Supplementary Material
Spherical Regression:
Learning Viewpoints, Surface Normals and 3D Rotations on *n*-Spheres

Shuai Liao Efstratios Gavves Cees G. M. Snoek
QUVA Lab, University of Amsterdam

1. S^1: Viewpoint estimation with Euler angles

We show the viewpoint estimation network architecture used in this paper in Fig. 1. Given ResNet101 as backbone to provide a shared Pool5 feature (with 2048 output unit), we have 3 branches to estimate azimuth, elevation and in-plane rotation (theta) angels. Each branch begins with a fully-connected layer (Fc8), with 1024 output units, and makes a prediction for the 12 categories in Pascal3D+. Our prediction head is composed of two components: 1) absolute value prediction and 2) sign prediction.

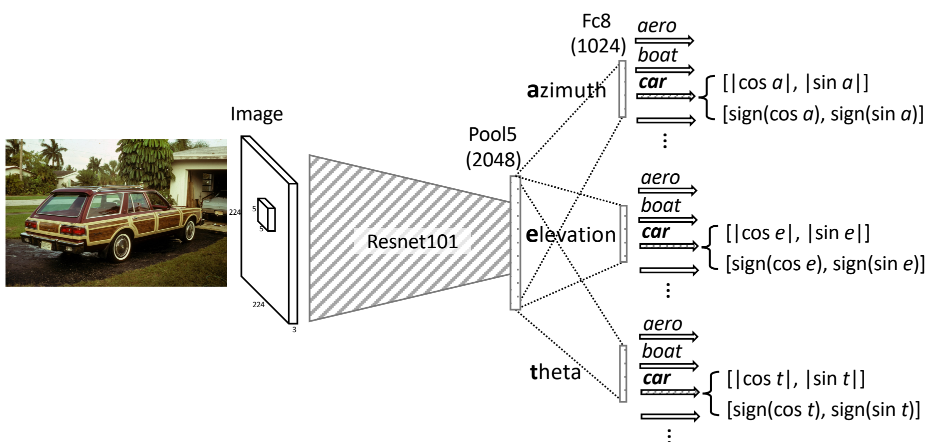

Figure 1. **Network architecture for viewpoint estimation by Euler angles on Pascal3D+**.

We show the fine-grained evaluation in Table. 1. In comparison with Penedones *et al*. [5], spherical regression improves the performance in all evaluation metrics, namely Acc@$\{\frac{\pi}{6}, \frac{\pi}{12}, \frac{\pi}{24}\}$.

We report the class-wise performance comparison in Table. 2. Prokudin *et al*. [6] wins the most categories under MedError metric (5 out of 12). However, they made a larger mistake on difficult categories like boat, where the visual appearance has larger variance. For Acc@$\frac{\pi}{6}$ metric, our method wins the most (6 out of 12 categories). In comparison with Penedones *et al*. [5], adding spherical regression module consistently helps increase the accuracy across almost all categories.

Table 1. **Viewpoint estimation with fine-grained evaluation on Pascal3D+**. We report results of **Acc**@$\{\frac{\pi}{6}, \frac{\pi}{12}, \frac{\pi}{24}\}\uparrow$. Results generated by spherical regression module (S^3_{exp}) have a better alignment to the ground truth models.

	MedErr↓	Acc@$\frac{\pi}{6}\uparrow$	Acc@$\frac{\pi}{12}\uparrow$	Acc@$\frac{\pi}{24}\uparrow$
Penedones *et al*. [5]†	11.6	83.6	66.3	35.9
This paper: [5]†+ S^1_{exp}	**9.2**	**88.2**	**74.1**	**46.0**

† Based on our implementation.

Table 2. **Category-wise evaluation of viewpoint estimation on Pascal3D+**.

