Beyond Fixed Grid: Learning Geometric Image Representation with a Deformable Grid

Supplementary Material

Jun Gao[1,2,3], Zian Wang[1,2], Jinchen Xuan[4], and Sanja Fidler[1,2,3]

[1]University of Toronto, [2]Vector Institute, [3]NVIDIA, [4]Peking University
{jungao, zianwang, fidler}@cs.toronto.edu
1600012865@pku.edu.cn

In the supplementary material, we provide additional details for our applications in Sec. 1, ablation studies in Sec. 2, as well as additional experimental results in Sec. 3.

1 Applications Details

We here provide details for our boundary-based object annotation approach. Details for other applications were presented in the main paper.

1.1 Boundary-based Object Annotation Method

Following the notation introduced in the main paper, we search a closed path that has minimal distance transform energy in the deformed grid. The distance transform is a distance map which labels each pixel of the image with the distance to the nearest boundary pixels. We first use PSP-Deeplab to extract the feature map, and apply two 3×3 conv filters with batch normalization and relu activation to predict the distance transform map. The model is trained with L2 loss. Note that the PSP-Deeplab encoder is shared among DT preditor, Curve-GCN predictor and grid decoder. We denote the path as: $Q = \{v_{Q_1}, v_{Q_2}, \cdots, v_{Q_M}\}$, where v_{Q_i} are the vertices in the grid, and two consecutive vertices need to be connected with an edge in the grid. Suppose Curve-GCN [8] predicts M control points, which we denote as $\mathbf{cp}_1, \mathbf{cp}_2, \cdots, \mathbf{cp}_M$. For every control point $\mathbf{cp}_i$, we first find its top-k nearest vertices, $v_{\mathbf{cp}_i^1}, v_{\mathbf{cp}_i^2}, \cdots, v_{\mathbf{cp}_i^k}$, in the deformed grid. For each vertex, we compute its distance transform energy via bilinear sampling using the vertex's position in the predicted distance transform energy map. We then snap each control point to the vertex that has minimal distance transform energy among its top-k closest vertices. Specifically:

$$v_{Q_i} = \arg\min_{v_{\mathbf{cp}_i^k}} DT(v_{\mathbf{cp}_i^k}). \tag{1}$$

To search a path from v_{Q_i} to $v_{Q_{(i+1)\%M}}$, we use the Dijkstra Algorithm[1]. Specifically, we first construct the same graph as for the deformable grid. To get the

[1] https://en.wikipedia.org/wiki/Dijkstra%27s_algorithm

energy of each edge, we compute the average distance transform energy of the uniformly sampled points along the edge. We then use the Dijkstra Algorithm to find the minimal energy path between two points. Computing the path this way introduces minimal computational overhead on top of the runtime of CurveGCN, since the number of grid's vertex is relatively small.

2 Ablations

In this section, we ablate different choices for the grid topology, different loss combinations and different image encoders for superpixel experiments.

2.1 Experimental Settings

Datasets: We train all our models on Cityscapes-Multicomp dataset [2], which are demonstrated in the main paper. We use the same training, validation and testing split as in CurveGCN [8] and DELSE [11].

Training Details: We set the hyperparameters as follows: $\lambda_{\text{recons}}, \lambda_{\text{area}}$, and λ_{lap} are set to 0.5, 0.02, 0.02, respectively. The δ is set to $\delta = 0.001 * \frac{20}{\text{grid-size}}$. We experiment with grid size set to 30×30. When the one-hot segmentation mask is available, we multiply the values in one-hot mask by 0.5 when appending them to RGB values, which is first scaled to $[0, 1]$. We train all models using the Adam [5] optimizer with 1e-4 learning rate and 5e-4 weight decay.

Evaluation Metrics: Since we ablate variants of the deformable grid, we utilize metrics typically employed by superpixel methods. Specifically, we report the Boundary Precision (BP), Boundary Recall (BR), and Achievable Segmentation Accuracy (ASA). The ground truth is the human annotated segmentation mask that is stretched according to the (stretched) bounding box. All the metrics are averaged over all test images. The pixel tolerance for BP and BR is set to 3.

2.2 Experimental Results

The quantitative results are reported in Table 1.

