Supplementary Material PS-Transformer: Learning Sparse Photometric Stereo Network using Self-Attention Mechanism

Satoshi Ikehata
https://satoshi-ikehata.github.io/
National Institute of Informatics
Tokyo, JAPAN

1 Details about CyclesPS+ Training Dataset

Motivation: As described in the main manuscript, the recent deep-learning-based photometric stereo models [3, 4, 5, 6, 12] were trained either on the Blobby shape dataset [3] or the CyclesPS dataset [6]. The Blobby shape dataset contains much bigger number of samples (*i.e.*, 85212), however the material is spatially uniform and the number of different lights in each dataset is small (*i.e.*, 64). On the other hand, CyclesPS dataset only contains 15 different objects but the material is spatially varying and the number of different lights in each dataset is large (*i.e.*, 740). We empirically compared these two training datasets and got some interesting observations. First, the training speed for our model is much faster when the model was trained on the CyclesPS dataset than on the Blobby shape dataset even with the significant difference in the data size. Second, we found that näively taking a training object in an alphabet order from the CyclesPS dataset often made the training instable. By examining these phenomena, we concluded that the training data in the CyclesPS dataset should be used from "Hard" samples to "Easy" samples to make the test error decrease. Concretely, the CyclesPS dataset includes *diffuse*, *specular* and *metallic* subsets and the training succeeded only when it started to train with metallic objects and then finished with diffuse or specular objects[1]. In addition, we also found that the object concavity and surface normal distribution on the object surface affected the difficulty of the training. Concretely, we should start with objects of highly-concave and biased surface normal distribution ("Hard" case) and then finish by concave objects such as a sphere ("Easy" case). We should note that only using "Easy" samples cannot improve the generalization error therefore using both "Hard" and "Easy" samples is important.

With these observations, we decided to create a new training dataset namely *CyclesPS+*, which is aimed to improve the stability in the training. There are two major modifications. First, in addition to the original objects in CyclesPS dataset, we added 10 additional objects to the dataset to increase the variation of the surface normal distribution. Secondary, and more importantly, we rendered the images w/ and w/o inter-reflections (Blender [1] supports the control of the number of bounces of lights in the raytracing). We found that training

---

[1] This result was true for all the models we tested including CNN-PS [6], PS-FCN+ [5] and GPS-Net [12].

the network only with samples with inter-reflections didn't decrease the generalization error properly but it worked when the network was trained on both w/ and w/o inter-reflections in this order (by following the "Hard" to "Easy" rule). This is why we need training samples rendered both w/ and w/o inter-reflections. Thirdly, we merged *diffuse* and *specular* subset since we didn't find any benefit to separate them.

Dataset detailes: *CyclesPS+* photometric stereo dataset contains 25 objects (including 15 objects in *CyclesPS* [6]) as shown in Fig. 1. For each object, we provide 16-bit integer TIFF images with resolution of 256×256 rendered under 740 lighting directions which are uniformly distributed on a unit sphere. The mask image and true surface normal map are also provided. For each object, we created four subsets according to the material type and presence of the inter-reflection (See differences in Fig. 2). The material type is either of *Specular* or *Metallic*. In a similar manner to the original *CyclesPS* dataset [6], we rendered images with the Disney's Principled BRDF [2] using the physics-based *Blender Cycles renderer* [1]. The difference between *Specular* and *Metallic* is the possible range of Prinsipled BRDF's parameters of *metallic* and *roughness*[2]. In the *Specular* subset, *roughness* value ranges between 0 and 1, and *metallic* value is fixed by zero. On the other hand, in the *Metallic* subset, *roughness* value ranges between 0.3 and 0.7, and *metallic* value ranges between 0.5 to 1.0 (we allow a value less than one to accept the metallic-painted surface). As illustrated in Fig. 1, the same BRDF values are assigned to a superpixel of roughly uniform size (1k superpixels per image). The red, green and yellow channels in *base color* are randomly selected between 0 and 1 for both material types. For the reference, we put two examples of CyclesPS+ dataset in this supplementary package.

Training protocol: We here detail the hyperparameters to control the learning process. For the fair comparison, we used the same optimizer (*i.e.*, Adam optimizer [8] with betas=(0.9,0.999) and weight decay=0) and the same learning rate schedule (*i.e.*, initial learning rate is 2.0×10^{-4} and dropped by half every 3 epochs) for training all models appeared in this work. The training sample size in each epoch was $200K$ for regionwise algorithms (*i.e.*, PS-Transformer, GPS-Net and PS-FCN+) and $600K$ for the pixelwise algorithm (*i.e.*, CNN-PS). Exactly speaking, for each epoch, we firstly trained a network by randomly taking half of entire samples (*e.g.*, $100K$ for a regionwise algorithm) from *Specular* and *Metallic* subsets "w/" inter-reflection, and then trained the model with other half-size random samples from subsets "w/o" inter-reflection. This is because the training progresses better when starts with "Hard" samples and ends with "Easy" samples (Metallic is harder than Specular, w/ inter-reflection is harder than w/o inter-reflection). As for the stopping criteria, we simply took weights with lowest training errors within 15 epochs for all the methods (the entire training time was less than 10 hours for every method). The training batch size was fixed by 512 for CNN-PS, 128 for GPS-Net, 256 for PS-FCN+ and 64 for ours. The difference in batch size was due to the GPU memory limit (in other words, due to the difference of the model complexity as represented by the number of model parameters in Table 1 and required intermediate cashes to keep the gradient record).

2 Implementation Details

Here, we describe some important implementation details whose summary is illustrated in Table 1. In our evaluation, we compared one pixelwise algorithm (*i.e.*, CNN-PS [6]) and

---

[2] Unlike *CyclesPS*, other parameters of the BRDF model such as *SepcularTint* and *Sheen* is fixed by zero to narrow the entire parameter space.

