Spin-UP: Spin Light for Natural Light Uncalibrated Photometric Stereo
Supplementary Material

Zongrui Li^{1,2,*} Zhan Lu^{2,4,*,†} Haojie Yan^{3,4} Boxin Shi^{5,6} Gang Pan^{3,4} Qian Zheng^{3,4,‡} Xudong Jiang^{1,2}

^{1}Rapid-Rich Object Search (ROSE) Lab, Interdisciplinary Graduate Programme, Nanyang Technological University
^{2}School of Electrical and Electronic Engineering, Nanyang Technological University
^{3}College of Computer Science and Technology, Zhejiang University
^{4}The State Key Lab of Brain-Machine Intelligence, Zhejiang University
^{5}National Key Laboratory for Multimedia Information Processing, School of Computer Science, Peking University
^{6}National Engineering Research Center of Visual Technology, School of Computer Science, Peking University
{zongrui001,zhan007,EXDJiang}@ntu.edu.sg, {hjyan,gpan,qianzheng}@zju.edu.cn, shiboxin@pku.edu.cn

In this supplementary material,

1. we give more implementation details in Sec. 7, including details of framework structure (footnote 6) and hyperparameters setup (footnote 7).
2. we introduce more about boundary normal calculation and normal calculation for rendering equation in perspective projection in Sec. 8 (footnote 6);
3. we provide an overview of the synthetic and real-world dataset in Sec. 9. We also explain how we collect and preprocess the real-world dataset;
4. we showcase a qualitative comparison between Spin-UP and other methods on the real-world dataset in Sec. 10 (footnote 10). More results from the real-world dataset are also included in this section (footnote 10);

7. Implementation Details

7.1. Network Structure

We use the similar multi-layer perceptrons (MLPs)' structures in [4, 5], shown in Fig. 9. The input of MLPs is pixels' 2D coordinate ($p = (x, y)$) in an image, which will pass through a positional encoding module similar in [6] calculated as

$$\gamma(p) = \left(\sin\left(2^0\pi p\right), \cos\left(2^0\pi p\right), \cdots, \sin\left(2^{L_p-1}\pi p\right), \cos\left(2^{L_p-1}\pi p\right)\right), \tag{3}$$

where L_p is the positional code's dimension, set as 10 for $\gamma^1(.)$ and 6 for $\gamma^2(.)$

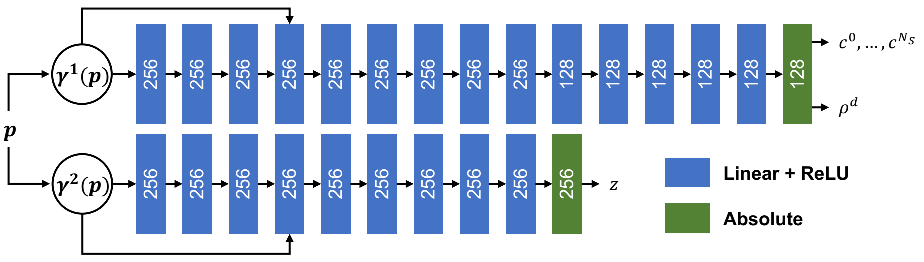

Figure 9. Network structure of MLPs for depth and material estimation in Spin-UP.

*Co-first author. ‡Corresponding author.
†Work completed while interning at the State Key Lab of Brain-Machine Intelligence, Zhejiang University.

7.2. Loss Functions and Hyperparameters Setup

In Spin-UP, we implement:

1. L1 inverse rendering loss L_r calculated as, $\sum_{i=1}^{N_P}\sum_{j=1}^{N_I}|\boldsymbol{m}_{ij} - \hat{\boldsymbol{m}}_{ij}|$.
2. Normalized color loss L_{color}, calculated as, $\lambda_c\|\text{Nor}(\boldsymbol{A}) - \text{Nor}(\boldsymbol{I})\|$, where $\lambda_c = 0.5$.
3. Boundary loss L_{b}, calculated as the cosine similarity between the pre-computed and estimated boundary normal.
4. Smoothness terms L_{sm} on albedo map $\boldsymbol{A}$, normal map $\boldsymbol{N}$, spatially varying Gaussian bases weights c^n, is calculated as,

$$L_{\text{sm}} = \frac{\lambda}{N_P}\sum_{i=1}^{N_P}\left|\frac{\partial \boldsymbol{A}}{\partial x} + \frac{\partial \boldsymbol{A}}{\partial y}\right| + \frac{\lambda_N}{N_P}\sum_{i=1}^{N_P}\left|\frac{\partial \boldsymbol{N}}{\partial x} + \frac{\partial \boldsymbol{N}}{\partial y}\right| + \frac{\lambda_S}{N_P}\sum_{n=1}^{N_S}\sum_{i=1}^{N_P}\left|\frac{\partial c_i^n}{\partial x} + \frac{\partial c_i^n}{\partial y}\right|, \tag{4}$$

where, $\lambda = 0.01$, $\lambda_N = 0.02$, $\lambda_S = 0.01$.

