Multi-View Azimuth Stereo via Tangent Space Consistency
Supplementary material

Xu Cao Hiroaki Santo Fumio Okura Yasuyuki Matsushita
Osaka University
{cao.xu, santo.hiroaki, okura, yasumat}@ist.osaka-u.ac.jp
Source code: https://github.com/xucao-42/mvas

This supplementary material provides more details and analysis of our method, as listed below.

Contents

A. Analysis of TSC loss

This section provides more details about our modification for TSC loss to account for the $\pm\frac{\pi}{2}$ ambiguity in polarimetric azimuth observations, discusses the necessity of considering multi-view consistency, and provides more details and an efficiency analysis of our visibility determination strategy.

A.1. Accounting for $\pm\frac{\pi}{2}$ ambiguity in TSC loss

We modify our TSC loss to account for $\pm\frac{\pi}{2}$ ambiguity in polarimetric observations. Given an observed polarimetric phase angle $\hat{\phi}$, the surface azimuth angle ϕ is either $\hat{\phi} \pm \frac{\pi}{2}$ or $\hat{\phi} (= \hat{\phi} + \pi)$ depending on whether the surface point is polarimetric specular or diffuse reflection dominated [4, 13]. Unfortunately, labeling the specular or diffuse domination is non-trivial [4, 5, 19, 20]. In our approach, although TSC is invariant to π ambiguity, the $\pm\frac{\pi}{2}$ ambiguity still requires specific handling for polarimetric observations.

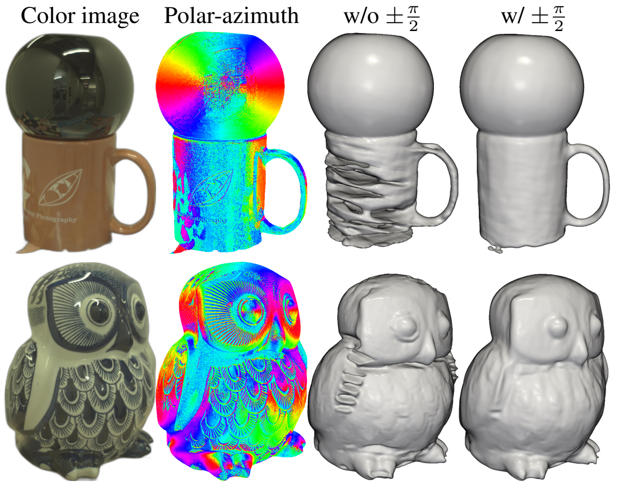

Figure 11. Accounting for the $\pm\frac{\pi}{2}$ ambiguity in TSC loss resolves the *twisted* surface problem. The results by dealing with the $\pm\frac{\pi}{2}$ ambiguity are presented in the main paper Fig. 11.

Our idea is to allow both possibilities in the TSC loss. The $\pm\frac{\pi}{2}$ ambiguity introduces one more candidate tangent vector, and the surface normal should be perpendicular to either of the vectors deduced from π or $\pm\frac{\pi}{2}$ phase angles. By main paper's Eq. (6), the projected tangent vector $\mathbf{t}'$ from the $\pm\frac{\pi}{2}$ phase angle is

$$\begin{aligned}\mathbf{t}'(\phi) = \mathbf{t}\left(\hat{\phi} + \frac{\pi}{2}\right) &= \mathbf{r}_1 \sin\left(\hat{\phi} + \frac{\pi}{2}\right) - \mathbf{r}_2 \cos\left(\hat{\phi} + \frac{\pi}{2}\right) \\ &= -\mathbf{r}_1 \cos\left(\hat{\phi}\right) - \mathbf{r}_2 \sin\left(\hat{\phi}\right).\end{aligned} \tag{21}$$

Because $\mathbf{t}'$ is also parallel to the image plane, $\mathbf{t}'$ can be obtained by rotating $\mathbf{t}$ by $\pm\frac{\pi}{2}$ in the image plane. At this point, however, we cannot fully determine which vector, $\mathbf{t}$ or $\mathbf{t}'$, is the actual tangent vector. We only know that the surface normal is perpendicular to either of the vectors:

$$\mathbf{n} \perp \mathbf{t} \quad \text{or} \quad \mathbf{n} \perp \mathbf{t}'. \tag{22}$$

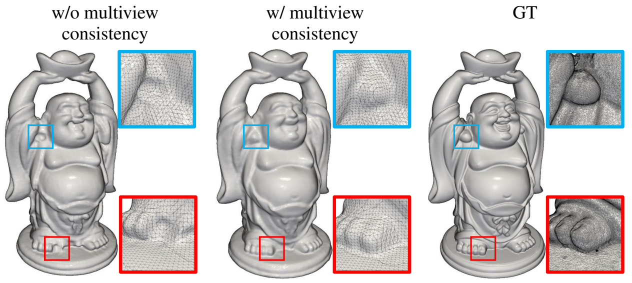

Figure 12. Considering multiview consistency resolves the convex-concave ambiguity because it encourages accurate correspondence.