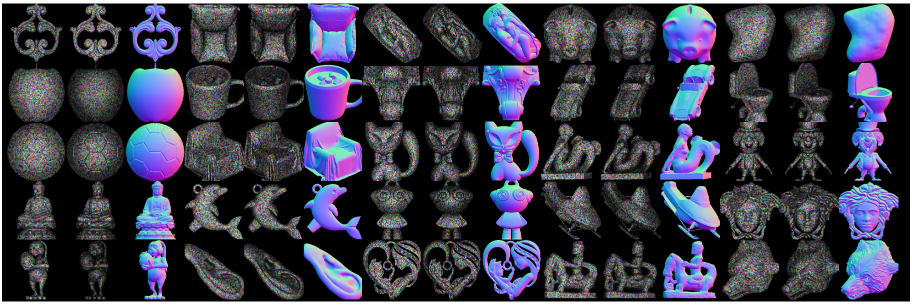


Figure 1: Illustration of Cycles+ dataset. From left to right in each triplet illustrates the mean specular image, mean metallic image and surface normal map for each object. Note that we show example of images without the inter-reflections.

three regionwise algorithms (*i.e.*, PS-Transformer (Ours), GPS-Net [12], PS-FCN+ [5]). CNN-PS takes a single-channel observation map as input, which is generated from randomly selected 10 perpixel observations and corresponding light directions on an upper hemisphere. The size of the observation map is fixed by 32×32 as suggested in [6]. On the other hand, other regionwise algorithms take image patches ($\mathcal{R}^{h \times w \times c}$) and corresponding light maps ($\mathcal{R}^{h \times w \times 3}$) as input. Here, h and w are the height and width of a region and c is the number of color channels (*i.e.*, $c = 1$ for GPS-Net, and $c = 3$ for PS-Transformer and PS-FCN+[3]). h and w are differ in training due to the GPU memory limitation. Concretely, we use $h = w = 32$ for PS-FCN+ and GPS-Net, and $h = w = 8$ for PS-Transformer (Ours). Concretely speaking, we randomly crop $h \times w$ region from the entire image at every training data sampling. As with the difference in training batch size, the difference in region size also comes from the GPU memory limit. Note that for low-level physics-based vision tasks such as PS, the global information is less informative than local information unlike for high-level tasks such as the recognition or detection. In reality, our evaluation indicated that even with a small training region size, PS-Transformer models could learn informative statistics from the local neighbors.

In addition to the input shape, the data normalization algorithm to equalize the value ranges of training/test images also differs among different models. Followed by the authors' implementation, the input observation map for CNN-PS is normalized by its maximum value. For PS-FCN+, GPS-Net[4] and PS-Transformer (Ours), the ℓ_2 normalization is applied to perpixel observations as suggested in [5].

3 Complete Results on Individual Objects

The main manuscript only presented the errors averaged over all objects in DiLiGenT and DiLiGenT-MV datasets. Here, we illustrate the mean angular errors of predicted surface normals (in degrees) *per object*. The results are illustrated in Table 3 and Table 4, Table 5, Ta-

---

[3] We simply followed the original implementation but we observed that the difference in number of channels does not significantly affect the prediction accuracy in our preliminary experiment.

[4] Though it wasn't clearly described in the GPS-Net paper, the authors' official implementation [12] required data normalization to make the intensity ranges of training and test data similar.

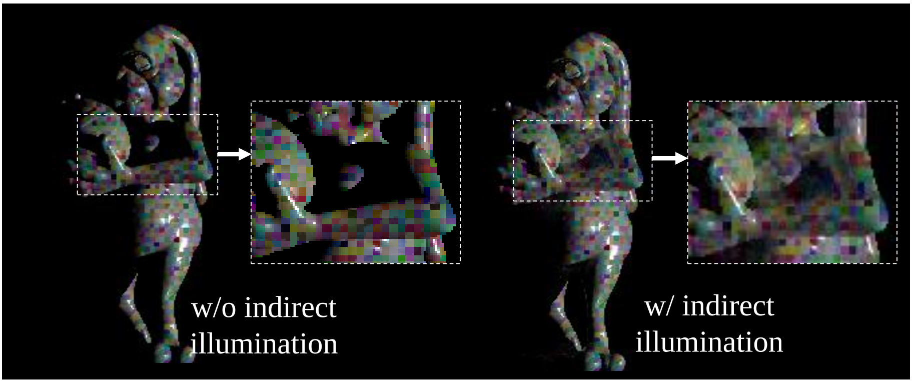


Figure 2: The illustration of the samples in CyclesPS+ dataset with (right) and without (left) the inter-reflections.

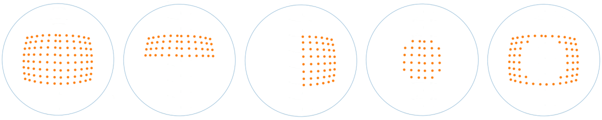
Method	#Parameters	Batch size	Data normalization	Input	Image channels	Available pretrained models
PS-Transformer	22046778	64	L2 normalization	Region	RGB	-
GPS-Net	441322	128	L2 normalization	Region	Grayscale	Trained with 96, **10**, 4 images
PS-FCN+	2210048	256	L2 normalization	Region	RGB	Trained with **32** images
CNN-PS	2950483	512	Max normalizatoin	Obsmap	Grayscale	Trained with 96 images

Table 1: Summary of properties of algorithms in our evaluation. We only used pretrained model for GPS-Net [12] (m=10) and PS-FCN+ [5] (m=32) as colored in red.

ble 6. Note that the data in Table 3 was used to create Fig.3 in the main manuscript and Table 4, Table 5 and Table 6 were used to create Fig.4 in the main manuscript. We observed that for each object our method (PS-Transformer) consistently outperformed other competitors as was observed in the main results.

4 Performance Analysis on the Different Light Source Distribution

As described in the main manuscript, in all the experiments, we extracted ten sets of light distributions where each light direction is randomly chosen from entire 96 light directions (See Fig. 3-(a)) in advance and applied the same light set to all the methods and averaged all results. In this section, we studied the effect in performance when the light source is sampled from only specific directions, instead of from all 96 candidates. The result is shown in Fig. 3. Here we evaluated the performance of the PS-Transformer on DiLiGenT dataset based on the five different sets of candidate directions (all, upper-half, right-half, middle and edge). We observed that the performance obviously degrades when the light direction is sampled only from the upper-half or right-half of the upper-hemisphere. Interestingly, the best performance was observed when the light is sampled from the edge rather than sampled from the entire possible directions even though the difference is slight. This observation is actually reasonable because when a surface is illuminated from the side, diffuse reflections



(a) all (b) upper-half (c) right-half (d) middle (e) edge

Sampling light source distribution

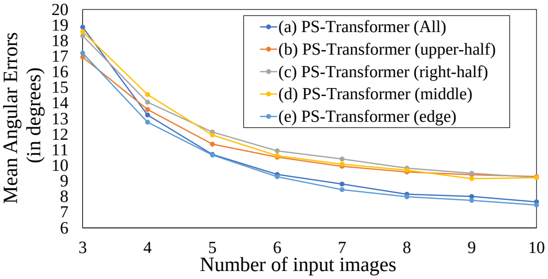


Figure 3: The performance comparison about the different light sampling distributions. When randomly selecting light directions, we varied the distribution of possible sampling directions by five different patterns (a) all, (b) upper-half, (c) right-half, (d) middle and (e) edge.