We train the Spin-UP in three stages similar to [5]. For the first stage, the loss $\mathcal{L}_{\text{stage1}}$ is calculated as below for a faster convergence.

$$\mathcal{L}_{\text{stage1}} = \mathcal{L}_r + \mathcal{L}_{\text{b}} + \lambda_c\mathcal{L}_{\text{color}} + \mathcal{L}_{\text{sm}}, \tag{5}$$

For the second stage, we drop the smoothness term on the albedo map and reduce λ_N to 0.05 for details refinement, where $\mathcal{L}_N = \text{TV}(\boldsymbol{N})$ is the smoothness term on normal map.

$$\mathcal{L}_{\text{stage2}} = \mathcal{L}_r + \mathcal{L}_{\text{b}} + \lambda_c\mathcal{L}_{\text{color}} + \lambda_N\mathcal{L}_N, \tag{6}$$

For the third stage, we drop the smoothness terms $\mathcal{L}_N$ to further refine the details.

$$\mathcal{L}_{\text{stage3}} = \mathcal{L}_r + \mathcal{L}_{\text{b}} + \lambda_c\mathcal{L}_{\text{color}}. \tag{7}$$

The three stages take 500, 1000, and 500 epochs, respectively. During training, we use Adam as the optimizer with a learning rate $\alpha = 0.001$ and a batch size of 4 images per iteration.

8. Normal Calculation in Perspective View

8.1. Boundary Normal Calculation

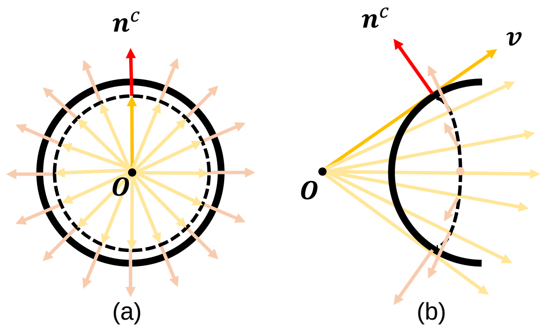

Figure 10. An illustration of occluding boundaries' normal relationship with view directions for (a) front view and (b) side view of a surface. The dotted line in (b) indicates the outermost boundaries of an object in perspective projection.

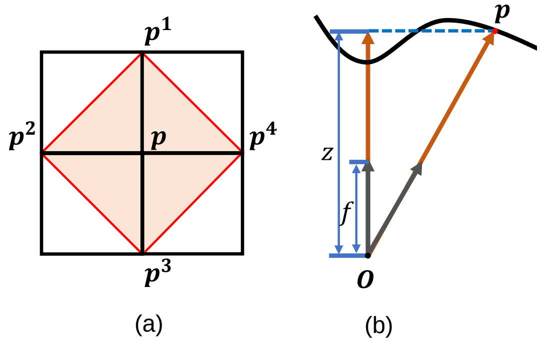

Figure 11. An illustration of (a) adjacent points' positions for normal fitting method [5] in perspective projection, (b) Eq. (11).

In perspective projection, the surface normal is perpendicular to the object's occluding boundaries $B(x, y)$ and view direction $\boldsymbol{v}$, as shown in Fig. 10. Therefore, the boundaries' normal $\boldsymbol{n}^b$ is calculated as

$$\boldsymbol{n}^b \cdot \boldsymbol{v}^b = 0,\ \boldsymbol{n}^b \cdot (\frac{\partial B}{x}, \frac{\partial B}{y}, 1)^\top = 0, \tag{8}$$

In practice, the outer boundaries of an object in images may not precisely match its actual boundaries due to limited image resolution. Therefore, we add a small offset ($\beta = 0.1$) to make the pre-computed boundaries normal more accurate:

$$\boldsymbol{n}^b = \text{Nor}(n^{bx}, n^{by}, n^{bz} + \beta). \tag{9}$$

Table 5. Length, width, height, and capturing distance for SOLDIER, PLAYER, DANCER, POLICEMAN and EEVEE.

Properties	SOLDIER	PLAYER	DANCER	POLICEMAN	EEVEE
Length (cm)	9.50	11.50	4.00	4.00	4.00
Width (cm)	7.00	11.00	5.00	4.00	4.00
Height (cm)	3.00	28.00	4.00	9.00	9.00
Distance (m)	0.90	0.90	0.40	0.40	0.30

8.2. Normal Calculation For Rendering Equation

The normal fitting method [5] in orthogonal projection is shown below:

$$\begin{aligned} \boldsymbol{n} &= \sum_{k=1}^{4} \gamma^k \boldsymbol{n}^k = \sum_{k=1}^{4} \gamma^k \,\mathrm{Nor}\left[\left(\boldsymbol{p}^{k+1} - \boldsymbol{p}\right) \times \left(\boldsymbol{p}^k - \boldsymbol{p}\right)\right]^\top, \\ \gamma^k &= \frac{\left|d^k\right|^{-1}}{\sum_{k=1}^{4} \left|d^k\right|^{-1}}, \quad d^k = z^k + z^{k+1} - 2z, \end{aligned} \tag{10}$$

where, $\boldsymbol{p}^k = (x^k, y^k, z^k)$ is the adjacent point of the query point $\boldsymbol{p} = (x, y, z)$ and $x, y \in [-1, 1]$, $k = 1$ if $k + 1 > 4$, as shown in Fig. 11, (a). To extend the normal fitting method to the perspective projection, we first compute the points' coordinates in the camera coordinate system by

$$\begin{aligned} \boldsymbol{p}^{k\prime} &= (x^k \frac{z^k}{f} s_x, y^k \frac{z^k}{f} s_y, z^k), \\ \boldsymbol{p}' &= (x \frac{z}{f} s_x, y \frac{z}{f} s_y, z). \end{aligned} \tag{11}$$

where f is the camera's focal, s_x and s_y are half of the width and height of the camera's frame. Replace $\boldsymbol{p}^k$ and $\boldsymbol{p}$ in Eq. (10) by $\boldsymbol{p}^{k\prime}$ and $\boldsymbol{p}'$, we get the normal fitting method in perspective projection.

