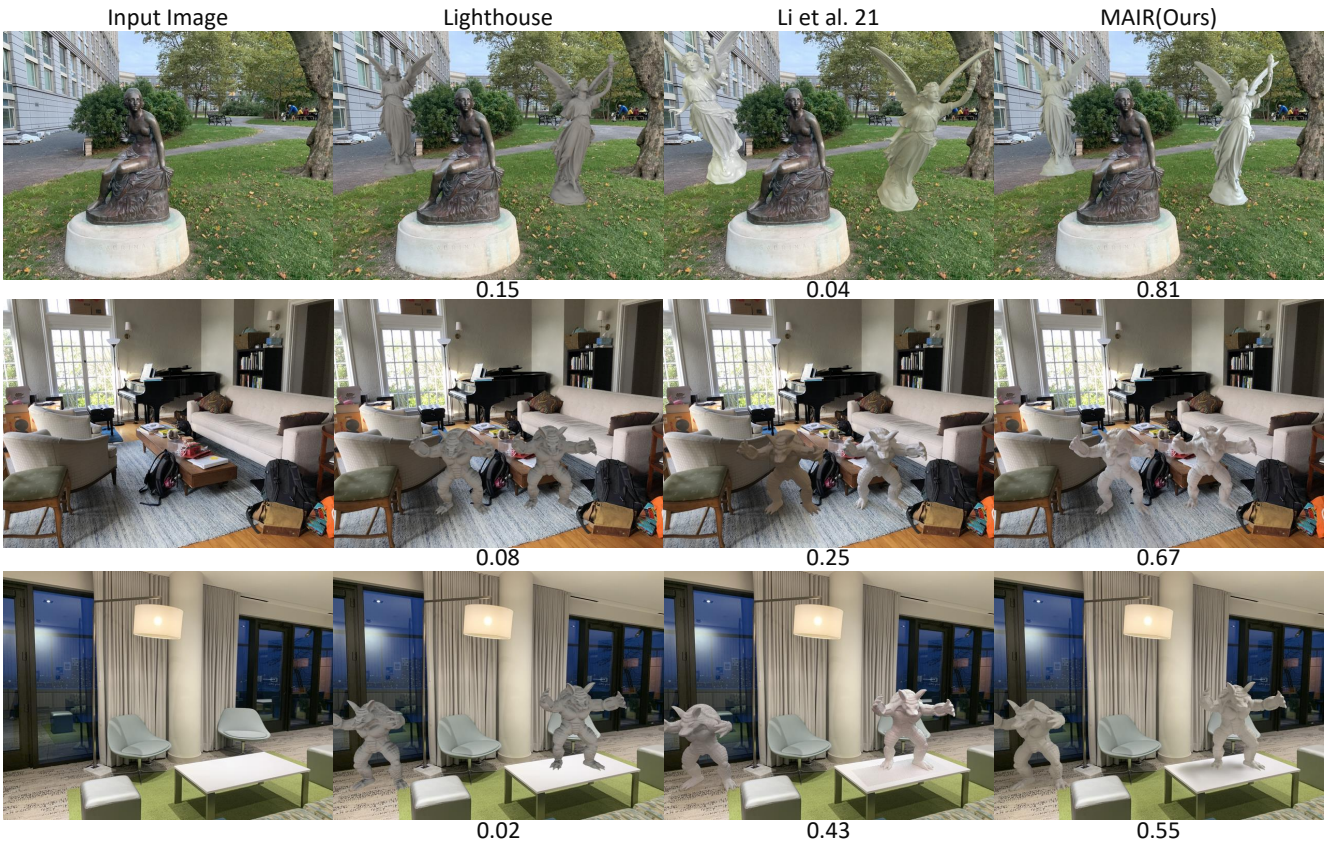

Figure 10. Additional virtual object [2] insertion results on IBRNet dataset [13]. The number under the image is the result of user study.