Different Grid Topologies: We first ablate different topologies that are listed in the main paper. For a fair comparison, we set the grid size of topology-a to be 43×43, which gives $42 \times 42 = 1764$ number of grid cells (treated as superpixels), and set the grid size of topology-b to be 22×22, which gives $21 \times 21 \times 4 = 1764$ number of grid cells, while topology-c and topology-d have $29 \times 29 \times 2 = 1682$ number of grid cells. Topology-d achieves the best performance in terms of BR-1, BP-1, BP-2, BP-3 and ASA, which is the main choice of topology for all other experiments.

Grid Size	N Enc. Layer	+ mask	+ recons.	Top-a	Top-b	Top-c	Top-d	BR-1	BR-2	BR-3	BP-1	BP-2	BP-3	ASA
30x30	5	✓	✓	✓				90.44	99.97	100.00	4.36	7.31	10.20	98.55
30x30	5	✓	✓		✓			96.98	99.98	100.00	4.84	8.08	11.19	98.62
30x30	5	✓	✓			✓		95.90	99.99	100.00	4.66	7.77	10.77	98.66
30x30	5	✓	✓				✓	**97.32**	99.99	100.00	**5.03**	**8.34**	**11.51**	**98.72**
30x30	5	✓					✓	95.55	**99.99**	100.00	4.43	7.41	10.30	98.50
30x30	5		✓				✓	94.77	99.99	100.00	4.36	7.29	10.13	98.44
30x30	3	✓	✓				✓	96.32	99.99	100.00	4.77	7.94	10.99	98.60
30x30	3		✓				✓	94.98	99.99	100.00	4.40	7.36	10.22	98.45
30x30	1	✓	✓				✓	95.71	99.99	100.00	4.58	7.63	10.58	98.52
30x30	1		✓				✓	94.75	99.99	100.00	4.37	7.29	10.13	98.44

Table 1: **Ablation study on different variants of our Deformable Grid.** "Top-" here denotes the choice of topology, and "N Enc Layer" reports which layer from ResNet we employ as the encoder.

Different Loss Combinations: We ablate the use of the differentiable reconstruction loss as well as the use of the segmentation mask in the differentiable variance loss. Adding the differentiable reconstruction loss helps to align grid edges with image boundaries. The segmentation mask provides strong signal for learning the semantic boundaries.

Different Encoders: To ablate different image encoders, we choose different output layers in ResNet as the feature map for the grid decoder. Specifically, we experimented with layer1, layer3 and layer5. The deformable grid model achieves competitive performance with shallow feature maps. With deeper neural networks, the segmentation masks help more, as the network typically learns the semantics in deeper layers, and thus has better semantic boundary alignment (as opposed to exploiting image gradients alone).

3 Experimental Details

3.1 Superpixels

In this section, we frist provide details of experiments in the main paper, and further show superpixel experiments using another network structure as the image encoder as well as adding supervisory loss in the form of a one-hot mask.

Datasets As each image in the BSDS500 [1] is provided with multiple ground-truth annotations, following SSN [4], we treat each annotation as an independent sample for both training and evaluation. In total, we have 1633 training pairs and 1063 test pairs of images and annotations.

Network Architecture: We use a shallow network, AffinityNet from SEAL [10] as the image encoder. For each vertex in the grid, we use bilinear sampling to extract the feature from the feature map using vertex's position in the image plane. Our grid decoder is a 4-layer Graph Convolutional Network [6,8] that predicts the offset for each vertex. We train the network purely with unsupervised losses, where the pixel feature $\mathbf{f}$ uses only RGB colors.

Training Details: We train the models from scratch. The number of grid cells K is set to 3200. We train the network using Adam [5] optimizer with 1e-4 learning rate and 5e-4 weight decay. Hyperparameters $\lambda_{\text{recons}}, \lambda_{\text{area}}$, and λ_{lap} are set to $0.5, 0.025, 0.025$, respectively. We apply random image cropping, reflections and resizing as data augmentation to train our model. For supervised agglomerative clustering, we average the RGB and learned distances by weights of 0.6 and 0.4 respectively. We then run hierarchical merge on the combined distances of cells.