9. Datasets

9.1. Synthetic Dataset

In Fig. 12, we showcase all 5 objects with 6 materials under 5 HDR environment maps rendered by Blender Cycles[1]. This results in 16 scenes[2] of synthetic data that are classified into 4 groups, *i.e*, the shape group, light group, reflectance group, and spatially varying group.

9.2. Real-world Dataset

The real-world dataset contains 5 objects captured under indoor and outdoor environments with spatially-varying materials. The five real-world objects used in our study are the SOLDIER, PLAYER, POLICEMAN, DANCER, and EEVEE. The objects' sizes are shown in Table 5.

Device introduction. SOLDIER, PLAYER, POLICEMAN, and DANCER's observed images were captured by a customized device shown in Fig. 13 (left), which consists of two stands (one for holding the subject being photographed, the other for supporting the camera) and a rotating mechanism. The distance from the camera to the object is adjustable. In addition to this, we also consider a more portable device shown in Fig. 13 (right), which is made up of a wooden rotatable platform[3] with a diameter of 39mm and the camera. We capture EEVEE's observed images based on this device.

Photographing requirements. Before photographing, the distance between the camera and the object is determined based on the proportion of the object in the viewfinder, ensuring a balance of the occupied portion between the objects and the camera. Three typical distances were used: 0.9 meters for large and 0.4 meters (or 0.3 meters) for small objects. During photographing, the thumb rule is to capture a clear image with less noise and keep rotation velocity as uniform as possible. For the camera's parameters, we chose ISO 1600, an aperture size of f/13 for outdoor scenes; and ISO 3200, an aperture size of f/6.3 for indoor scenes, respectively. The focal size is fixed at 31mm for different scenes.

[1] https://www.blender.org

[2] One scene representing an object with one material rendered under HDR environment maps.

[3] https://www.ikea.com/sg/en/p/snudda-lazy-susan-solid-wood-40176460/

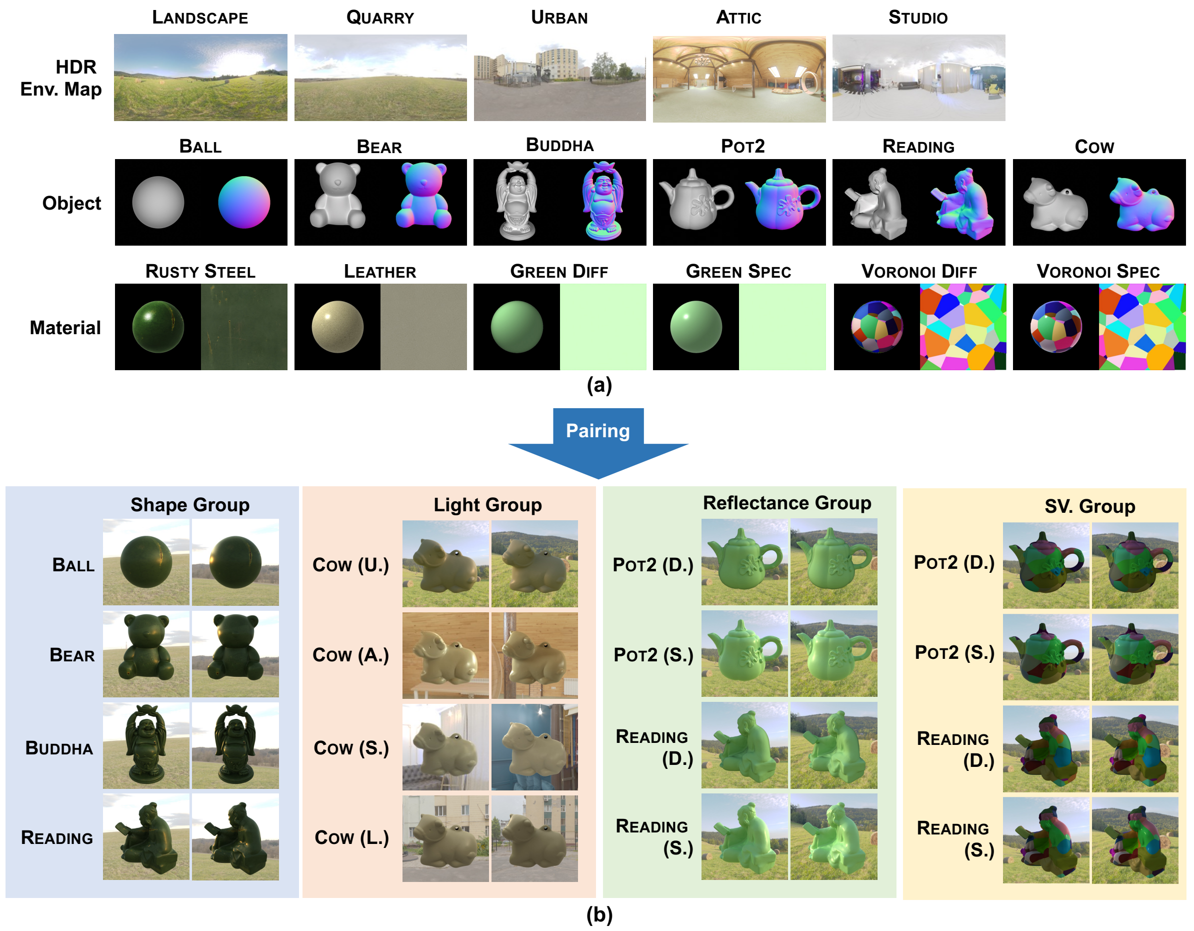

Figure 12. (a) HDR environment maps (row 1), objects (row 2), and materials (row 3) involved in the synthetic dataset. Each figure in row 2 consists of two subfigures for 3D model preview (left) and normal map (right). Each figure in row 3 consists of two subfigures for material rendered on a sphere (left) and albedo (right). (b) Example images from each scene in four groups.

