# *Supplementary Material:* A Spherical Approach to Planar Semantic Segmentation

Chao Zhang*,[1]
chao.zhang@crl.toshiba.co.uk
Sen He*,[2]
sh752@exeter.ac.uk
Stephan Liwicki[1]
stephan.liwicki@crl.toshiba.co.uk

[1] Cambridge Research Laboratory
Toshiba Europe Limited
Cambridge, United Kingdom

[2] Department of Computer Science
University of Exeter
Exeter, United Kingdom

## 1 Geometric Distortion

An image consists of rays that represent a reduced description of the 3D world. Specifically, using camera calibration matrix $\mathbf{K}$ for planar images, pixel $\mathbf{p}_p$ of a feature at 3D location $\mathbf{P} = [P_x P_y P_z]^\mathrm{T}$ is represented by

$$\mathbf{p}_p = \mathbf{K}\frac{\mathbf{P}}{P_z} \propto \mathbf{K}\mathbf{P}. \tag{1}$$

Let us consider the distance between pixels $\mathbf{p}_p$ and $\mathbf{q}_p$, representing rays towards 3D features $\mathbf{P}$ and $\mathbf{Q}$ respectively. During non-zero camera translation $\mathbf{t} = [t_x t_y t_z]^\mathrm{T}$, the pixel distance changes to

$$\left\| \mathbf{K}\left(\frac{\mathbf{P}-\mathbf{t}}{P_z - t_z} - \frac{\mathbf{Q}-\mathbf{t}}{Q_z - t_z}\right)\right\|, \tag{2}$$

which is equivalent to the original distance $\|\mathbf{p}_p - \mathbf{q}_p\|$ only when $t_z = 0$ and $P_z = Q_z$. Thus, translation based distortion is frequently observed on the image plane. Similarly, with out-of-plane rotation $\mathbf{R}$, it typically holds that

$$\left\| \mathbf{K}\left(\frac{\mathbf{R}^\mathrm{T}\mathbf{P}}{[0\,0\,1]\mathbf{R}^\mathrm{T}\mathbf{P}} - \frac{\mathbf{R}^\mathrm{T}\mathbf{Q}}{[0\,0\,1]\mathbf{R}^\mathrm{T}\mathbf{Q}}\right)\right\| \neq \|\mathbf{p}_p - \mathbf{q}_p\|. \tag{3}$$

On a sphere, pixel $\mathbf{p}_s$ provides a unit vector, such that

$$\mathbf{p}_s = \frac{\mathbf{P}}{\|\mathbf{P}\|} \propto \mathbf{P}. \tag{4}$$

Now, while the distance between two vertices $\|\mathbf{p}_s$ - $\mathbf{q}_s\|$ is still likely to change during camera translations, for arbitrary camera rotation $\mathbf{R}$ we get

$$\left\| \frac{\mathbf{R}^\mathrm{T}\mathbf{P}}{\|\mathbf{R}^\mathrm{T}\mathbf{P}\|} - \frac{\mathbf{R}^\mathrm{T}\mathbf{Q}}{\|\mathbf{R}^\mathrm{T}\mathbf{Q}\|}\right\| = \left\|\mathbf{R}^\mathrm{t}\left(\frac{\mathbf{P}}{\|\mathbf{P}\|} - \frac{\mathbf{Q}}{\|\mathbf{Q}\|}\right)\right\|, \tag{5}$$

since rotations are orthonormal. Therefore, while planar images observe distortions, no distortion is observed for camera rotations in the spherical image domain.