A comprehensive evaluation is also provided. We use a modified version of the PSPNet architecture [12] as our feature extractor. Specifically, the layer1, layer2, layer3 and layer4 feature maps from the ResNet backbone [3] and the feature map from the PSP module all go through one convolution layer to reduce the number of feature channel to 64. We then perform bilinearly upsampling to the original image size and concatenate all these feature maps to get a feature map of size $320 \times w \times h$, where $w \times h$ is the size of the image. We apply another convolution layer to the concatenated features to get the pixel-wise feature map. The grid decoder is the same as described in the main paper.

For the supervised setting, when the annotated masks for training images are provided, we also learn the affinity for use with agglomerative clustering. We first obtain the feature for each grid cell using the mean feature of every pixel inside the cell. For each two cells, we concatenate their features and pass them through a 4-layer fully-connected network to predict the affinity between two cells. The network is trained using Binary Cross Entropy Loss, and the ground truth is 1 if two cells have same annotation, otherwise 0. We evaluate all methods using BP, BR and ASA, and we show the results when the tolerance is set to 1, 2 and 3 pixels.

3.2 Object Instance Annotation

We use an eight-layer GCN for the grid decoder to predict grid deformation. The feature map is the final feature of our image encoder. We use the same hyperparameters settings as provided in Sec. 2. We set $k = 2$ when running experiments for the grid size 20×20, and $k = 3$ for the grid size 30×30 and 40×40. For the comparisons with pixel-wised methods on Cityscapes-stretch, we use grid size 30×30.

Training Details: We predict grid deformation on the 224×224 image plane. As the ground truth segmentation masks are provided, we append a one-hot mask to the RGB values when calculating the differentiable variance and reconstruction loss to better align the grid with semantic boundary. Following DELSE [11] and DEXTR [9], we initialize the image encoder using pretrained weights from COCO [7]. We simultaneously train all the modules using Adam [5]. Detailed hyperparameter settings are provided in the supplementary material. For grid-20, we use top2, for grid-30/40 we use top3 with $1e-4$ learning rate and $5e-4$ weight decay. The $\lambda_{\text{recons}}, \lambda_{\text{area}}, \lambda_{\text{lap}}$ are set to 0.5, 0.02, 0.02, respectively.

3.3 Learnable Downsampling

Experiment Details: For the proof of concept, we use a modified ResNet50, which is a more lightweight network compared to current SOTA models, to build the model shown in the main paper. The shallow CNN encoder consists of the conv1 layer and the first two bottlenecks in layer1. It is shared by the segmentation branch and grid deformation branch. The deep CNN after pooling consists of the other three ResNet conv blocks and one conv1x1 classifier. All downsampling modules are removed except for the layer2 block. The stride of conv1 is set to 1. The grid decoder consumes the shallow feature map and shares the same architecture as pervious experiments. We resize the full image to 512x1024 as input, and produce a shallow feature map at the same resolution. Each square on the deformed grid contains two triangle cells. To ensure fair comparison, each triangle cell corresponds to one pixel on the new feature map, e.g. a grid with 33x33 vertices will generate a feature map of 32x64.

The predicted semantic heatmaps are pasted back to original image coordinate and was trained with cross entropy loss. We use mean IoU over 19 classes and boundary F scores with threshold as 4 and 16 pixels to evaluate model performance. The model takes full image as input and produces a shallow feature map with resolution as 512x1024. The baseline methods are directly doing feature pooling on the shallow feature map, while our grid pooling methods operates on deformed triangle cells.

4 Additional Qualitative Results

4.1 Superpixel

We show additional quantitative results in Fig. 1, with comparisons presented in Fig. 2, 3, and 4 for pixel tolerance in BR and BP set to 1, 2, and 3 pixels, respectively. Comparing performance of the AffinityNet vs PSPNet, having a deeper network helps to learn the semantics of the boundaries. Comparing supervised and unsupervised settings, adding human annotation also helps learning the semantics.

4.2 Object Annotation

We present additional qualitative results of our deformable grid in Fig. 5, 6, 7, 10, and 9, showcasing the predicted grid deformation on both CityScapes and cross domain datasets (ADE, KITTI, Card.MR, SSTEM). For boundary-based segmentation method, we show additional qualitative results in Fig. 11, and 13. We show qualitative results for the pixel-based segmentation approach in Fig. 12.