Pre-processing pipeline. In the pre-processing pipeline, we extracted 50 images from the video at equal intervals to use as our data. We then obtain the objects' masks in each scene from the first frame by Photoshop. Those masks help separate objects and backgrounds. In practice, there are translational motions in the horizontal and vertical directions, mostly obvious on objects due to structural instability. Therefore, after calculating the relative rotation angle θ_j, we used a simple algorithm for motion correction, assuming that the only motion of the object relative to the camera was translational in the horizontal and vertical directions. Specifically, we pre-set the range of motion and iterate over the distance vector to find the distance of movement (plus or minus 20 pixels) that minimizes the difference between the front and back frames after applying the mask. Note that although large movement is corrected in this step, minor movements still exist and are hard to eliminate. Fortunately, our method can tolerate those minor movements.

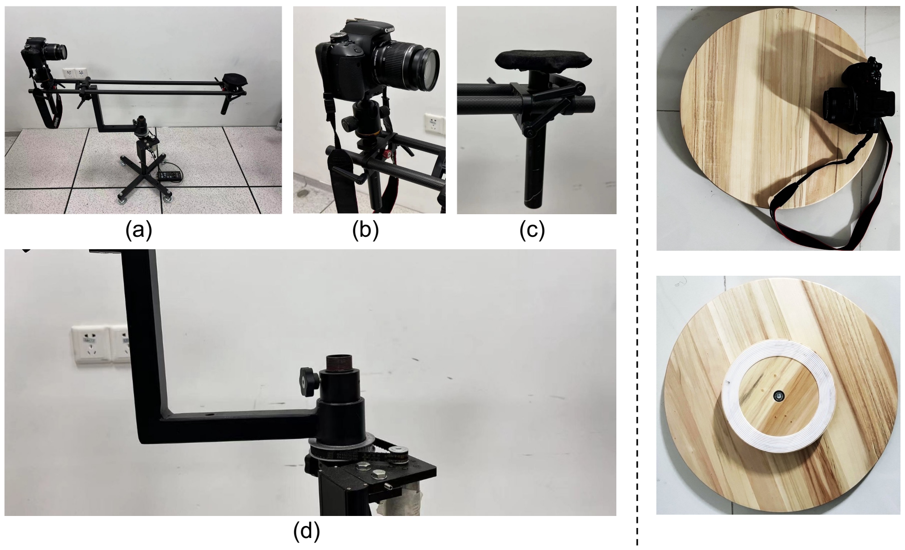

Figure 13. Left: (a) Overview of the device, (b) Stand for the camera, (c) Stand for the object being photographed with dark cloth for interreflection removal, (d) Rotating hinge. Right: A portable version of image capturing device, shown in top and bottom views.

10. Qualitative Comparison

10.1. Qualitative Comparison on Synthetic Dataset

We show all the estimated normal maps, error maps of Spin-UP, S23 [3], S22 [2], and HY19 [1] of shape, light, reflectance, and spatially-varying material groups in Fig. 15-Fig. 17.