	sec/pixel (m=10)	algorithm	device	sec/image (256x256)
Lambertian[27]	4.9E-05	Pixel	CPU	3.2
Bivariate [14]	3.9E-04	Pixel	CPU	25.4
CNN-PS[13]	2.0E-04	Pixel	GPU	13.2
PS-FCN+[6]	4.4E-06	Patch	GPU	0.3
GPS-Net[29]	5.3E-06	Patch	GPU	0.3
SPS-Net[20]	1.6E-05	Patch	GPU	1.1
PS-Transformer	1.8E-05	Patch	GPU	1.2

Table 2: Computational time for the inference.

are dominantly observed rather than specular reflections, and it has been shown that the estimation of the normal becomes easier from the diffuse observations [10].

5 Computation Time Analysis

In this section, we studied the computational time at test (note that the training time is mostly based on the number of training epochs and we kept same epoch numbers for all methods for the fair evaluation). We compared the inference time of Lambertian-based method [11], the optimization-based constrained bi-variate regression [7], CNN-PS [6], PS-FCN+ [5], GPS-Net [12], SPS-Net [9] and PS-Transformer for estimating surface normals from 10 input images on the same machine. Note that the first two methods are based on the numerical optimization and others are deep-learning-based algorithms.

The result is illustrated in Table 2. Since some algorithms estimate surface normal pixelwisely and others estimate an entire surface normal map (or patch) directly, we compute the computational time *per pixel* rather than per surface normal map by dividing the inference time by the number of valid pixels in the estimation for the fair comparison (Just for reference, we also include the computational time per image as sec/pixel × 256 × 256 assuming that the image size is 256 × 256 pixels). We observe that the deep-learning-based algorithms generally much more efficient than optimization-based algorithms due to two reasons. First, the parallel computing using GPU is available for the learning-based algorithms. Second, the neural networks do not require the numerical optimization unlike conventional optimization-based algorithms such as [7]. We also observe that CNN-PS is less efficient than other deep-learning-based methods due to the convolutions of *pixelwise* observation maps. PS-Transformer and SPS-Net is less efficient than PS-FCN+ and GPS-Net due to the expensive computation of the self-attention even though the difference is practically not problematic.

6 Visualization of Predicted Surface Normals

Because the number of objects in our evaluation is huge (110 in total) due to the viewpoint variations in the DiLiGenT-MV dataset, we put most of the predicted surface normal maps into directories in this supplementary package (*i.e.*, "DiLiGenT_Results" and "DiLiGenTMV_Results") for qualitatively comparing the results (All surface normal maps were averaged over 10 trials). Here, we describe the configuration of images. In *DiLiGenT_Results* directory, we put the normal maps corresponding to Table 3 in the main manuscript. The title of each image file indicates the number of test images and the object name (*e.g.*, "fig_N=03_ball.jpg" means that the number of different lights in the input is 10 and the object name is "ball"). From left to right in each image, the ground truth surface normal map and the predicted surface normal maps by ours, CNN-PS [6], PS-FCN+ [3], PS-FCN+ using the pretrained model, GPS-Net [12] and GPS-Net using the pretrained model are lined up, respectively. We also attached the error maps (at bottom line) and the object mask (below the ground truth normal map). The intensity of the error map is the angular error of the predicted surface normal which linearly scaled from zero to one (one when the angular error is larger than 90 degrees). In *DiLiGenTMV_Results* directory, we put all the results predicted from the DiLiGenT-MV dataset [5] which are corresponding to Table 4, Table 5 and Table 6. The title of a file contains the number of different lights and view index of the object (*e.g.*, "fig_N=04_V=01_bear.jpg" means that the number of different lights in the input is 4 and the view index is 1.). We put the ground truth surface normal map and predictions by ours, CNN-PS and PS-FCN+ which were trained on the CyclesPS+ dataset.

References

[1] Blender. https://www.cycles-renderer.org/.

[2] B. Burley. Physically-based shading at disney, part of practical physically based shading in film and game production. *SIGGRAPH 2012 Course Notes*, 2012.

[3] G. Chen, K. Han, and K-Y. K. Wong. Ps-fcn: A flexible learning framework for photometric stereo. *Proc. ECCV*, 2018.

GPS-Net (Pretrained)

# Images	3	4	5	6	7	8	9	10
ball	15.03	6.70	8.13	8.26	7.12	5.21	5.94	4.98
bear	19.13	10.31	11.31	10.34	7.85	7.18	8.49	6.77
buddha	22.97	12.96	14.68	13.87	10.68	10.29	10.27	9.26
cat	17.65	11.73	10.81	10.60	9.09	7.67	8.04	7.21
cow	24.22	18.73	15.53	13.35	13.84	13.74	10.37	12.03
goblet	22.86	14.84	15.99	15.35	13.07	12.33	12.69	11.28
harvest	31.18	21.50	23.92	21.28	20.17	18.21	18.54	17.19
pot1	18.17	12.40	12.06	9.85	8.70	8.63	8.13	7.90
pot2	19.90	11.29	13.15	10.90	10.49	8.37	8.94	8.04
reading	28.56	21.47	20.51	16.98	15.92	17.08	15.35	16.17
MAE	21.97	14.19	14.61	13.08	11.69	10.87	10.68	10.08

GPS-Net (Trained on CyclesPS+)

# Images	3	4	5	6	7	8	9	10
ball	19.27	14.85	12.82	10.76	11.22	10.29	9.74	9.03
bear	21.49	17.67	13.06	11.42	10.42	9.95	9.57	9.11
buddha	18.96	17.21	12.53	11.36	11.01	9.99	9.76	9.82
cat	22.82	16.67	13.10	11.25	11.62	10.73	9.99	9.96
cow	24.73	19.59	15.58	14.56	12.90	11.92	12.11	10.61
goblet	24.16	18.48	16.86	14.34	14.10	13.97	12.16	13.42
harvest	43.13	39.28	38.22	37.40	37.12	37.74	36.73	36.80
pot1	29.64	20.90	14.28	14.01	13.05	11.88	11.41	11.25
pot2	21.01	17.31	13.13	11.98	10.67	10.28	9.65	8.60
reading	35.04	35.90	33.60	34.34	33.75	33.89	33.95	32.95
MAE	26.03	21.79	18.32	17.14	16.58	16.06	15.51	15.15