	Method	aero	bike	boat	bottle	bus	car	chair	table	mbike	sofa	train	tv	mean
MedError	Mahendran *et al*. [2]	14.5	22.6	35.8	9.3	4.3	8.1	19.1	30.6	18.8	13.2	7.3	16.0	16.6
	Tulsiani *et al*. [8]	13.8	17.7	21.3	12.9	5.8	9.1	14.8	15.2	14.7	13.7	8.7	15.4	13.6
	Mousavian *et al*. [4]	13.6	12.5	22.8	8.3	3.1	5.8	11.9	12.5	12.3	12.8	6.3	11.9	11.1
	Su *et al*. [7]	15.4	14.8	25.6	9.3	3.6	6.0	9.7	10.8	16.7	9.5	6.1	12.6	11.7
	Penedones *et al*. [5]†	12.3	**11.5**	31.3	6.9	4.4	7.1	12.2	13.9	13.1	**7.7**	7.0	12.1	11.6
	Prokudin *et al*. [6]	9.7	15.5	45.6	**5.4**	**2.9**	**4.5**	13.1	12.6	**11.8**	9.1	**4.3**	12.0	12.2
	Grabner *et al*. [1]	10.0	15.6	**19.1**	8.6	3.3	5.1	13.7	11.8	12.2	13.5	6.7	11.0	10.9
	Mahendran *et al*. [3]	**8.5**	14.8	20.5	7.0	3.1	5.1	9.3	11.3	14.2	10.2	5.6	11.7	10.1
	This paper: [5]†+ S^1_{exp}	9.2	11.6	20.6	7.3	3.4	4.8	**8.2**	**8.5**	12.1	8.7	6.1	**10.1**	**9.2**
$Acc@\pi/6$	Mahendran *et al*. [2]	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
	Tulsiani *et al*. [8]	0.81	0.77	0.59	0.93	**0.98**	0.89	0.80	0.62	0.88	0.82	0.80	0.80	0.808
	Mousavian *et al*. [4]	0.78	0.83	0.57	0.93	0.94	0.90	0.80	0.68	0.86	0.82	0.82	0.85	0.810
	Su *et al*. [7]	0.74	0.83	0.52	0.91	0.91	0.88	0.86	0.73	0.78	0.90	**0.86**	0.92	0.820
	Penedones *et al*. [5]†	0.80	0.85	0.48	**0.96**	0.94	0.91	0.84	0.70	0.86	0.95	0.84	0.91	0.836
	Prokudin *et al*. [6]	**0.89**	0.83	0.46	**0.96**	0.93	0.90	0.80	**0.76**	0.90	0.90	0.82	0.91	0.838
	Grabner *et al*. [1]	0.83	0.82	**0.64**	0.95	0.97	0.94	0.80	0.71	0.88	0.87	0.80	0.86	0.839
	Mahendran *et al*. [3]	0.87	0.81	**0.64**	**0.96**	0.97	**0.95**	0.92	0.67	0.85	0.97	0.82	0.88	0.859
	This paper: [5]†+ S^1_{exp}	0.88	**0.88**	0.61	**0.96**	0.97	0.93	**0.93**	0.74	**0.93**	**0.98**	0.84	**0.95**	**0.882**

† Based on our implementation.

2. S^2: Surface normal estimation

We show the visualization of surface normal prediction in Fig. 2. The results from Zhang *et al*. [9] are smoother than our results from spherical regression, but it makes some mistake with quite large surface area, *e.g*. the wall on the picture at row 3 column 2. In terms of boundaries, our results tend to be sharper. This is mainly due to the classification branch, which forces the prediction to choose the main direction in one out of four quadrants. Overall, our results maintain more details than Zhang *et al*. [9].