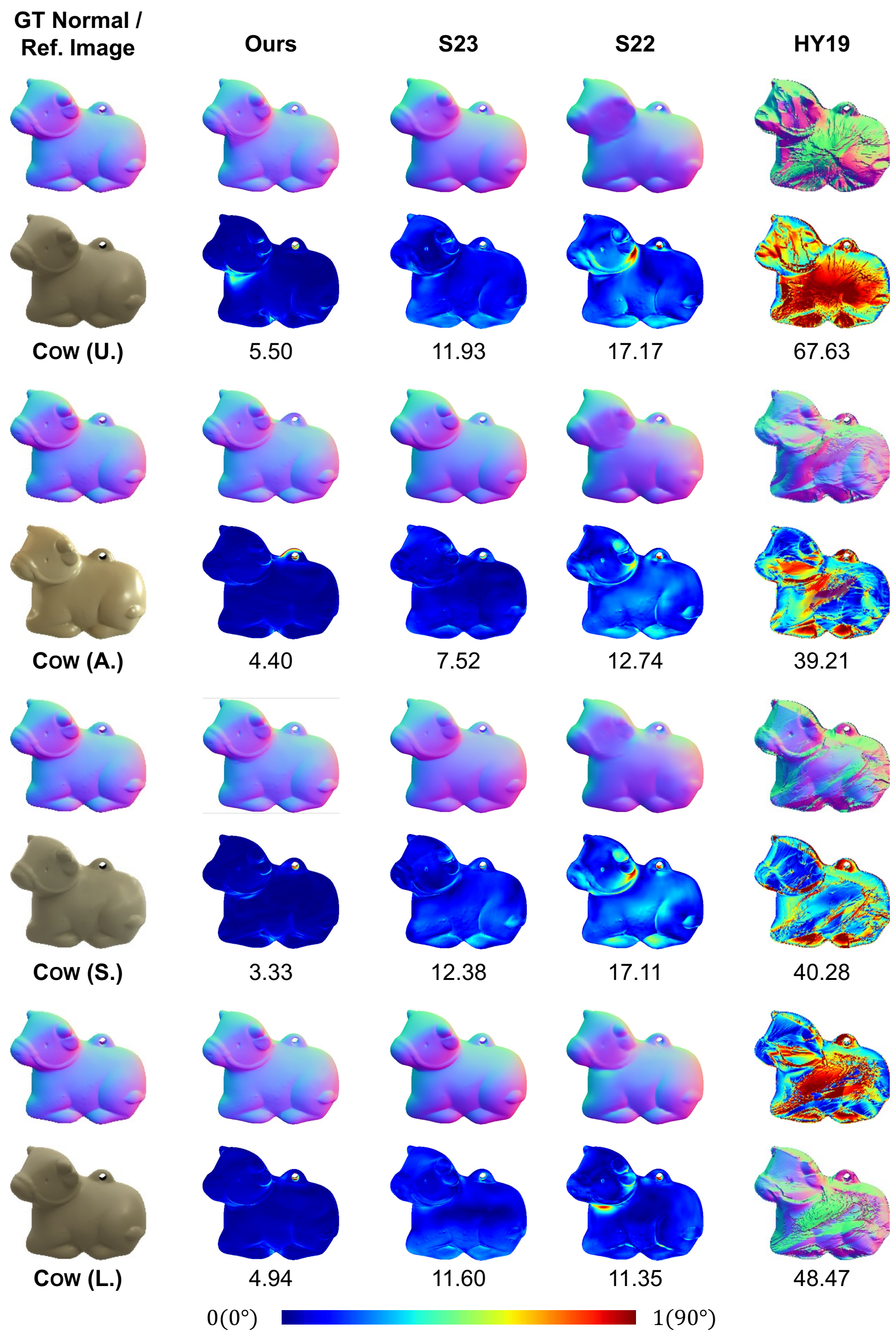

Figure 14. The visual quality comparison among Spin-UP, S23 [3], S22 [2], and HY19 [1] on the light group in terms of normal map (row 1, 3, 5, 7), error map (row 2, 4, 6, 8). Numbers indicate the MAE for surface normal.

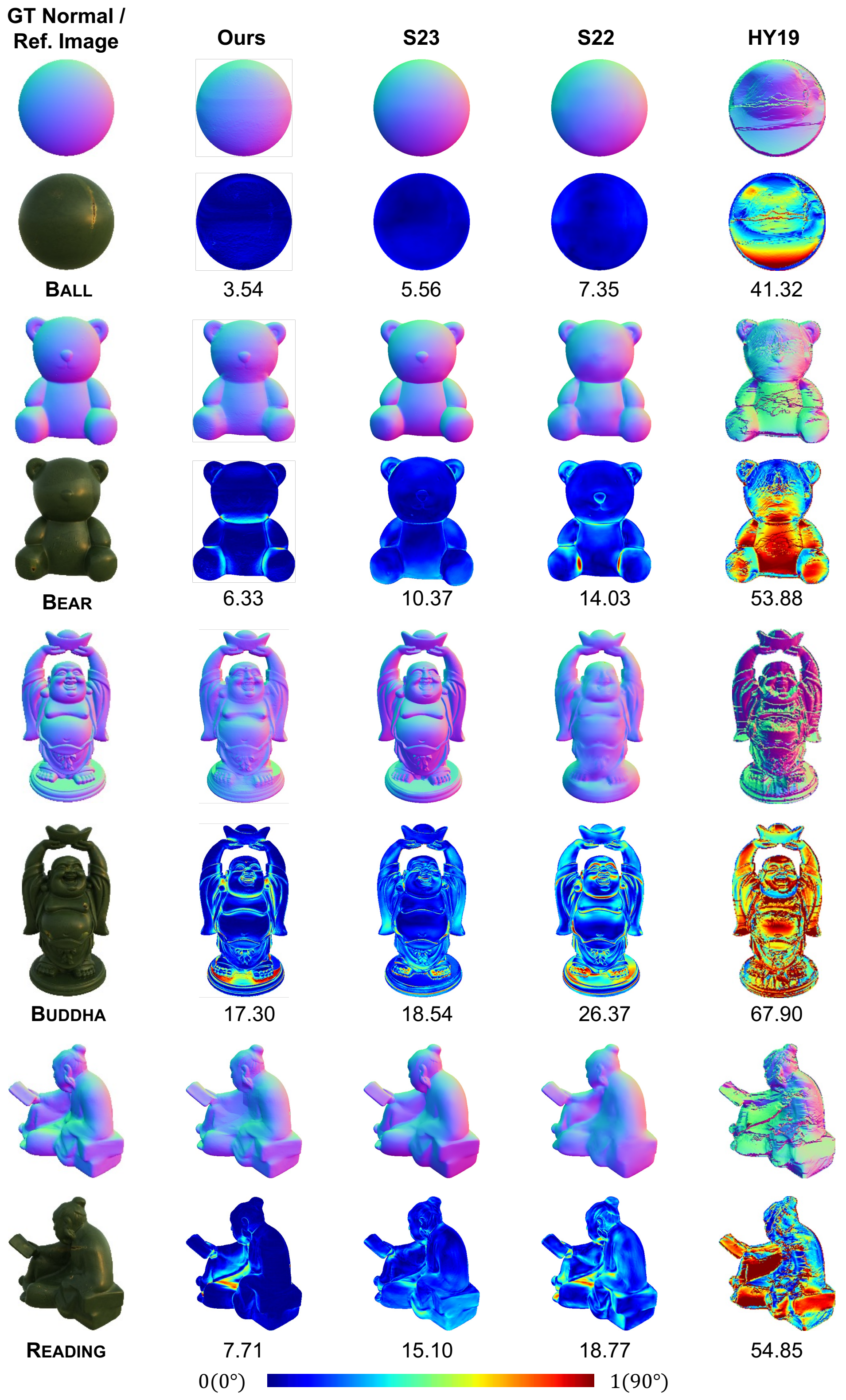

Figure 15. The visual quality comparison among Spin-UP, S23 [3], S22 [2], and HY19 [1] on the shape group in terms of normal map (row 1, 3, 5, 7), error map (row 2, 4, 6, 8). Numbers indicate the MAE for surface normal.