PS-FCN+ (Pretrained)

# Images	3	4	5	6	7	8	9	10
ball	33.25	21.79	19.66	15.81	16.32	12.00	12.81	12.65
bear	22.11	16.96	14.03	10.81	8.30	8.45	7.15	7.10
buddha	23.38	22.10	14.69	13.81	12.43	11.75	11.72	11.27
cat	22.29	10.72	9.50	8.32	7.78	6.81	6.48	6.27
cow	27.37	21.67	18.58	13.32	14.95	12.37	11.54	12.40
goblet	28.32	19.36	16.27	13.41	11.93	11.47	11.76	11.47
harvest	32.19	26.52	25.39	23.13	23.77	21.55	20.52	20.84
pot1	19.39	19.05	10.63	9.05	9.43	8.01	7.90	7.39
pot2	21.22	14.91	17.04	11.60	11.24	10.27	9.42	9.63
reading	34.80	30.30	23.62	24.24	20.74	18.44	17.86	17.49
MAE	26.43	20.34	16.94	14.35	13.69	12.11	11.72	11.65

PS-FCN+ (Trained on CyclesPS+)

# Images	3	4	5	6	7	8	9	10
ball	25.14	13.01	9.86	8.50	7.70	8.13	7.37	7.17
bear	22.22	15.69	11.30	10.29	10.82	8.39	8.13	8.60
buddha	21.48	18.01	16.04	14.28	14.20	13.48	12.87	12.58
cat	16.18	14.17	10.50	10.77	10.10	8.23	8.53	8.15
cow	26.54	21.03	17.52	17.87	16.81	16.18	14.22	13.20
goblet	23.98	18.66	15.85	15.59	13.71	13.22	12.91	12.53
harvest	26.71	24.36	22.11	21.31	19.90	19.11	18.27	17.59
pot1	19.65	13.86	11.26	11.46	10.08	9.95	9.58	9.46
pot2	24.60	16.64	13.87	12.54	12.64	12.64	12.10	10.88
reading	26.04	20.08	17.45	16.71	15.78	15.14	14.99	15.05
MAE	23.25	17.55	14.58	13.93	13.17	12.45	11.90	11.52

SPS-Net (Trained with 32×32 patches on CyclesPS+)

# Images	3	4	5	6	7	8	9	10
ball	13.50	9.46	6.30	6.35	4.86	4.98	4.69	4.64
bear	16.22	10.23	9.30	9.40	8.40	7.68	7.17	6.84
buddha	19.23	14.55	13.43	12.52	12.26	11.96	11.46	11.11
cat	15.03	10.49	10.15	8.87	8.49	8.03	7.83	7.74
cow	19.87	16.12	14.95	12.74	11.76	11.33	10.42	10.57
goblet	18.39	14.61	11.99	10.72	10.60	9.56	9.77	8.86
harvest	23.71	22.37	19.57	19.21	18.61	18.08	17.25	16.61
pot1	17.13	12.87	11.39	10.56	9.85	9.56	9.19	9.07
pot2	16.13	14.74	12.23	11.33	10.70	10.05	9.90	9.06
reading	19.92	16.73	14.78	13.48	13.78	13.41	13.15	12.66
MAE	17.91	14.22	12.41	11.52	10.93	10.46	10.08	9.72

SPS-Net (Trained with 8×8 patches on CyclesPS+)

# Images	3	4	5	6	7	8	9	10
ball	15.58	10.95	7.38	6.12	4.91	4.50	4.32	4.32
bear	14.33	9.17	7.81	6.01	5.82	5.66	5.87	5.48
buddha	18.32	15.54	13.54	12.82	12.01	11.52	10.94	11.15
cat	13.66	12.15	9.41	7.78	7.87	7.54	7.63	7.48
cow	16.75	13.37	10.50	10.03	8.95	8.10	8.32	7.91
goblet	18.18	13.33	11.61	11.14	10.10	9.80	9.54	9.10
harvest	22.59	20.30	18.70	17.60	17.12	16.52	16.13	15.73
pot1	15.73	11.88	10.30	9.93	9.92	9.35	9.35	9.20
pot2	15.33	13.02	9.54	9.30	9.14	9.13	8.70	8.54
reading	18.98	16.16	14.85	14.58	13.63	13.69	13.27	13.96
MAE	**16.94**	13.59	11.36	10.53	9.95	9.58	9.41	9.29

CNN-PS (Trained on CyclesPS+)

# Images	3	4	5	6	7	8	9	10
ball	13.99	8.72	6.69	5.61	4.97	4.51	4.22	4.06
bear	17.05	11.29	8.91	7.62	6.90	6.40	6.06	5.79
buddha	20.65	15.59	13.24	11.99	11.19	10.64	10.26	9.97
cat	15.31	10.55	8.71	7.84	7.32	6.94	6.72	6.54
cow	28.74	21.83	17.45	14.50	12.69	11.38	10.49	9.88
goblet	24.25	18.35	15.18	13.32	12.12	11.34	10.82	10.45
harvest	32.58	26.49	22.96	20.83	19.44	18.53	17.89	17.44
pot1	15.85	11.05	9.24	8.35	7.82	7.43	7.16	6.94
pot2	21.52	16.00	12.93	11.13	9.92	9.16	8.61	8.23
reading	25.14	19.61	17.02	15.67	14.88	14.30	13.91	13.63
MAE	21.51	15.95	13.23	11.69	10.72	10.06	9.61	9.29

PS-Transformer (Trained on CyclesPS+)

# Images	3	4	5	6	7	8	9	10
ball	12.92	7.99	5.95	4.93	4.04	3.64	3.80	3.27
bear	15.61	8.93	6.62	5.81	5.68	5.26	5.20	4.88
buddha	17.61	13.76	11.09	9.82	9.79	8.93	8.91	8.65
cat	15.49	8.47	7.05	6.52	5.69	5.46	5.41	5.34
cow	21.22	14.95	10.86	8.64	7.75	7.09	6.81	6.54
goblet	22.14	14.28	12.87	10.48	10.79	9.03	9.03	9.28
harvest	28.39	22.76	20.14	17.80	16.29	15.73	15.30	14.41
pot1	15.71	10.05	7.75	7.13	6.56	6.41	6.07	6.06
pot2	18.65	13.17	11.12	9.79	8.67	8.16	7.99	6.97
reading	20.93	17.98	13.66	13.31	12.83	11.84	11.61	11.24
MAE	18.87	**13.23**	**10.71**	**9.42**	**8.81**	**8.15**	**8.01**	**7.66**

Table 3: The prediction errors on DiLiGenT dataset. The values are mean angular errors per object (in degrees).