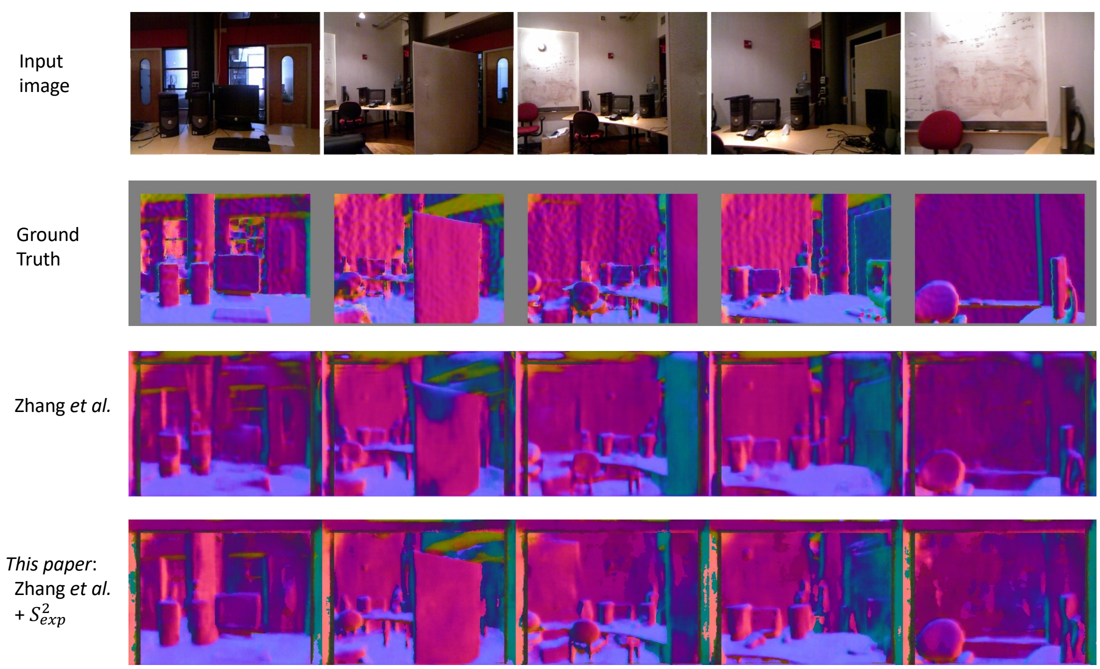

Figure 2. **Visualization of Surface Normal Estimation on *NYU v2***. Predictions are made by model: "Zhang *et al*. [9]" and "Zhang *et al*. [9] + S^2_{exp}". While results from Zhang *et al*. [9] are smoother, our method generates sharp boundaries and thus maintains details.

3. S^3: 3D Rotation estimation with quaternions

We show a class-wise performance comparison based on Acc@$\frac{\pi}{6}$ in Fig. 3. Since we are predicting the 3D rotation just from a single image, it can be seen that categories with high degree of symmetry have worse performance, *e.g.* bathtub, desk, night-stand and table. In comparison with the regression of quaternion with flat VGG16, spherical regression consistently helps increase the accuracy.

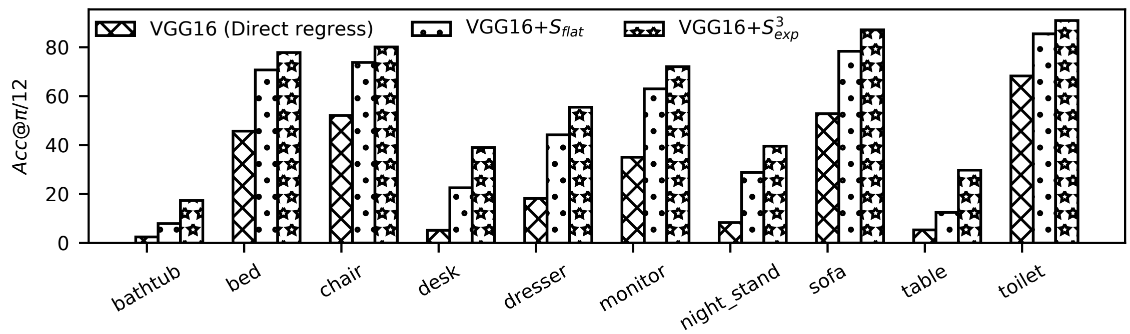

Figure 3. **Class-wise comparison of 3D rotation estimation on *ModelNet10-SO3***. Categories with high degree of symmetry are observed to have worse performance, *e.g.* bathtub, desk, night-stand and table. Spherical regression module (S^3_{exp}) consistently helps increase the performance over flat regression of quaternion by VGG16.

We show a visualization of 3D rotation estimation in Fig. 4. The first row is the ground truth input images. We render the predicted rotations from VGG16 and VGG16+S^3_{exp} in second and third rows. We can see our result have a better alignment to the ground truth models.

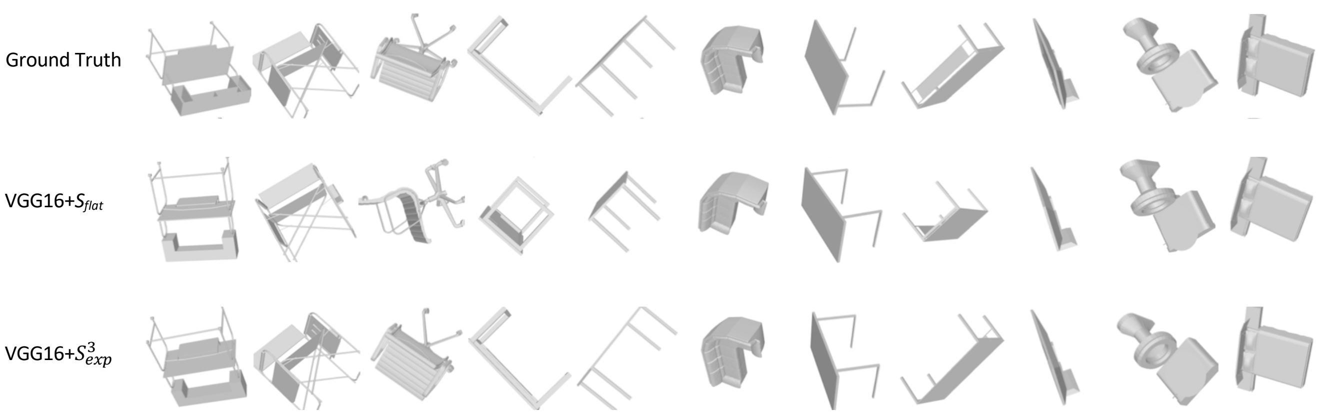

Figure 4. **Visualization of 3D rotation estimation on *ModelNet10-SO3***.