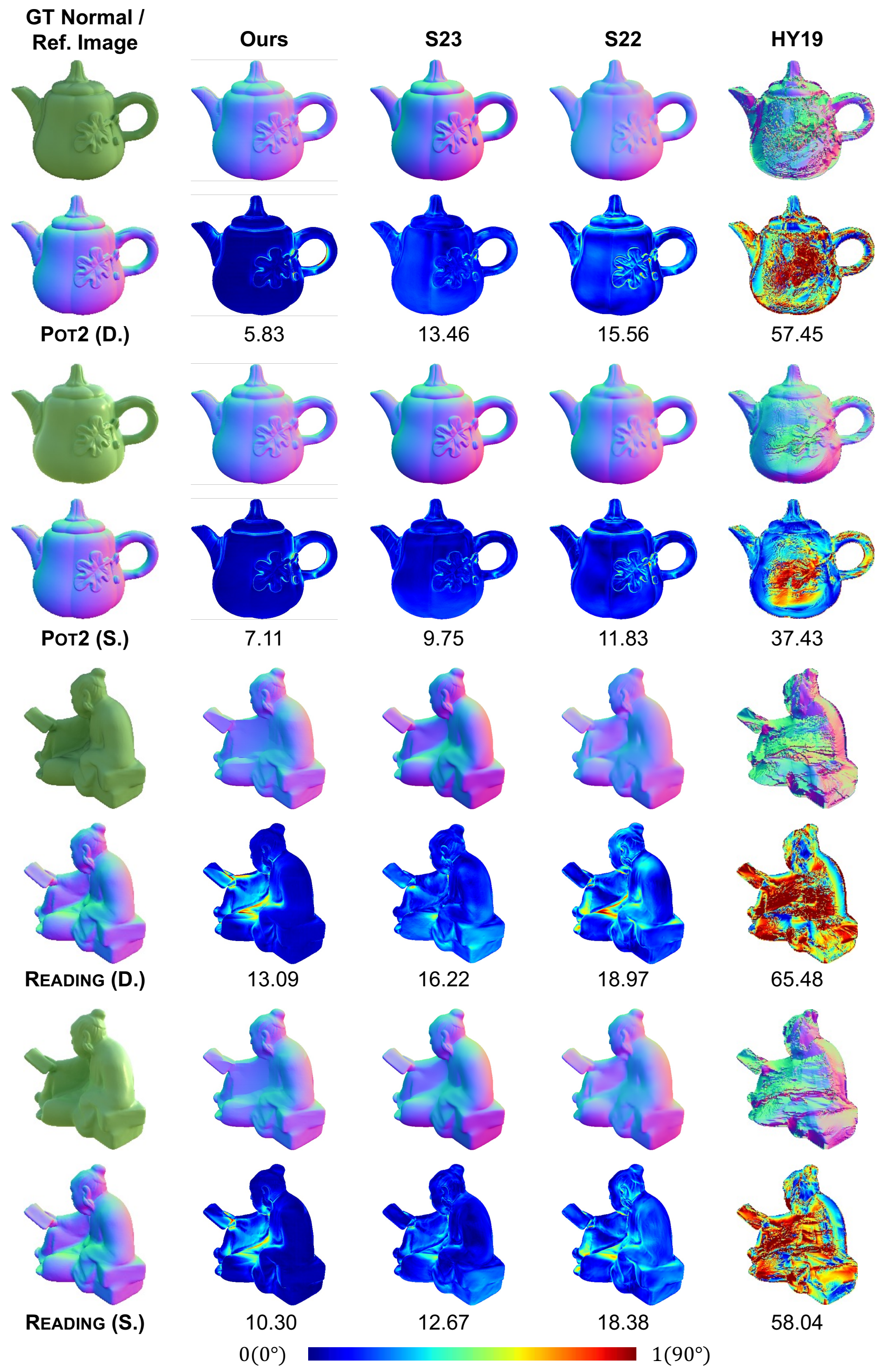

Figure 16. The visual quality comparison among Spin-UP, S23 [3], S22 [2], and HY19 [1] on the reflectance group in terms of normal map (row 1, 3, 5, 7), error map (row 2, 4, 6, 8). Numbers indicate the MAE for surface normal.