Putting together the notations in the main paper's Eqs. (12) and (17), we can rewrite our TSC loss as

$$\mathcal{L}_{\text{TSC}} = \frac{1}{P}\sum_{\mathbf{x}\in\mathbf{X}} \frac{\sum_{i=1}^{C} \Phi_i \left(\mathbf{n}^\top \mathbf{t}_i\right)^2}{\sum_{i=1}^{C} \Phi_i}. \tag{23}$$

Based on Eq. (22), we modify Eq. (23) as

$$\mathcal{L}'_{\text{TSC}} = \frac{1}{P}\sum_{\mathbf{x}\in\mathbf{X}} \frac{\sum_{i=1}^{C} \Phi_i \left(\mathbf{n}^\top \mathbf{t}_i\right)^2 \left(\mathbf{n}^\top \mathbf{t}'_i\right)^2}{\sum_{i=1}^{C} \Phi_i}. \tag{24}$$

The modified TSC loss allows the surface normal to be perpendicular to either of the two candidate tangent vectors.

Figure 11 shows that this strategy yields better reconstruction quality, which gives us the results presented in the main paper's Fig. 10. If we do not deal with $\pm\frac{\pi}{2}$ ambiguity, the recovered shapes appear twisted due to wrong tangent vectors (*i.e.*, rotated by $\pm\frac{\pi}{2}$ from actual tangent vectors in the image space).

A.2. Ablation study on multi-view consistency

Accumulating projected tangent vectors from all visible views to compute the TSC loss is necessary for accurate shape recovery. Without considering multi-view consistency, we can simplify our original TSC loss from Eq. (23) to

$$\mathcal{L}''_{\text{TSC}} = \frac{1}{P}\sum_{\mathbf{x}\in\mathbf{X}} (\mathbf{n}(\mathbf{x})^\top \mathbf{t}(\phi(\Pi(\mathbf{x}))))^2, \tag{25}$$

where the projected tangent vector $\mathbf{t}$ is computed from the input pixel location, and visibility or tangent vectors in other views need no longer be considered.

This simplified loss Eq. (25), however, can lead to convex-concave ambiguity in the recovered surfaces, as shown in Fig. 12. Without multi-view consistency, the tangent vector from one view can only constrain the surface normal loosely on a plane and cannot constrain the surface positions correctly. Therefore, locally concave or convex surfaces with the same tangent vectors can both minimize the simplified loss, thus resulting in the ambiguity.

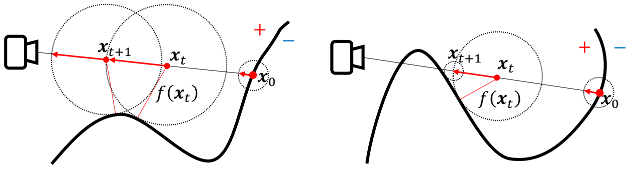

Figure 13. Visibility determination via reverse sphere tracing. We march a surface point $\mathbf{x}_0$ towards the camera center. At each step, the marching distance is the signed distance $f(\mathbf{x}_t)$ from the current point $\mathbf{x}_t$ to the surface, which requires one MLP evaluation. **(Left)** The marching diverges quickly towards the camera if $\mathbf{x}_0$ is visible. **(Right)** The marching converges to another surface point as ordinary sphere tracing [6] if $\mathbf{x}_0$ is occluded.

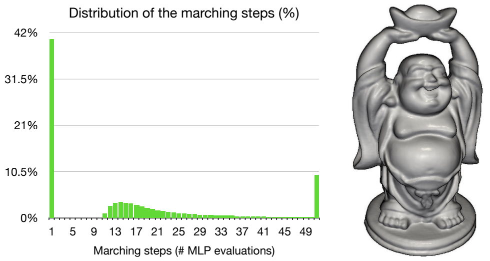

Figure 14. The distribution of marching steps required to determine the visibility of surface points of the DiLiGenT-MV object "Buddha" [10]. On average, 16 MLP evaluations are required per surface point per view over the training.

A.3. More details on visibility determination

We determine the visibility of a surface point in a view by marching the point toward the corresponding camera, *i.e.*, performing sphere tracing [6] in the reverse direction.

We consider four conditions when marching the surface point. Initially, we push the surface point $\mathbf{x}_0$ by a tiny distance (1×10^{-3} in our experiments) to the camera. (1) The surface point is invisible if the signed distance becomes negative, as the marching direction is towards inside the surface. As long as the marching point is outside the surface, we move the point $\mathbf{x}_t$ at step t by a distance $f(\mathbf{x}_t)$ towards the camera. The surface point is (2) visible if the marching point goes beyond the camera center (Fig. 13 left) or (3) invisible if the marching point hits another surface point (Fig. 13 right). (4) We treat the surface point as invisible if the marching is not terminated within certain steps.