 *Equal Contribution

Finally we hypothesize that on spherical images a CNN will be able to generalize to varying pixel locations without added need for learning distortions that otherwise arise in planar images. As we strive for a conformation of this, we note, since most datasets provide planar images, we employ the camera calibration matrix to project to the sphere. Specifically, from (1) and (4) we derive

$$\mathbf{p}_s = \frac{\mathbf{K}^{-1}\mathbf{p}_p}{\|\mathbf{K}^{-1}\mathbf{p}_p\|}. \tag{6}$$

## 2 Improved Run-time for Graph-convolution

The convolution in (2) of the paper is written as $f \circledast k$, where

$$f = \begin{bmatrix} \alpha_i^{(r)} q_1^i + (1-\alpha_i^{(r)}) q_6^i \\ \alpha_i^{(r)} q_2^i + (1-\alpha_i^{(r)}) q_1^i \\ \alpha_i^{(r)} q_3^i + (1-\alpha_i^{(r)}) q_2^i \\ \alpha_i^{(r)} q_4^i + (1-\alpha_i^{(r)}) q_3^i \\ \alpha_i^{(r)} q_5^i + (1-\alpha_i^{(r)}) q_4^i \\ \alpha_i^{(r)} q_6^i + (1-\alpha_i^{(r)}) q_5^i \\ p_i \end{bmatrix}_{i=1}^{M^{(r)}}, k = \begin{bmatrix} w_1 \\ w_2 \\ w_3 \\ w_4 \\ w_5 \\ w_6 \\ w_7 \end{bmatrix}. \tag{7}$$

We define $f_j^{(r)} = \begin{bmatrix} q_j^i \\ q_{j+1}^i \end{bmatrix}_{i=1}^{M^{(r)}}$, and rewrite convolutions:

$$\begin{aligned} & [p_i]_{i=1}^{M^{(r)}} \circledast [w_7] + [1-\alpha_i^{(r)}]_{i=1}^{M^{(r)}} \odot \left( f_5^{(r)} \circledast \begin{bmatrix} w_6 \\ w_1 \end{bmatrix} \right) \\ + & \quad [\alpha_i^{(r)}]_{i=1}^{M^{(r)}} \odot \left( \sum_{j\in\{1,3,5\}} f_j^{(r)} \circledast \begin{bmatrix} w_j \\ w_{j+1} \end{bmatrix} \right) \\ + & \quad [1-\alpha_i^{(r)}]_{i=1}^{M^{(r)}} \odot \left( \sum_{j\in\{1,3\}} f_j^{(r)} \circledast \begin{bmatrix} w_{j+1} \\ w_{j+2} \end{bmatrix} \right), \end{aligned} \tag{8}$$

where $\odot$ computes element-wise multiplication. Notice, only three $M^{(r)} \times 2$ feature maps, *i.e.* $f_1^{(r)}$, $f_3^{(r)}$ and $f_5^{(r)}$, need to be gathered once for the convolution. We emphasize, operations are sequential to reduce memory needs.

While the mask can be arbitrary, we can further reduce memory and run-time requirements, if vertices in the mask are highly connected (as is the case in image data). Specifically, we utilize that the neighborhoods of vertices frequently coincide, *i.e.* often there exists two vertices $p_i$ and $p_k$ such that $q_{j+1}^i = q_j^k$. We rearrange the feature maps such that neighborhood connectivity is optimized (Fig. 1):

$$g_j^{(r)} = \begin{bmatrix} \dots & q_j^i & q_{j+1}^i = q_j^k & q_{j+1}^k & \dots \end{bmatrix} \tag{9}$$

of size $L_j^{(r)} \times 1$, where $M^{(r)} < L_j^{(r)} \ll 2M^{(r)}$. Now, a $1 \times 2$ convolution can be applied over the $1 \times L_j^{(r)}$ feature map, replacing $f_j^{(r)} \circledast \begin{bmatrix} w_j & w_{j+1} \end{bmatrix}^\intercal$. Note, the indices of $g_j^{(r)}$ are precomputed.