4. Derivation of Jacobian for S_{flat} and S_{exp}

.

First, we provide detailed derivation of Eq. 7 in the main paper. Given the ℓ_2 normalization form:

$$p_j = g(o_j; \boldsymbol{O}) = \frac{f(o_j)}{\sqrt{\sum_k f(o_k)^2}}$$

with arbitrary univariate mapping $f(\cdot)$, we have:

$$\frac{\partial p_j}{\partial o_i} = \frac{\frac{df(o_j)}{do_i} \cdot A - f(o_j) \cdot \frac{\partial A}{\partial o_i}}{A^2} \tag{1}$$

$$= \frac{\frac{df(o_j)}{do_i} \cdot A - f(o_j) \cdot p_i \cdot \frac{df(o_i)}{do_i}}{A^2} \tag{2}$$

$$= \frac{1}{A} \left[\frac{df(o_j)}{do_i} - p_i \cdot p_j \cdot \frac{df(o_i)}{do_i} \right] \tag{3}$$

$$= \begin{cases} \frac{f'(o_i)}{A} \cdot (1 - p_i \cdot p_j), & \text{when j=i} \\ \frac{f'(o_i)}{A} \cdot (0 - p_i \cdot p_j), & \text{when j} \neq \text{i} \end{cases} \tag{4}$$

where $A = \sqrt{\sum_k f(o_k)^2}$.

Thus the Jacobian matrix of $g : \boldsymbol{O} \to \boldsymbol{P}$ is as follows

$$\mathbf{J}_g = \frac{\partial \boldsymbol{P}}{\partial \boldsymbol{O}} = \left[\frac{\partial \boldsymbol{P}}{\partial o_0}, \frac{\partial \boldsymbol{P}}{\partial o_1}, \cdots, \frac{\partial \boldsymbol{P}}{\partial o_n} \right] \tag{5}$$

$$= \begin{bmatrix} \frac{\partial p_0}{\partial o_0} & \frac{\partial p_0}{\partial o_1} & \cdots & \frac{\partial p_0}{\partial o_n} \\ \frac{\partial p_1}{\partial o_0} & \frac{\partial p_1}{\partial o_1} & \cdots & \frac{\partial p_1}{\partial o_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial p_n}{\partial o_0} & \frac{\partial p_n}{\partial o_1} & \cdots & \frac{\partial p_n}{\partial o_n} \end{bmatrix} \tag{6}$$

$$= \begin{bmatrix} 1 - p_0 p_0 & -p_1 p_0 & \cdots & -p_n p_0 \\ -p_0 p_1 & 1 - p_1 p_1 & \cdots & -p_n p_1 \\ \vdots & \vdots & \ddots & \vdots \\ -p_0 p_n & -p_1 p_n & \cdots & 1 - p_n p_n \end{bmatrix} \begin{bmatrix} \frac{f'(o_0)}{A} & & & \\ & \frac{f'(o_1)}{A} & & \\ & & \ddots & \\ & & & \frac{f'(o_n)}{A} \end{bmatrix} \tag{7}$$

$$= \left(\boldsymbol{I} - \begin{bmatrix} p_0 p_0 & p_1 p_0 & \cdots & p_n p_0 \\ p_0 p_1 & p_1 p_1 & \cdots & p_n p_1 \\ \vdots & \vdots & \ddots & \vdots \\ p_0 p_n & p_1 p_n & \cdots & p_n p_n \end{bmatrix} \right) \begin{bmatrix} \frac{f'(o_0)}{A} & & & \\ & \frac{f'(o_1)}{A} & & \\ & & \ddots & \\ & & & \frac{f'(o_n)}{A} \end{bmatrix} \tag{8}$$