This strategy is advantageous in both efficiency and accuracy compared to other visibility determination strategies used in neural rendering methods. First, it avoids densely evaluating an MLP on the point-to-camera rays [7, 18]. The marching quickly terminates and only requires a few MLP evaluations, *e.g.*, 16 MLP evaluations on average (Fig. 14). Second, it does not rely on the visibility predicted by an additional trainable MLP [15].

B. Evaluation on DiLiGenT-MV

This section provides more details of our evaluation metrics, additional visual comparisons on DiLiGenT-MV benchmark [10], and investigates the effect of number of input viewpoints.

B.1. More details on evaluation metrics

The definition of our evaluation metrics follow [8,9]. We present their definitions here for completeness.

Chamfer distance Chamfer distance measures the point-set-to-point-set distance by accumulating the point-to-point-set distances. Given two point sets χ_1, and χ_2, the distance from a point to another point set is defined as

$$\begin{aligned} d_{\mathbf{x}_1 \to \chi_2} &= \min_{\mathbf{x}_2 \in \chi_2} \|\mathbf{x}_1 - \mathbf{x}_2\|_2 \quad \text{and} \\ d_{\mathbf{x}_2 \to \chi_1} &= \min_{\mathbf{x}_1 \in \chi_1} \|\mathbf{x}_1 - \mathbf{x}_2\|_2. \end{aligned} \tag{26}$$

The Chamfer distance $d(\chi_1, \chi_2)$ is then

$$d(\chi_1, \chi_2) = \frac{1}{2|\chi_1|} \sum_{\mathbf{x}_1 \in \chi_1} d_{\mathbf{x}_1 \to \chi_2} + \frac{1}{2|\chi_2|} \sum_{\mathbf{x}_2 \in \chi_2} d_{\mathbf{x}_2 \to \chi_1}. \tag{27}$$

F-score F-score considers both the precision and recall of the recovered surfaces to the GT surfaces. The precision and recall are defined based on the point-to-point-set distances as

$$\begin{aligned} \mathcal{P} &= \frac{1}{|\chi_1|} \sum_{\mathbf{x}_1 \in \chi_1} [d_{\mathbf{x}_1 \to \chi_2} < \tau] \quad \text{and} \\ \mathcal{R} &= \frac{1}{|\chi_2|} \sum_{\mathbf{x}_2 \in \chi_2} [d_{\mathbf{x}_2 \to \chi_1} < \tau]. \end{aligned} \tag{28}$$

Here, $[\cdot]$ is the Iverson bracket, and τ is the distance threshold for a point to be considered close enough to a point set. The F-score then takes the geometric average of precision and recall:

$$\mathcal{F} = \frac{2\mathcal{P}\mathcal{R}}{\mathcal{P} + \mathcal{R}}. \tag{29}$$

We set $\tau = 0.5\,\mathrm{mm}$ in our evaluations.

As mentioned in the main paper, our evaluation takes the first ray-surface intersection points from all views as the input point sets to the Chamfer distance and F-score. This puts more focus on evaluating visible surface regions in input images and avoids a heuristic crop of the surface [8].

Our evaluation metrics do not consider the *cleanness* of inner space (*i.e.*, correctness of inner topology) of the recovered surfaces. To assess how accurate the inner space of the surfaces is, we visualize the inner space of the mesh in Fig. 15. The visualization shows that our method does not produce unwilling structures inside recovered meshes.

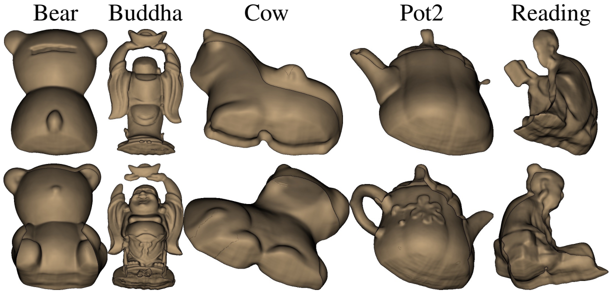

Figure 15. Visualization of the inner space of our recovered surfaces (cut in half vertically). We consider that our evaluation of shape accuracy using visible surface points is fair because the inner space is clean. No post-processing is performed on the meshes after we extract them using marching cubes [11].