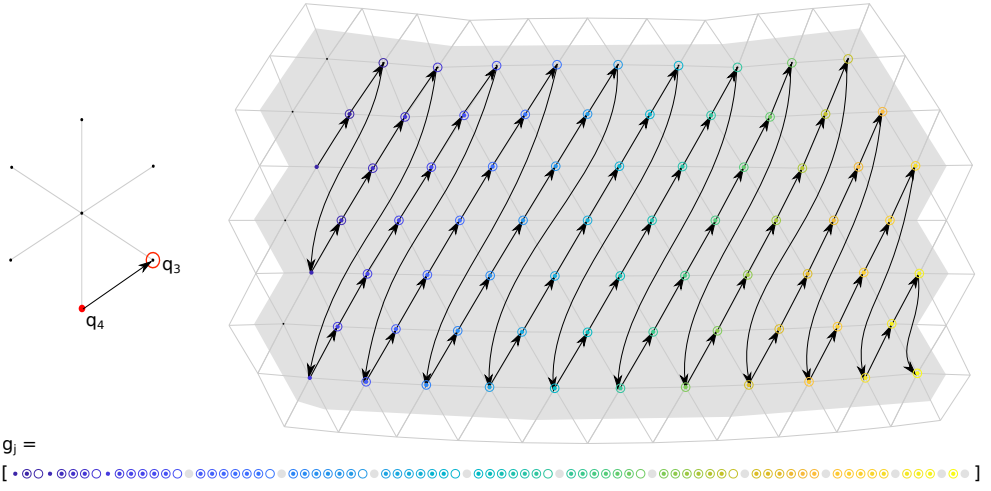


Figure 1: Rather than gathering a $2 \times M$ feature map for $q_3$ and $q_4$, we extract $g_j$ of size $1 \times L$, where $L \ll 2M$ (*e.g.* $L = 83$ vs $M = 72$ for Cityscapes at $r = 5$). Zero-padding is denoted by greyed out circles.

| Dataset | Resolution | U-Net mIoU(%) | DANet mIoU (%) | |
|---|---|---|---|---|
| | | | No Pretraining | Pretraining |
| Cityscapes | planar@1/6 | 51.5 | 51.2 | 57.9 |
| | hexagonal@1/6 | 51.4 | 51.1 | 57.1 |
| | planar@1/3 | 55.5 | 63.0 | 67.0 |
| | hexagonal@1/3 | 55.2 | 62.9 | 66.9 |

Table 1: Ablation study for hexagonal kernel without spherical projection (hexagonal). Overall, hexagonal performs very comparable to planar, but consistently with slightly reduced accuracy.

## 3 Checking Kernel Bias

In this section, we check if the improved results are due to the hexagonal filter, rather than the spherical projection. In particular, using the method applied to pretraining in §4 of the main paper, we now apply a hexagonal kernel on a planar version of Cityscapes. Table 1 shows the results, where the hexagonal kernel consistently performs with slightly less accuracy to standard methods with $3 \times 3$ kernels. We conclude, the hexagonal kernel is not the reason for improved results.

## 4 Per-class Results

In Table 2 and Table 3, we show the per-class mIoU results for planar and spherical representation on Synthia-S and Cityscapes. Both U-Net and DANet results are shown and compared. Note, 'bike' class in Synthia-S only takes 0.00055% of the whole dataset.

On Synthia-S, results of planar input at 1/3 resolution are shown. For spherical representation, results for input at a level-8 mesh are presented. Overall, the performance of U-Net is