4.1. $\mathcal{S}_{flat}$ case

In this case, we only take flat ℓ_2 normalization on $\boldsymbol{O}$ to obtain $\boldsymbol{P}$, namely $p_j = g(o_j; \boldsymbol{O}) = \frac{o_j}{\sqrt{\sum_k o_k^2}}$. This means $f(o_i) = o_i$ and $f'(o_i) = 1$. Thus Eq. 8 becomes:

$$\mathbf{J}_{\mathcal{S}_{flat}} = \frac{\partial \boldsymbol{P}}{\partial \boldsymbol{O}} \tag{9}$$

$$= \left(\boldsymbol{I} - \begin{bmatrix} p_0 p_0 & p_1 p_0 & \cdots & p_n p_0 \\ p_0 p_1 & p_1 p_1 & \cdots & p_n p_1 \\ \vdots & \vdots & \ddots & \vdots \\ p_0 p_n & p_1 p_n & \cdots & p_n p_n \end{bmatrix} \right) \begin{bmatrix} \frac{1}{A} & & & \\ & \frac{1}{A} & & \\ & & \ddots & \\ & & & \frac{1}{A} \end{bmatrix} \tag{10}$$

$$= [\frac{\partial \boldsymbol{P}}{\partial o_0}, \frac{\partial \boldsymbol{P}}{\partial o_1}, \cdots, \frac{\partial \boldsymbol{P}}{\partial o_n}] \tag{11}$$

$$= (\boldsymbol{I} - \boldsymbol{P} \otimes \boldsymbol{P}) \cdot \frac{1}{A} \tag{12}$$

where $\otimes$ denotes outer product.

4.2. $\mathcal{S}_{exp}$ case

In this case, we take spherical normalization on $\boldsymbol{O}$ to obtain $\boldsymbol{P}$, namely $p_j = g(o_j; \boldsymbol{O}) = \frac{e^{o_j}}{\sqrt{\sum_k (e^{o_k})^2}}$. This means $f(o_i) = e^{o_i}$ and $f'(o_i) = e^{o_i}$. Thus Eq. 8 becomes:

$$
\begin{aligned}
\mathbf{J}_{\mathcal{S}_{exp}} &= \frac{\partial \boldsymbol{P}}{\partial \boldsymbol{O}} && (13)\\
&= \left(\boldsymbol{I} - \begin{bmatrix} p_0p_0 & p_1p_0 & \cdots & p_np_0 \\ p_0p_1 & p_1p_1 & \cdots & p_np_1 \\ \vdots & \vdots & \ddots & \vdots \\ p_0p_n & p_1p_n & \cdots & p_np_n \end{bmatrix} \right) \begin{bmatrix} p_0 & & & \\ & p_1 & & \\ & & \ddots & \\ & & & p_n \end{bmatrix} && (14)\\
&= (\boldsymbol{I} - \boldsymbol{P} \cdot \boldsymbol{P}^T) \cdot diag(\boldsymbol{P}) && (15)\\
&= (\boldsymbol{I} - \boldsymbol{P} \otimes \boldsymbol{P}) \cdot diag(\boldsymbol{P}) && (16)
\end{aligned}
$$

where $\otimes$ denotes outer product.

References

[1] Alexander Grabner, Peter M Roth, and Vincent Lepetit. 3d pose estimation and 3d model retrieval for objects in the wild. In *CVPR*, 2018.

[2] Siddharth Mahendran, Haider Ali, and René Vidal. 3d pose regression using convolutional neural networks. In *ICCV*, 2017.

[3] Siddharth Mahendran, Haider Ali, and Rene Vidal. A mixed classification-regression framework for 3d pose estimation from 2d images. In *BMVC*, 2018.

[4] Arsalan Mousavian, Dragomir Anguelov, John Flynn, and Jana Košecká. 3d bounding box estimation using deep learning and geometry. In *CVPR*, 2017.

[5] Hugo Penedones, Ronan Collobert, Francois Fleuret, and David Grangier. Improving object classification using pose information. Technical report, Idiap, 2012.

[6] Sergey Prokudin, Peter Gehler, and Sebastian Nowozin. Deep directional statistics: Pose estimation with uncertainty quantification. In *ECCV*, 2018.

[7] Hao Su, Charles R Qi, Yangyan Li, and Leonidas J Guibas. Render for cnn: Viewpoint estimation in images using cnns trained with rendered 3d model views. In *ICCV*, 2015.

[8] Shubham Tulsiani and Jitendra Malik. Viewpoints and keypoints. In *CVPR*, 2015.

[9] Yinda Zhang, Shuran Song, Ersin Yumer, Manolis Savva, Joon-Young Lee, Hailin Jin, and Thomas Funkhouser. Physically-based rendering for indoor scene understanding using convolutional neural networks. In *CVPR*, 2017.