[4] G. Chen, K. Han, B. Shi, Y. Matsushita, and K. K. K. Wong. Self-calibrating deep photometric stereo networks. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8731–8739, 2019.

[5] Guanying Chen, Kai Han, Boxin Shi, Yasuyuki Matsushita, and Kwan-Yee K. Wong. Deep photometric stereo for non-Lambertian surfaces. *TPAMI*, 2020.

[6] S. Ikehata. Cnn-ps: Cnn-based photometric stereo for general non-convex surfaces. In *Proc. ECCV*, 2018.

[7] S. Ikehata, D. Wipf, Y. Matsushita, and K. Aizawa. Photometric stereo using sparse bayesian

m = 10	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	AVE
bear	11.07	5.80	5.69	5.97	6.55	6.13	6.12	5.50	6.61	5.07	4.94	5.06	4.75	5.20	13.31	6.69	7.15	6.49	6.16	5.51	6.49
buddha	8.66	11.19	12.49	12.38	12.18	11.50	11.92	11.37	9.68	8.86	8.29	8.17	9.33	10.05	10.74	10.95	11.18	11.11	10.11	10.68	10.54
cow	6.59	7.44	7.97	7.75	7.16	7.40	6.80	6.44	6.05	6.22	5.96	7.25	8.68	7.33	8.06	8.98	9.11	8.43	7.97	7.39	7.45
pot2	7.51	7.66	9.89	8.57	9.62	9.18	7.93	7.45	7.05	6.50	6.90	7.54	8.05	8.20	7.50	8.47	8.29	8.64	7.85	6.82	7.98
reading	11.24	11.42	11.02	10.52	10.03	11.20	10.04	10.24	9.24	9.48	8.61	10.75	11.91	14.30	14.81	14.26	13.65	12.82	11.87	12.22	11.48
m = 8	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	AVE
bear	12.08	5.61	5.64	6.66	6.67	6.28	6.42	6.22	7.66	5.97	5.77	5.71	5.53	6.01	13.18	6.91	7.00	6.49	5.85	5.61	6.86
buddha	9.13	11.68	12.63	12.94	12.60	11.93	12.36	11.24	10.08	9.13	8.49	9.12	9.74	10.45	11.61	10.76	11.28	11.24	10.18	10.60	10.86
cow	7.33	7.83	8.36	8.10	7.61	7.75	6.78	6.25	6.83	7.23	5.93	8.24	9.60	8.21	8.86	10.34	9.89	9.59	9.09	8.32	8.11
pot2	7.84	7.85	11.57	8.44	9.92	10.15	8.20	8.42	7.77	7.88	7.67	7.69	8.32	8.55	8.31	9.24	8.25	8.12	8.31	7.18	8.48
reading	11.32	11.88	11.94	10.58	10.43	10.81	9.73	10.07	9.92	9.67	9.16	10.74	12.79	14.89	14.01	14.28	13.78	13.50	12.55	12.33	11.72
m = 6	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	AVE
bear	10.35	6.69	6.87	9.79	8.71	8.39	7.24	7.20	9.18	6.82	6.67	6.99	6.51	6.49	13.79	8.78	8.37	7.63	8.28	6.94	8.08
buddha	10.49	12.36	13.61	13.99	13.95	13.84	13.45	12.65	11.07	9.83	9.26	9.84	11.54	12.56	13.98	11.86	12.49	12.06	11.04	11.57	12.07
cow	10.05	9.26	9.65	9.64	10.68	9.46	8.67	8.43	8.17	8.33	8.36	9.08	10.18	10.08	10.68	11.53	11.05	11.46	9.65	9.39	9.69
pot2	8.27	10.72	12.56	10.94	13.89	12.07	10.60	9.58	9.61	8.35	8.41	9.36	10.30	9.82	11.37	10.74	9.49	9.79	9.69	8.29	10.19
reading	13.85	14.02	13.05	11.52	11.95	12.15	10.97	11.75	10.31	11.00	10.42	12.07	13.88	16.68	16.13	16.18	15.05	14.53	13.26	14.05	13.14
m = 4	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	AVE
bear	15.07	9.06	11.43	9.33	10.70	11.71	8.90	8.81	10.12	8.39	9.21	8.10	8.39	9.04	17.95	11.92	12.62	11.43	11.20	8.74	10.61
buddha	12.40	15.62	16.34	17.30	17.74	17.82	17.89	16.54	13.62	12.66	11.94	12.18	13.39	17.20	17.55	16.15	15.47	14.18	13.44	14.95	15.22
cow	16.94	13.74	13.36	15.41	14.52	14.17	15.59	12.95	14.54	14.67	14.48	14.90	14.01	12.61	14.57	15.61	17.13	15.31	16.39	15.41	14.82
pot2	13.32	12.32	14.87	14.48	17.43	18.13	14.90	12.51	11.69	11.71	12.08	12.46	12.98	14.13	13.23	12.91	14.53	13.69	14.42	12.88	13.73
reading	16.73	17.34	17.82	12.82	15.55	15.03	14.29	13.82	13.78	13.37	12.00	14.83	17.12	21.25	22.19	18.69	18.41	18.66	16.76	16.55	16.35

Table 4: The prediction errors by PS-Transformer (Trained on CyclesPS+) on the DiLiGenT-MV dataset. The values are mean angular errors per object, per view (in degrees). m indicates the number of different lights and the counts (1 to 20) indicate the view index.