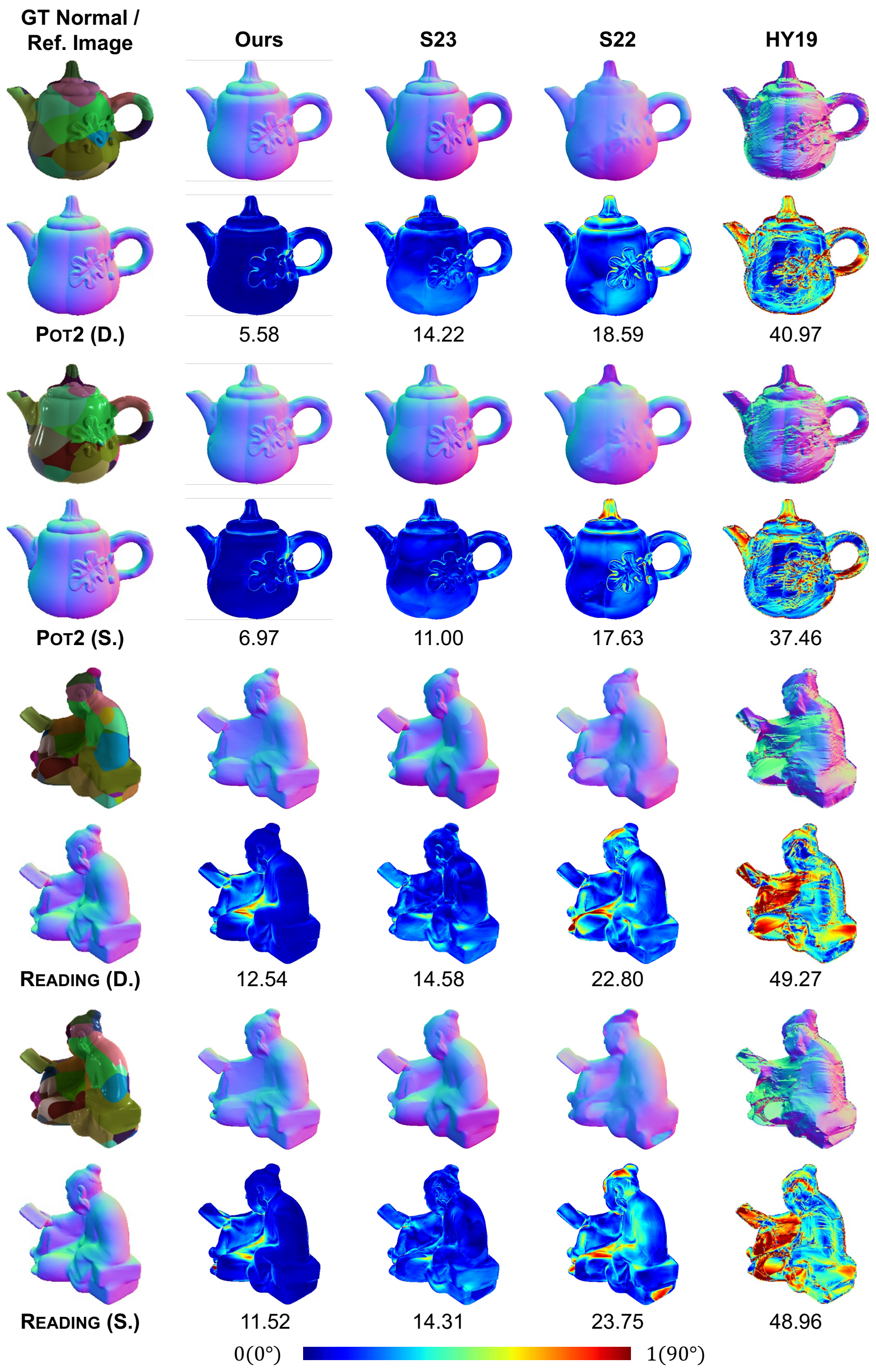

Figure 17. The visual quality comparison among Spin-UP, S23 [3], S22 [2], and HY19 [1] on the spatially varying material group in terms of normal map (rows 1, 3, 5, 7), error map (rows 2, 4, 6, 8). Numbers indicate the MAE for surface normal.

10.2. Qualitative Comparison on Real-world Dataset

We show all the estimated normal maps of Spin-UP, S23 [3], and S22 [2] of real-world dataset in Fig. 18 and Fig. 19.