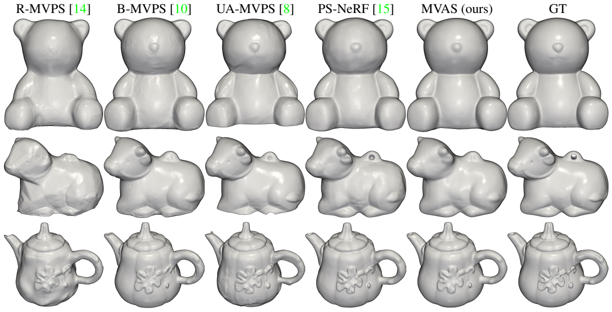

Figure 16. More visual comparisons of recovered shapes of DiLiGenT-MV [10] objects “Bear,”“Cow,” and “Pot2.”

B.2. Additional visual comparisons

Figures 16 and 17 show the visual comparisons on DiLiGenT-MV objects [10] in addition to the ones presented in the main paper's Figs. (7) and (8). Our method consistently recovers accurate and detailed shapes and normals.

Figure 18 shows the comparison of surface normals to PS-NeRF [15] from the 5 unseen viewpoints during the training. PS-NeRF [15] use the 15-view SDPS normal maps [3] to initialize shapes, therefore sharing the same access to underlying azimuth information as ours. The comparison verifies that accurate shape and normal recovery can be realized using only azimuth maps without developing the rendering process for the multi-view case.

B.3. The effect of number of viewpoints

MVAS is robust to sparse view input. As shown in Tab. 3 and Fig. 19, we evaluate the shape and normal recovery accuracy by gradually reducing the number of input views. Figure 19 shows that using as few as 5-view azimuth maps can still achieve detailed reconstruction, while large errors are observed mainly at heavily occluded regions.

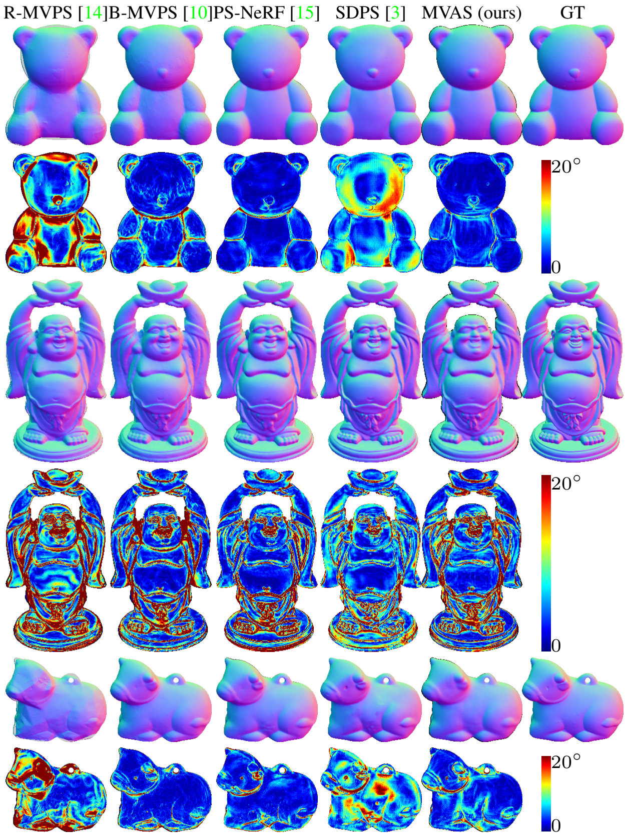

Figure 17. More visual comparisons of recovered normal maps and angular error maps from the first view of DiLiGenT-MV [10] objects "Bear," "Buddha," and "Cow."

Table 3. Effect of the number of views used for shape and normal recovery. MAE is averaged over {5, 10, 12, 14, 15} unseen views, respectively.

Metrics	15	10	8	6	5
CD (↓)	0.357	0.372	0.449	0.424	0.422
F-score (↑)	0.754	0.739	0.648	0.702	0.715
MAE (↓)	9.90	10.80	12.23	13.35	14.25

C. Implementation details

This section describes the architecture of our neural SDF, the training details, and the camera normalization process.

C.1. Neural network architecture

Following IDR [17], our neural SDF consists of a positional encoding layer [12] followed by an 8-layer MLP, as shown in Fig. 20. The positional encoding layer is defined as

$$\gamma(\mathbf{x}) = [\sin(2^0\pi\mathbf{x}), \cos(2^0\pi\mathbf{x}), ..., \sin(2^{L-1}\pi\mathbf{x}), \cos(2^{L-1}\pi\mathbf{x})]. \tag{30}$$

We use $L = 10$ in our experiments. The input position $\mathbf{x}$ and $\gamma(\mathbf{x})$ are skip-connected to the 4-th layer of the MLP. For the activation functions in the MLP, we use the softplus function

$$\text{softplus}(x) = \frac{1}{\beta}\log\left(1 + \exp(\beta x)\right) \tag{31}$$

with $\beta = 100$.