m = 10	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	AVE
bear	10.29	5.86	6.01	6.21	6.51	6.27	6.05	5.96	7.03	5.86	5.78	5.76	5.86	6.08	11.41	6.84	6.90	6.85	6.41	5.94	6.69
buddha	10.26	12.40	13.44	13.58	13.38	12.82	13.07	12.54	10.88	9.94	9.32	9.46	10.42	11.61	12.87	12.29	12.78	12.27	11.37	11.76	11.82
cow	9.85	10.03	10.75	10.51	9.56	9.55	9.51	9.80	9.97	9.46	8.71	9.51	11.51	10.84	10.73	11.21	11.32	11.29	11.22	10.83	10.31
pot2	8.21	8.77	11.39	9.90	11.03	11.90	9.47	8.92	8.40	7.52	7.66	8.30	9.21	9.17	9.03	9.36	8.78	9.07	8.83	8.08	9.15
reading	13.65	13.51	13.18	11.05	11.09	11.40	10.85	10.77	10.32	10.49	9.83	11.52	13.83	15.93	15.68	16.38	16.12	15.70	14.51	14.25	13.00
m = 8	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	AVE
bear	10.82	6.45	6.64	6.81	7.13	6.92	6.64	6.47	7.55	6.38	6.31	6.25	6.36	6.61	11.90	7.46	7.56	7.50	6.98	6.46	7.26
buddha	10.94	12.98	13.97	14.17	14.07	13.71	13.98	13.29	11.49	10.54	9.93	10.16	11.17	12.48	13.94	13.19	13.41	12.84	11.90	12.44	12.53
cow	11.40	11.44	11.59	11.35	10.72	10.78	10.66	10.90	11.34	11.06	10.25	10.99	12.32	11.71	12.06	12.69	12.68	12.51	12.43	12.37	11.56
pot2	9.16	9.65	12.28	10.86	12.09	13.02	10.49	9.93	9.28	8.38	8.57	9.27	10.19	10.06	9.98	10.33	9.75	10.13	9.81	9.06	10.11
reading	14.31	14.80	13.92	11.49	11.55	11.74	11.50	11.31	10.88	11.06	10.47	12.21	14.53	16.61	16.50	16.94	16.60	16.26	15.00	14.78	13.62
m = 6	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	AVE
bear	12.22	7.69	7.97	8.15	8.51	8.35	7.94	7.66	8.81	7.53	7.51	7.35	7.38	7.73	13.30	8.73	8.95	8.87	8.27	7.63	8.53
buddha	12.30	14.19	15.11	15.39	15.58	15.50	15.83	15.00	12.93	11.85	11.14	11.50	12.66	14.23	15.94	15.07	14.97	14.03	13.02	13.83	14.00
cow	14.57	14.36	13.64	13.39	13.03	13.32	13.16	13.39	14.31	14.10	13.61	14.23	14.28	13.70	14.60	15.49	15.43	14.97	15.05	15.58	14.21
pot2	11.14	11.40	14.02	12.85	14.13	14.95	12.59	12.06	11.09	10.39	10.61	11.27	12.18	11.87	11.75	12.22	11.81	12.39	11.86	11.12	12.08
reading	15.63	15.62	15.49	12.59	12.62	13.03	12.80	12.47	12.03	12.22	11.65	13.41	15.93	18.14	18.20	18.21	17.77	17.42	16.20	15.94	14.87
m = 4	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	AVE
bear	16.18	11.38	11.86	11.99	12.23	12.16	11.75	11.27	12.45	10.97	11.05	10.67	10.70	11.17	17.08	12.29	12.73	12.74	11.97	11.22	12.19
buddha	16.06	17.58	18.43	19.03	19.68	19.99	20.25	19.42	16.99	15.49	14.54	15.10	16.58	18.62	20.56	19.42	19.23	17.58	16.27	17.47	17.91
cow	21.87	21.03	19.23	18.49	18.41	18.94	18.88	19.30	21.08	21.30	21.33	21.36	19.64	18.71	20.28	21.79	21.70	20.59	21.04	22.52	20.37
pot2	15.95	15.74	18.33	17.79	18.83	19.61	17.94	17.27	16.07	15.59	15.96	16.45	17.29	16.52	16.11	16.66	16.82	17.76	17.03	16.17	16.99
reading	19.56	20.40	19.90	16.45	16.30	17.07	16.56	16.16	15.73	16.15	15.76	17.35	19.82	22.12	22.36	22.58	21.82	21.34	19.93	19.56	18.85

Table 5: The prediction errors by CNN-PS (Trained on CyclesPS+) on the DiLiGenT-MV dataset. The values are mean angular errors per object, per view (in degrees). m indicates the number of different lights and the counts (1 to 20) indicate the view index.

regression for general diffuse surfaces. *IEEE Trans. Pattern Anal. Mach. Intell.*, 36(9):1816–1831, 2014.

[8] D. Kingma and J. Ba. Adam: A method for stochastic optimization. *In proc. ICLR*, 2014.

[9] Huiyu Liu, Yunhui Yan, Kechen Song, and Han Yu. Sps-net: Self-attention photometric stereo network. *IEEE Transactions on Instrumentation and Measurement*, 70:1–13, 2021.