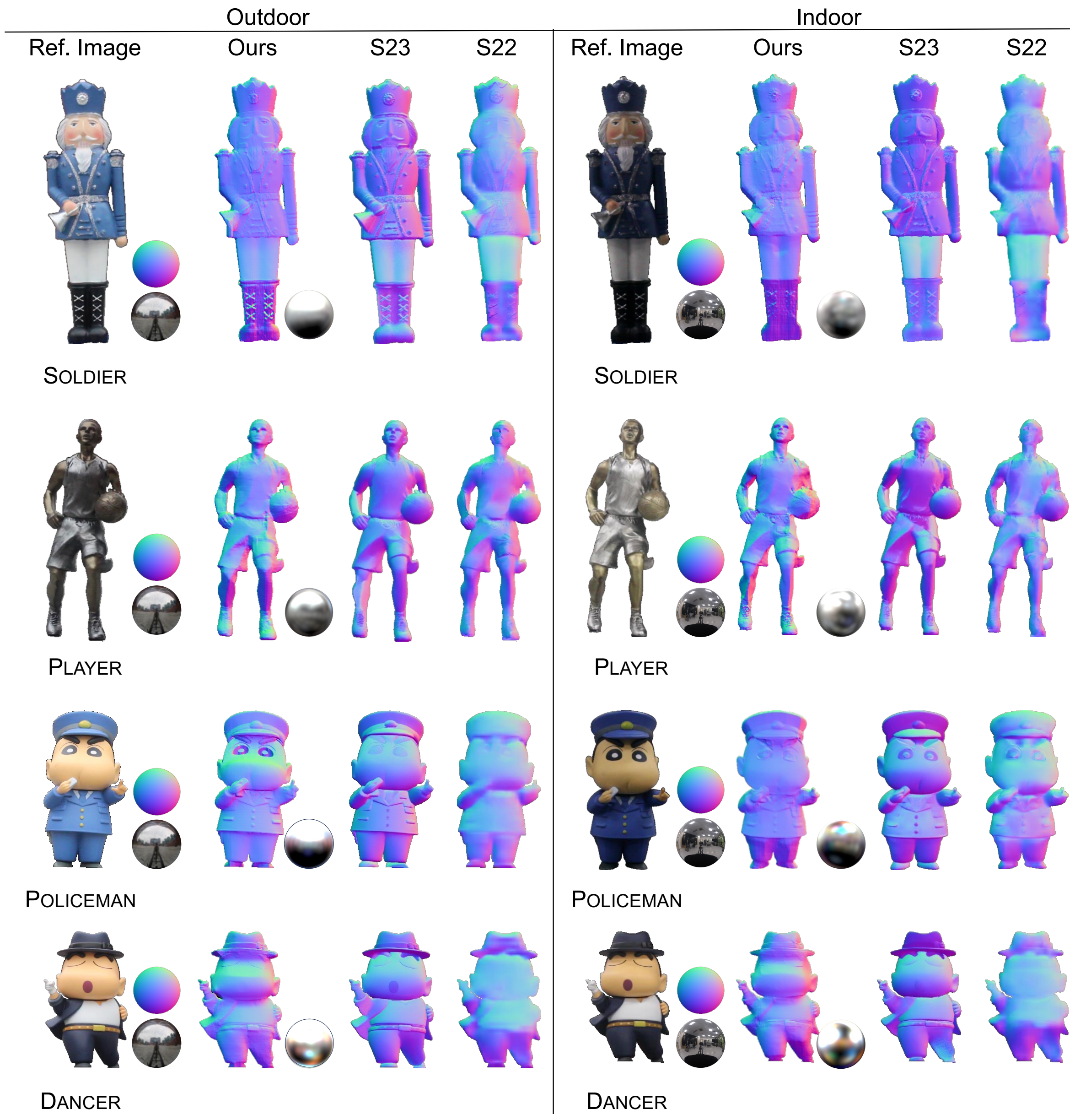

Figure 18. The visual quality comparison among Spin-UP, S23 [3], and S22 [2] on the SOLDIER, PLAYER, POLICEMAN, and DANCER in terms of the normal map. Left (right) side of the solid line: objects captured in CAMPUS (WORKPLACE) environment.

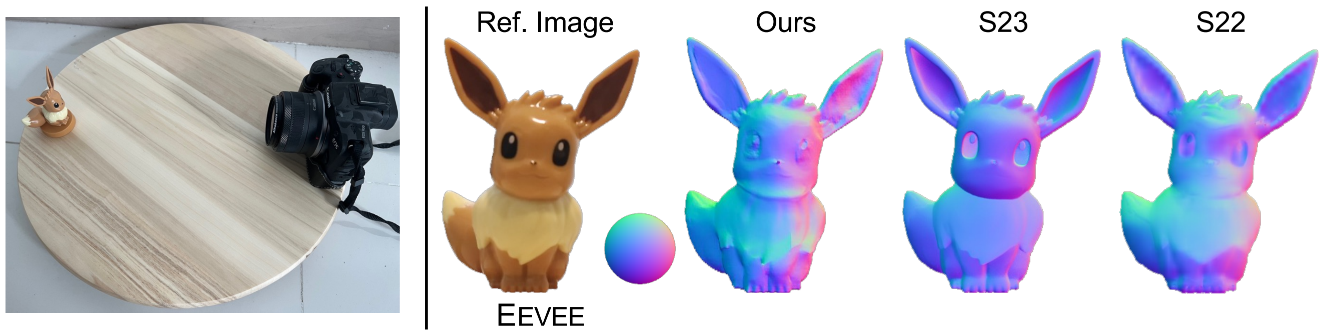

Figure 19. The visual quality comparison among Spin-UP, S23 [3], and S22 [2] on EEVEE captured in a living room in terms of the normal map based on more portable device.

References

[1] Bjoern Haefner, Zhenzhang Ye, Maolin Gao, Tao Wu, Yvain Quéau, and Daniel Cremers. Variational uncalibrated photometric stereo under general lighting. In *Proc. International Conference on Computer Vision (ICCV)*, 2019. 6, 7, 8, 9

[2] Satoshi Ikehata. Universal photometric stereo network using global lighting contexts. In *Proc. Computer Vision and Pattern Recognition (CVPR)*, 2022. 6, 7, 8, 9, 10, 11

[3] Satoshi Ikehata. Scalable, detailed and mask-free universal photometric stereo. In *Proc. Computer Vision and Pattern Recognition (CVPR)*, 2023. 6, 7, 8, 9, 10, 11

[4] Junxuan Li and Hongdong Li. Self-calibrating photometric stereo by neural inverse rendering. In *Proc. European Conference on Computer Vision (ECCV)*, 2022. 1

[5] Zongrui Li, Qian Zheng, Boxin Shi, Gang Pan, and Xudong Jiang. DANI-Net: Uncalibrated photometric stereo by differentiable shadow handling, anisotropic reflectance modeling, and neural inverse rendering. *Proc. Computer Vision and Pattern Recognition (CVPR)*, 2023. 1, 2, 3

[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 *Proc. European Conference on Computer Vision (ECCV)*, 2020. 1