The neural SDF shown in Fig. 20 is the only MLP we optimize. Unlike recent works using additional rendering networks to model surface light field [16,17] or reflectance [15] for computing re-rendering loss, multi-view azimuth maps directly regularize the geometry and eliminate the necessity to model a rendering process.

C.2. Training details

We initialize the MLP parameters such that the initial zero level set approximates a sphere with a radius 0.6 [2]. We set $\lambda_1 = 100$ and $\lambda_2 = 0.1$ for the loss function. ADAM optimizer is used with an initial learning rate 1×10^{-4}. We optimize the MLP parameters for 50 epochs with a batchsize 4096 pixels. The learning rate and α in silhouette loss are divided by 2 every 10 epochs.

As most pixels from the input images are outside silhouette, randomly sampling from all pixels can be inefficient for training. To improve the efficiency, we dilate the silhouette (*i.e.*, the boundary of the mask) for 30 times and sample pixels from the expanded regions as input. For DiLiGenT-MV [10] objects, we use their provided masks. For PANDORA [5] and our captured images, we use an automatic image background removal tool [1] to generate the masks. The input image dimensions are 612 × 512 for DiLiGenT-MV [10], 1224 × 1024 for PANDORA [5], and 1566 × 1045 for our objects.

The training took about 3 hours per DiLiGenT-MV object [10], about 7 hours per PANDORA object [5], and about 10 hours for our captured objects using one GTX 2080Ti graphics card. As a comparison, PS-NeRF took about 22 hours to train one DiLiGenT-MV object [15]. It took us about 30 hours to reproduce PANDORA results per object [5].

C.3. Camera normalization

Following VolSDF [16], we normalize the world coordinates such that the object is bounded by a unit sphere. As we cannot know the shape and its center position beforehand, we approximate the object center location by the position that is closet to all camera principle axes. This approximation assumes all cameras surrounding the target scene and is satisfied in our experiments. We present the computation details here because we do not find such details in the VolSDF paper [16]. The normalization is done by shifting