$m = 10$	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	AVE
bear	8.78	8.41	8.32	7.88	8.00	8.63	8.54	8.07	8.03	8.87	9.35	8.06	7.55	8.38	11.62	8.67	8.42	8.51	8.64	8.44	8.56
buddha	12.80	13.94	14.15	14.79	14.95	14.72	14.67	13.93	12.27	11.71	11.12	10.70	12.00	13.40	14.63	14.89	13.62	13.50	13.47	13.63	13.44
cow	13.12	12.80	10.74	9.96	10.93	11.35	11.03	10.57	11.89	13.14	12.40	12.40	10.76	11.90	11.66	13.09	12.81	11.57	12.96	13.37	11.92
pot2	11.05	11.41	13.02	12.09	13.74	14.52	12.92	11.86	10.58	10.76	10.54	10.62	11.95	12.63	11.77	12.11	12.89	12.89	12.10	10.67	12.01
reading	14.71	14.14	13.35	11.68	12.28	11.98	11.33	11.32	9.98	10.88	10.16	12.01	14.61	16.10	15.88	15.67	16.67	16.27	16.04	15.84	13.55
$m= 8$	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	AVE
bear	9.99	9.75	9.43	8.71	9.08	9.14	9.14	8.89	8.80	8.98	9.11	7.73	7.46	8.74	12.71	9.06	9.00	9.34	9.06	9.71	9.19
buddha	13.64	14.55	15.14	15.39	16.18	15.39	15.79	14.35	13.50	12.72	11.91	11.86	13.12	14.40	16.19	16.26	15.34	14.66	14.28	14.11	14.44
cow	15.20	14.89	12.77	10.85	11.68	12.58	12.24	12.36	13.61	13.88	13.96	12.90	12.64	12.06	13.54	13.79	14.30	13.85	14.01	17.10	13.41
pot2	11.52	11.92	14.14	13.61	14.44	15.68	13.97	13.79	11.10	11.72	11.11	11.93	12.93	12.78	12.16	12.66	13.47	13.66	13.53	11.78	12.90
reading	15.75	14.70	14.37	13.32	12.78	12.47	13.15	12.06	11.37	12.17	10.90	12.65	15.16	17.62	16.54	16.87	18.56	17.16	16.09	15.90	14.48
$m = 6$	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	AVE
bear	11.11	10.77	10.52	10.99	12.18	12.69	11.73	10.60	11.76	10.30	11.59	10.02	9.62	10.30	21.97	11.94	10.91	12.52	10.80	10.60	11.64
buddha	14.36	15.93	15.75	16.91	17.43	16.89	17.84	16.80	15.10	14.89	14.85	13.25	14.43	17.31	18.33	16.95	16.30	15.60	15.22	15.65	15.99
cow	15.33	15.09	14.00	12.36	13.11	13.56	14.08	14.21	14.36	17.31	15.08	15.55	13.82	14.05	14.35	16.22	15.67	14.45	16.23	17.05	14.79
pot2	14.05	12.34	15.38	14.78	16.15	16.89	15.61	14.63	13.33	12.85	12.33	13.90	14.58	13.67	13.24	14.51	14.20	14.98	14.97	12.78	14.26
reading	16.41	16.80	15.88	14.48	13.77	15.38	14.02	13.34	13.86	13.19	13.06	13.59	15.98	18.23	18.26	18.81	18.28	18.52	17.60	17.06	15.82
$m = 4$	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	AVE
bear	14.23	13.52	16.87	15.24	13.32	16.89	15.22	16.61	16.23	15.00	15.16	15.59	13.15	13.43	20.22	13.49	14.98	16.08	13.38	15.22	15.19
buddha	18.60	17.27	20.14	21.25	20.63	20.42	20.49	22.76	18.00	16.61	17.86	19.56	19.37	22.63	22.88	21.17	18.71	18.31	17.82	19.18	19.68
cow	20.72	17.85	17.05	17.08	16.69	19.30	16.63	17.09	18.88	21.07	20.11	20.86	18.23	17.09	18.91	18.98	18.62	17.97	18.35	20.28	18.59
pot2	16.69	16.30	21.16	20.28	19.56	18.49	20.26	19.51	16.04	17.32	16.36	17.33	18.49	17.88	18.00	18.83	17.71	17.42	16.92	19.53	18.20
reading	22.39	20.94	21.01	19.02	16.60	17.42	17.91	20.00	16.32	18.92	16.18	19.18	20.86	21.48	22.98	21.62	21.89	23.95	21.58	20.75	20.05

Table 6: The prediction errors by PSFCN+ (Trained on CyclesPS+) on the DiLiGenT-MV dataset. The values are mean angular errors per object, per view (in degrees). m indicates the number of different lights and the numbers (1 to 20) indicate the view index.

$m = 10$	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	AVE
bear	8.23	6.59	6.91	6.55	7.48	7.15	6.39	6.58	6.68	6.73	6.51	6.98	6.42	6.68	9.02	6.88	7.29	7.88	7.27	7.30	7.08
buddha	11.89	13.11	13.80	13.78	13.59	13.64	13.75	12.99	11.45	10.40	9.99	10.51	11.16	11.95	13.34	13.36	13.36	12.69	12.10	12.57	12.47
cow	10.29	10.03	8.76	8.83	9.19	9.38	9.53	8.64	8.98	9.86	9.96	9.86	8.28	8.40	10.06	11.36	10.60	10.20	10.67	11.10	9.70
pot2	9.08	9.89	11.98	11.01	12.34	12.68	10.75	9.85	8.94	8.53	8.69	9.28	10.14	9.91	9.76	10.05	10.33	10.13	9.80	9.24	10.12
reading	12.98	12.69	12.81	11.38	11.26	11.18	10.66	10.99	10.11	10.02	9.98	11.29	13.35	15.61	15.45	15.58	14.94	15.04	14.22	13.53	12.65
$m = 8$	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	AVE
bear	8.50	7.32	7.08	7.48	7.41	7.62	7.48	7.76	7.41	6.98	7.59	7.22	7.12	7.10	8.67	7.43	7.91	8.37	7.63	7.72	7.59
buddha	11.71	13.84	14.65	14.16	14.45	14.26	14.81	13.56	12.37	10.99	10.18	10.60	11.72	12.50	14.68	14.24	13.76	13.40	12.64	13.16	13.08
cow	11.26	10.70	9.67	9.10	10.02	10.40	10.84	9.54	10.40	11.34	10.38	10.82	9.28	9.09	10.72	12.99	11.78	11.17	10.69	12.18	10.62
pot2	9.97	9.80	12.53	11.45	12.86	13.77	12.00	10.45	9.54	8.82	9.79	10.05	11.65	10.81	10.74	10.61	10.51	10.96	11.44	9.35	10.86
reading	13.22	13.65	13.33	11.26	10.84	11.22	10.32	11.06	10.50	10.93	10.27	11.27	13.35	16.22	15.87	16.07	15.48	14.85	14.11	13.69	12.88
$m = 6$	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	AVE
bear	11.75	8.72	8.82	8.49	8.87	8.99	8.53	9.09	9.36	8.86	8.48	8.23	8.47	8.21	12.33	9.71	8.59	9.90	9.12	8.69	9.16
buddha	12.41	15.06	15.08	15.12	15.66	14.95	15.48	14.77	13.53	12.36	11.86	11.81	13.25	14.49	15.32	14.55	15.12	13.97	13.39	13.85	14.10
cow	11.74	12.48	10.44	11.07	10.98	11.41	12.56	11.96	12.60	12.84	13.18	11.66	10.93	9.96	12.93	13.49	13.27	11.82	12.65	12.55	12.03
pot2	11.77	10.91	14.72	13.34	13.50	16.01	13.05	13.28	11.44	11.18	10.66	11.15	12.62	13.01	10.94	12.15	12.81	12.35	12.83	10.97	12.43
reading	14.58	14.60	14.66	11.93	11.48	12.17	10.90	12.24	10.82	10.96	10.03	12.18	15.88	17.41	16.66	16.42	16.14	15.40	15.19	13.98	13.68
$m = 4$	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	AVE
bear	13.33	11.69	10.92	11.20	12.94	12.25	11.82	11.12	10.43	10.47	10.88	10.87	12.42	11.35	17.58	11.71	11.06	11.97	15.20	11.49	12.03
buddha	15.09	16.54	17.60	17.61	18.50	19.22	18.93	17.74	16.83	14.13	14.48	16.11	15.20	18.13	20.05	17.99	18.41	16.46	15.17	15.65	16.99
cow	16.17	14.98	13.42	14.09	14.41	14.93	14.42	14.93	15.82	17.13	15.78	17.20	12.29	12.64	16.53	16.83	16.39	16.12	17.48	15.75	15.37
pot2	13.13	12.25	15.63	17.47	17.61	18.92	16.35	15.55	14.01	13.74	13.86	13.43	15.79	14.82	14.40	15.79	15.76	16.62	16.17	14.16	15.27
reading	15.69	16.75	16.42	13.05	14.46	17.18	14.26	12.24	12.60	12.05	11.81	13.87	16.57	19.46	17.60	19.38	18.59	18.48	16.10	16.12	15.63