| Method | misc | sky | build. | road | swalk | fence | veg. | pole | car | sign | pedes. | bike | lmark | mIoU |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| unet@1/3 | **29.5** | 92.6 | **88.2** | 92.8 | 88.3 | 19.6 | 70.8 | 45.2 | 79.0 | 12.0 | 37.7 | 0.0 | 78.9 | 56.5 |
| unet@8 | 23.7 | **93.1** | 86.7 | **94.0** | **89.7** | **22.1** | 70.8 | **46.5** | **83.7** | **17.2** | **40.6** | 0.0 | **80.1** | **57.5** |
| danet@1/3 | 29.0 | 92.1 | **89.6** | 92.1 | 87.8 | 10.4 | 69.6 | **40.3** | 82.5 | **14.9** | 29.4 | 0.0 | 75.4 | 54.8 |
| danet@8 | **32.0** | **92.8** | 89.4 | **93.2** | 87.8 | **21.1** | **70.9** | 38.5 | **82.8** | 12.3 | **29.9** | 0.0 | **77.0** | **56.0** |
| danet@1/3⋆ | **35.7** | 91.9 | **90.9** | 92.4 | **89.9** | 15.1 | 71.2 | 41.2 | **88.6** | **30.3** | **43.8** | 0.0 | 71.9 | 58.6 |
| danet@8⋆ | 33.8 | **92.2** | 89.8 | 93.3 | 89.7 | **22.5** | 71.2 | **45.2** | 84.1 | 24.5 | 42.0 | 0.0 | **75.5** | **58.7** |

Table 2: Per-class mIoU comparison on Synthia-S. 1/3 resolution and level-8 mesh are used for planar and spherical input, respectively. ⋆ indicates pre-training for DANet.

competitive when comparing to DANet without pre-training. Moreover, U-Net performs best in classes 'lanemarking' and 'pole', etc. With pre-training, DANet gains large improvements in classes such as 'sign' and 'pedestrian'.

On Cityscapes, 1/3 for planar images and level-10 spherical results are compared. Comparing U-Net and DANet without pre-training, DANet demonstrate its strength at classes 'truck', 'bus' and 'train'. With pre-training the performance of multiple classes are further improved. Among them, 'truck', 'sign' and 'motorbike' are the ones which benefit significantly.

| Method | road | swalk | build. | wall | fence | pole | tlight | sign | veg. | terrain | sky | person | rider | car | truck | bus | train | mbile | bike | mIoU |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| unet@1/3 | 95.4 | **71.3** | 85.4 | **27.4** | 34.3 | 45.7 | **47.9** | 59.3 | 88.5 | **52.4** | 90.1 | 62.3 | 33.3 | 86.8 | 22.2 | **43.5** | 24.7 | 21.4 | 62.9 | 55.5 |
| unet@10 | **95.7** | 71.1 | **85.5** | 27.1 | **34.5** | **46.0** | 46.5 | **62.0** | **88.6** | 49.3 | **91.2** | **62.4** | **36.8** | **87.0** | **23.4** | 42.1 | **27.1** | **29.0** | **63.5** | **56.3** |
| danet@1/3 | **96.7** | **76.0** | 86.6 | **40.5** | **41.8** | 43.2 | 45.6 | 56.5 | 87.9 | **57.3** | 90.7 | 64.2 | 42.1 | 89.0 | 54.8 | **71.4** | **59.2** | 32.3 | 61.4 | 63.0 |
| danet@10 | 96.3 | 75.2 | **87.1** | 33.5 | 41.7 | **46.6** | **52.2** | **61.6** | **88.7** | 56.4 | **91.5** | **66.6** | **44.5** | **89.4** | **55.9** | 62.2 | 49.1 | **36.0** | **63.7** | **63.1** |
| danet@1/3⋆ | **97.3** | **79.5** | 88.5 | **45.2** | **46.8** | 48.2 | 54.3 | 65.4 | 89.3 | **61.1** | 91.5 | 70.0 | **49.9** | **91.5** | 58.4 | 67.4 | **57.1** | 42.5 | 66.7 | 66.9 |
| danet@10⋆ | 97.1 | 78.8 | 88.5 | 38.7 | 43.2 | **51.3** | **57.3** | **68.1** | **89.5** | 58.0 | **91.9** | **71.0** | 49.1 | 91.4 | **69.6** | **75.6** | 56.9 | **44.4** | **67.3** | **67.8** |

Table 3: Per-class mIoU comparison on Cityscapes. 1/3 resolution and level-10 mesh are used for planar and spherical input, respectively. ⋆ indicates pre-training for DANet.