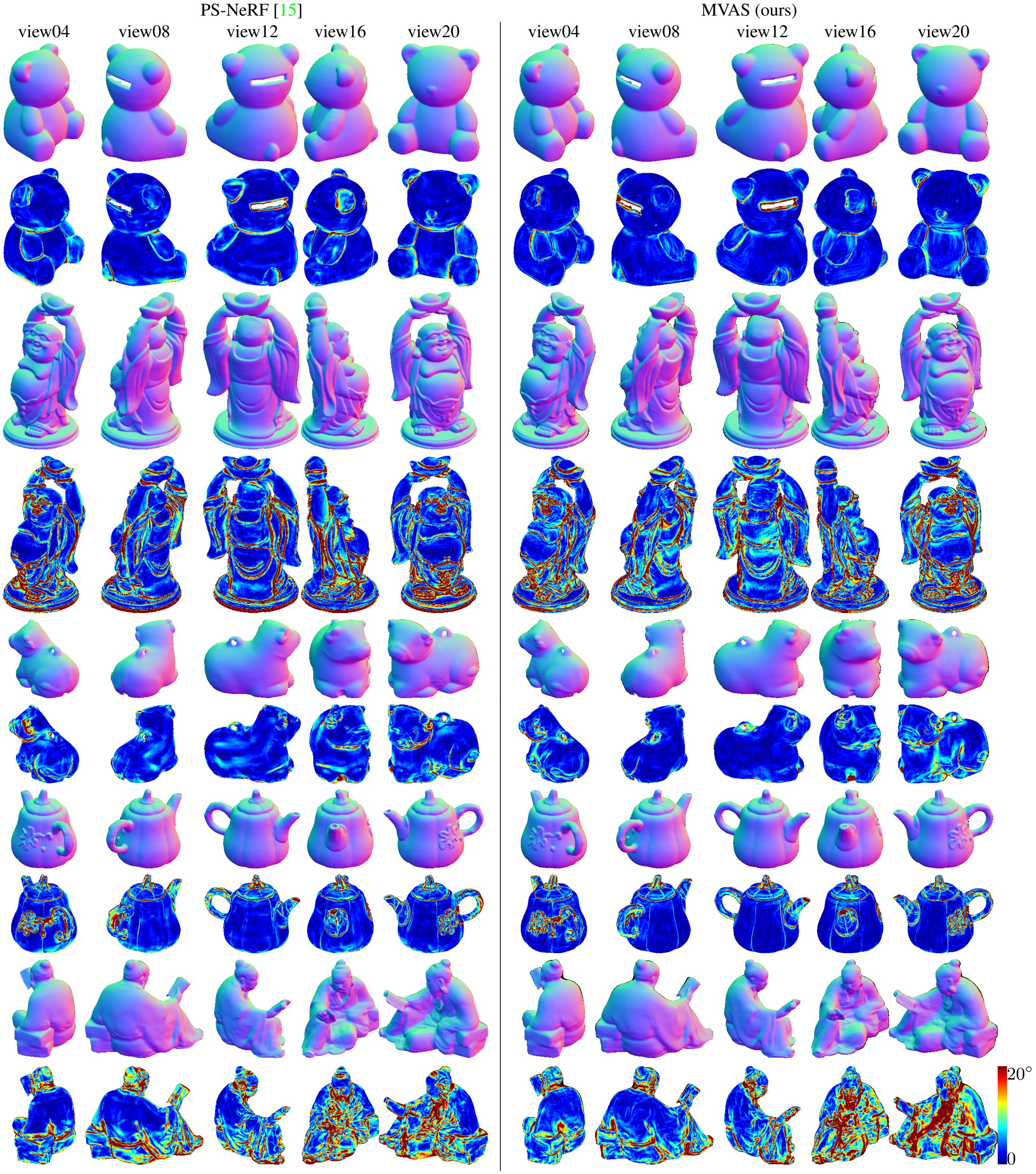

Figure 18. Visual comparisons to PS-NeRF [15] of the 5 unseen views over the training.

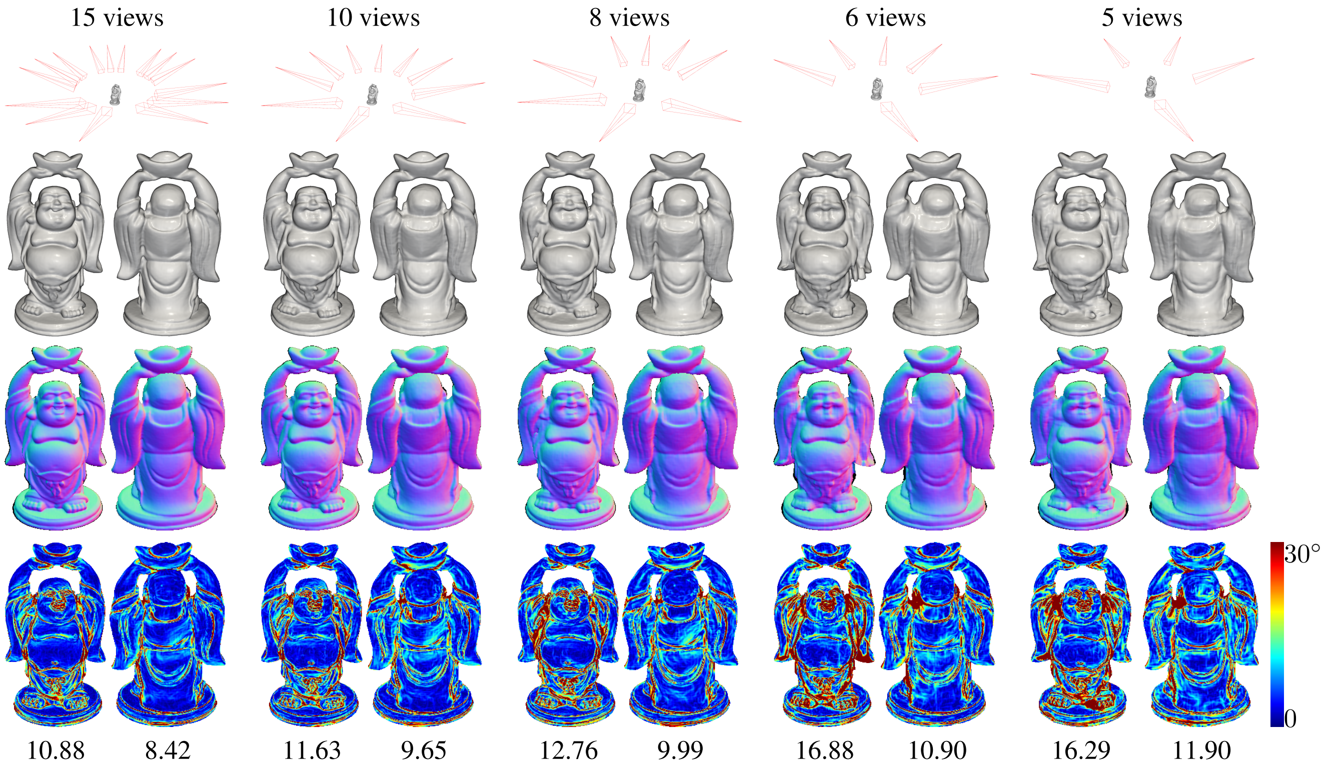

Figure 19. Surface and normal recovery results using different number of viewpoints. From top to bottom: input viewpoints, front and back views of recovered shapes, front and back normal maps, front and back angular error maps, and MAEs in corresponding views. It can be seen that MVAS is robust to sparse view inputs. Most surface details are still distinguishable using as few as 5-view azimuth maps.