Table 7: The prediction errors by SPS-Net (Trained on 32×32 patches of CyclesPS+) on the DiLiGenT-MV dataset. The values are mean angular errors per object, per view (in degrees). m indicates the number of different lights and the numbers (1 to 20) indicate the view index.

[10] B. Shi, P. Tan, Y. Matsushita, and K. Ikeuchi. Bi-polynomial modeling of low-frequency reflectances. *IEEE Trans. Pattern Anal. Mach. Intell.*, 36(6):1078–1091, 2014.

[11] P. Woodham. Photometric method for determining surface orientation from multiple images. *Opt. Engg*, 19(1):139–144, 1980.

[12] Z. Yao, K. Li, Y. Fu, H. Hu, and B. Shi. Gps-net: Graph-based photometric stereo network. *Proc. NIPS (NeurIPS)*, 2020.

m = 10	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	AVE
bear	10.58	5.92	6.35	6.84	6.32	6.37	6.93	6.09	7.76	6.37	6.31	6.60	5.70	6.52	9.11	6.87	7.04	7.15	6.57	5.77	6.86
buddha	10.63	12.55	13.78	13.46	13.33	12.90	12.89	11.90	10.93	9.89	9.59	9.36	10.79	11.73	13.72	12.64	12.69	12.33	11.48	12.11	11.93
cow	8.52	8.52	8.10	8.49	9.47	9.43	9.09	8.31	7.30	8.03	8.26	7.76	9.14	8.90	10.56	12.25	11.85	9.98	9.55	8.56	9.10
pot2	9.00	8.36	11.27	10.53	11.51	12.27	10.27	9.10	8.23	7.65	7.44	8.31	9.33	10.24	9.91	9.95	9.92	8.85	8.04	8.05	9.41
reading	14.52	14.26	14.10	13.20	13.09	15.30	13.19	14.09	12.90	12.72	12.10	13.37	15.18	17.27	16.08	16.96	16.30	16.42	15.33	14.38	14.54
m = 8	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	AVE
bear	10.90	6.45	6.92	7.46	6.66	6.87	7.27	6.55	9.34	7.26	7.11	7.17	6.22	6.84	10.56	7.23	7.71	8.05	7.00	6.10	7.48
buddha	10.78	12.86	13.75	13.42	13.93	14.45	13.72	12.66	11.25	10.37	10.19	9.65	11.28	12.28	13.55	13.22	13.18	12.50	11.74	12.09	12.34
cow	9.04	8.50	8.70	8.69	9.78	10.70	9.29	8.34	8.55	8.39	8.83	9.93	9.86	9.16	10.42	12.59	12.28	10.29	10.45	9.99	9.69
pot2	9.22	8.62	12.61	12.07	12.97	12.45	11.61	9.54	9.10	8.18	7.63	8.74	9.72	10.48	10.11	11.14	10.48	9.56	8.77	8.43	10.07
reading	14.15	13.69	13.43	13.46	12.92	12.67	12.69	13.82	13.18	12.09	12.39	12.97	15.07	18.11	17.03	16.76	15.92	17.47	14.57	14.57	14.35
m = 6	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	AVE
bear	9.96	8.28	8.42	7.90	8.81	7.76	8.32	7.43	9.61	7.54	8.09	7.76	6.89	7.63	14.05	8.47	8.43	8.96	9.17	8.14	8.58
buddha	12.03	13.66	14.49	14.87	15.31	14.87	16.05	15.73	11.22	11.41	11.21	11.11	11.80	12.77	16.29	14.41	13.82	13.37	12.53	13.11	13.50
cow	11.39	9.73	9.28	10.43	10.65	11.60	10.94	10.20	10.35	11.18	10.68	10.63	10.14	10.50	13.50	14.03	14.73	12.21	10.81	10.88	11.19
pot2	9.62	9.43	12.48	12.72	13.47	14.26	13.02	11.85	10.16	9.24	10.19	9.29	10.93	11.68	11.29	12.21	11.27	11.76	10.62	9.09	11.23
reading	14.79	14.51	13.22	13.92	13.17	13.45	12.73	12.59	12.14	12.10	10.77	13.09	16.18	19.04	16.49	16.37	17.00	16.54	14.87	15.40	14.42
m = 4	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	AVE
bear	17.57	10.59	10.86	11.39	11.38	13.82	12.12	10.51	13.34	11.19	13.83	10.28	9.87	10.85	17.03	12.41	13.27	13.00	11.24	9.74	12.21
buddha	14.75	16.14	17.36	17.79	16.86	17.94	21.01	18.28	15.60	14.32	13.12	15.16	13.72	17.51	20.28	16.95	17.09	15.64	14.72	15.47	16.49
cow	13.74	15.72	13.14	12.74	15.00	13.98	14.87	14.23	13.63	14.03	13.82	13.27	14.67	13.24	15.19	18.39	17.91	17.24	15.07	15.41	14.76
pot2	13.13	14.96	14.47	19.64	18.83	18.29	16.33	14.00	12.70	12.35	12.42	13.06	14.38	14.80	14.00	15.28	16.65	14.46	14.01	12.24	14.80
reading	17.47	18.43	18.36	18.26	15.34	17.05	18.47	14.34	14.95	13.91	14.85	15.01	16.28	21.07	19.61	18.39	19.56	18.43	17.62	16.75	17.21

Table 8: The prediction errors by SPS-Net (Trained on 8×8 patches of CyclesPS+) on the DiLiGenT-MV dataset. The values are mean angular errors per object, per view (in degrees). m indicates the number of different lights and the numbers (1 to 20) indicate the view index.