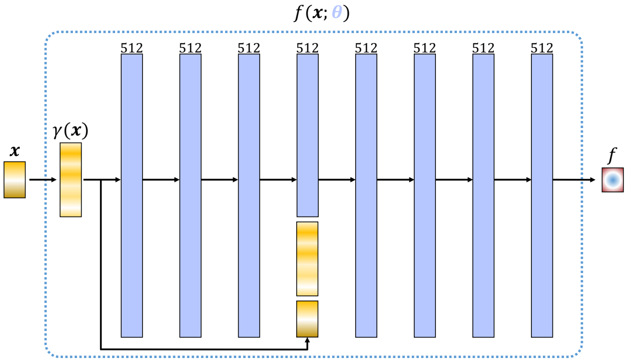

Figure 20. Our network consists of a positional encoding layer $\gamma(\cdot)$ and an 8-layer MLP with softplus activation functions. A skip connection is added to the 4-th layer from the input. This is the only network we optimize.

and then scaling the camera center locations:

$$\mathbf{o}_i \leftarrow \frac{\mathbf{o}_i - \mathbf{x}_o}{s}. \tag{32}$$

Here, $\mathbf{o}_i$ is the i-th camera's center location in the world coordinates, $\mathbf{x}_o$ and s are the global offset and scale factor to be detailed in the following.

Camera centers' offset The offset applied to all camera center locations can be computed using a linear system. Formally, let $\mathbf{o}_i \in \mathbb{R}^3$ and $\mathbf{z}_i \in \mathcal{S}^2 \subset \mathbb{R}^3$ be the i-th camera's center location and its principle axis direction in the world coordinates, respectively. The principle axis can then be represented as $\mathbf{x}_i(t) = \mathbf{o}_i + t\mathbf{z}_i$ with $t \in \mathbb{R}_+$. The shortest squared Euclidean distance from a point $\mathbf{x} \in \mathbb{R}^3$ to this principle axis is

$$\begin{aligned} d^2\left(\mathbf{x}, \mathbf{x}_i(t)\right) &= \min_t \|\mathbf{x} - \mathbf{x}_i(t)\|_2^2 \\ &= (\mathbf{x} - \mathbf{o}_i)^\top (\mathbf{x} - \mathbf{o}_i) - \left((\mathbf{x} - \mathbf{o}_i)^\top \mathbf{z}_i\right)^2 \\ &= \mathbf{x}^\top \mathbf{Z}_i \mathbf{x} - 2\mathbf{o}_i^\top \mathbf{Z}_i \mathbf{x} + \mathbf{o}_i^\top \mathbf{Z}_i \mathbf{o}_i, \end{aligned} \tag{33}$$

where $\mathbf{Z}_i = \mathbf{I} - \mathbf{z}_i\mathbf{z}_i^\top$. To approximate the object center, we find the point that is the closest to all camera principle axes:

$$\begin{aligned} \mathbf{x}_o &= \operatorname*{argmin}_{\mathbf{x}} \sum_i d^2\left(\mathbf{x}, \mathbf{x}_i(t)\right) \\ &= \mathbf{x}^\top \left(\sum_{i=1}^{C} \mathbf{Z}_i\right) \mathbf{x} - 2\left(\sum_{i=1}^{C} \mathbf{o}_i^\top \mathbf{Z}_i\right) \mathbf{x} + \sum_{i=1}^{C} \mathbf{o}_i^\top \mathbf{Z}_i \mathbf{o}_i. \end{aligned} \tag{34}$$

The global optimum $\mathbf{x}_o$ is attained by solving the following normal equation of Eq. (34):

$$\begin{aligned} &\mathbf{A}^\top \mathbf{A}\mathbf{x} = \mathbf{A}^\top \mathbf{b} \\ \text{with} \quad &\mathbf{A} = \sum_{i=1}^{C} \mathbf{Z}_i, \quad \mathbf{b} = \sum_{i=1}^{C} \mathbf{Z}_i \mathbf{o}_i \end{aligned} \tag{35}$$

Camera centers' scale After centering the scene, we apply a global scale to all camera center locations to ensure a unit sphere bounds the scene. We assume that all cameras surround the object. Then we can compute the global scale factor as the maximal camera center norm scaled by a suitable value s_r:

$$s = \max\{\|\mathbf{o}_i - \mathbf{x}_o\|_2\}/s_r. \tag{36}$$

We chose s_r such that it is slightly larger than the ratio of the camera-to-object distance to the object size. For DiLiGenT-MV [10] objects, we set $s_r = 10$ as they are captured about $1.5\,\mathrm{m}$ away from about $20\,\mathrm{cm}$ height objects. For PANDORA [5] and our objects, we set $s_r = 3$.

References

[1] Remove BG. https://www.remove.bg. Accessed: 2022-11-10. 4

[2] Matan Atzmon and Yaron Lipman. Sal: Sign agnostic learning of shapes from raw data. In *Proc. of Computer Vision and Pattern Recognition (CVPR)*, pages 2565–2574, 2020. 4

[3] Guanying Chen, Kai Han, Boxin Shi, Yasuyuki Matsushita, and Kwan-Yee K. Wong. SDPS-Net: Self-calibrating deep photometric stereo networks. In *Proc. of Computer Vision and Pattern Recognition (CVPR)*, 2019. 3, 4

[4] Zhaopeng Cui, Jinwei Gu, Boxin Shi, Ping Tan, and Jan Kautz. Polarimetric multi-view stereo. In *Proc. of Computer Vision and Pattern Recognition (CVPR)*, July 2017. 1

[5] Akshat Dave, Yongyi Zhao, and Ashok Veeraraghavan. PANDORA: Polarization-aided neural decomposition of radiance. *Proc. of European Conference on Computer Vision (ECCV)*, 2022. 1, 4, 7

[6] John C Hart. Sphere tracing: A geometric method for the antialiased ray tracing of implicit surfaces. *The Visual Computer*, 12(10):527–545, 1996. 2

[7] Andrew Hou, Michel Sarkis, Ning Bi, Yiying Tong, and Xiaoming Liu. Face relighting with geometrically consistent shadows. In *Proc. of Computer Vision and Pattern Recognition (CVPR)*, pages 4217–4226, 2022. 2

[8] Berk Kaya, Suryansh Kumar, Carlos Oliveira, Vittorio Ferrari, and Luc Van Gool. Uncertainty-aware deep multi-view photometric stereo. In *Proc. of Computer Vision and Pattern Recognition (CVPR)*, pages 12601–12611, 2022. 3

[9] Arno Knapitsch, Jaesik Park, Qian-Yi Zhou, and Vladlen Koltun. Tanks and temples: Benchmarking large-scale scene reconstruction. *ACM Transactions on Graphics (ToG)*, 36(4):1–13, 2017. 3

[10] Min Li, Zhenglong Zhou, Zhe Wu, Boxin Shi, Changyu Diao, and Ping Tan. Multi-view photometric stereo: A robust solution and benchmark dataset for spatially varying isotropic materials. *IEEE Transactions on Image Processing*, 29:4159–4173, 2020. 2, 3, 4, 7

[11] William E Lorensen and Harvey E Cline. Marching cubes: A high resolution 3D surface construction algorithm. *ACM siggraph computer graphics*, 21(4):163–169, 1987. 3

[12] 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. of European Conference on Computer Vision (ECCV)*, 2020. 4

[13] Daisuke Miyazaki, Robby T Tan, Kenji Hara, and Katsushi Ikeuchi. Polarization-based inverse rendering from a single view. In *Proc. of International Conference on Computer Vision (ICCV)*, volume 3, pages 982–982, 2003. 1

[14] Jaesik Park, Sudipta N Sinha, Yasuyuki Matsushita, Yu-Wing Tai, and In So Kweon. Robust multiview photometric stereo using planar mesh parameterization. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 39(8):1591–1604, 2016. 3, 4

[15] Wenqi Yang, Guanying Chen, Chaofeng Chen, Zhenfang Chen, and Kwan-Yee K. Wong. PS-NeRF: Neural inverse rendering for multi-view photometric stereo. In *Proc. of European Conference on Computer Vision (ECCV)*, 2022. 2, 3, 4, 5

[16] Lior Yariv, Jiatao Gu, Yoni Kasten, and Yaron Lipman. Volume rendering of neural implicit surfaces. *Advances in Neural Information Processing Systems (NeurIPS)*, 34:4805–4815, 2021. 4

[17] Lior Yariv, Yoni Kasten, Dror Moran, Meirav Galun, Matan Atzmon, Basri Ronen, and Yaron Lipman. Multiview neural surface reconstruction by disentangling geometry and appearance. *Advances in Neural Information Processing Systems (NeurIPS)*, 33, 2020. 4

[18] Xiuming Zhang, Pratul P Srinivasan, Boyang Deng, Paul Debevec, William T Freeman, and Jonathan T Barron. Nerfactor: Neural factorization of shape and reflectance under an unknown illumination. *ACM Transactions on Graphics (Proc. of ACM SIGGRAPH)*, 40(6):1–18, 2021. 2

[19] Jinyu Zhao, Yusuke Monno, and Masatoshi Okutomi. Polarimetric multi-view inverse rendering. In *Proc. of European Conference on Computer Vision (ECCV)*, pages 85–102. Springer, 2020. 1

[20] Dizhong Zhu and William AP Smith. Depth from a polarisation+ RGB stereo pair. In *Proc. of Computer Vision and Pattern Recognition (CVPR)*, pages 7586–7595, 2019. 1