4.3 Learnable Downsampling

We show qualitative comparison of the grid pooling methods and image coordinate pooling baselines with feature downsampling ratio as 1/8. The qualitative

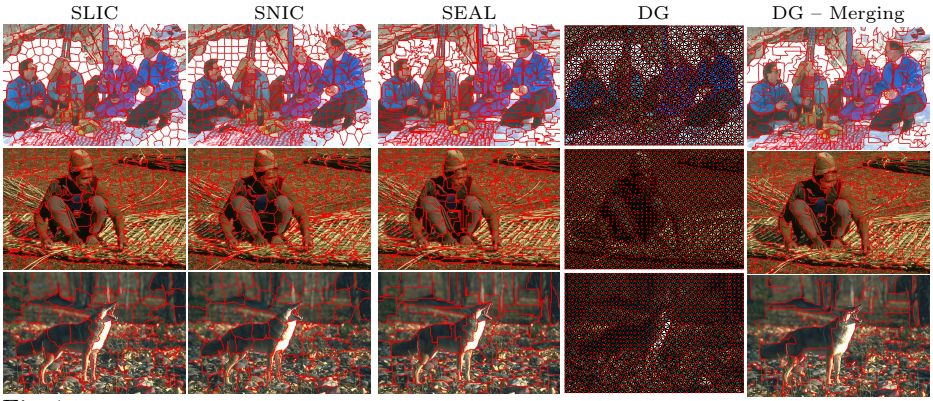

Fig. 1: **Superpixel Segmentation:** We compare our results to existing superpixel baselines. For our method we show the Deformable Grid (DG) using AffinityNet as the backbone, and results after clustering (right column). [Please zoom in]

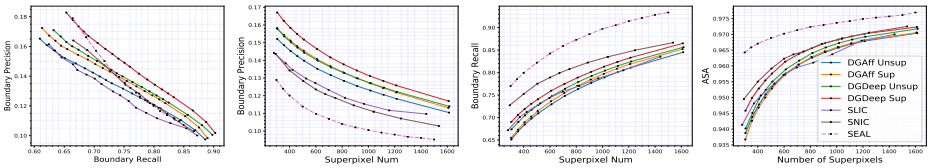

Fig. 2: **Superpixel Segmentation:** From left to right: BP-BR, BP, BR and ASA, The pixel tolerance is set to 1 pixel. DGAff denotes Deformable Grid (DG) with Affinity Net, and DGDeep denotes DG with Deeplab.

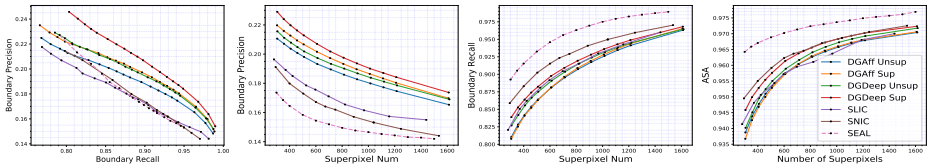

Fig. 3: **Superpixel Segmentation:** From left to right: BP-BR, BP, BR and ASA, The pixel tolerance is set to 2 pixel.

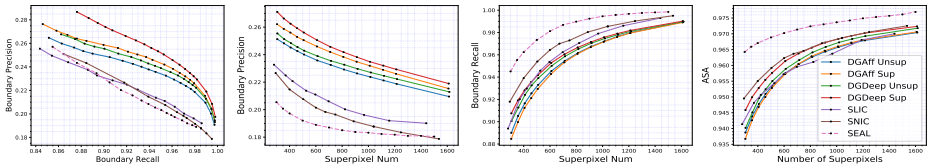

Fig. 4: **Superpixel Segmentation:** From left to right: BP-BR, BP, BR and ASA, The pixel tolerance is set to 3 pixels.

semantic segmentation results on Cityscapes are shown in Fig. 14, and 15. Compared to directly apply feature pooling in the image coordinates, grid pooling methods predict tighter boundary and are better at retrieving tiny instances,

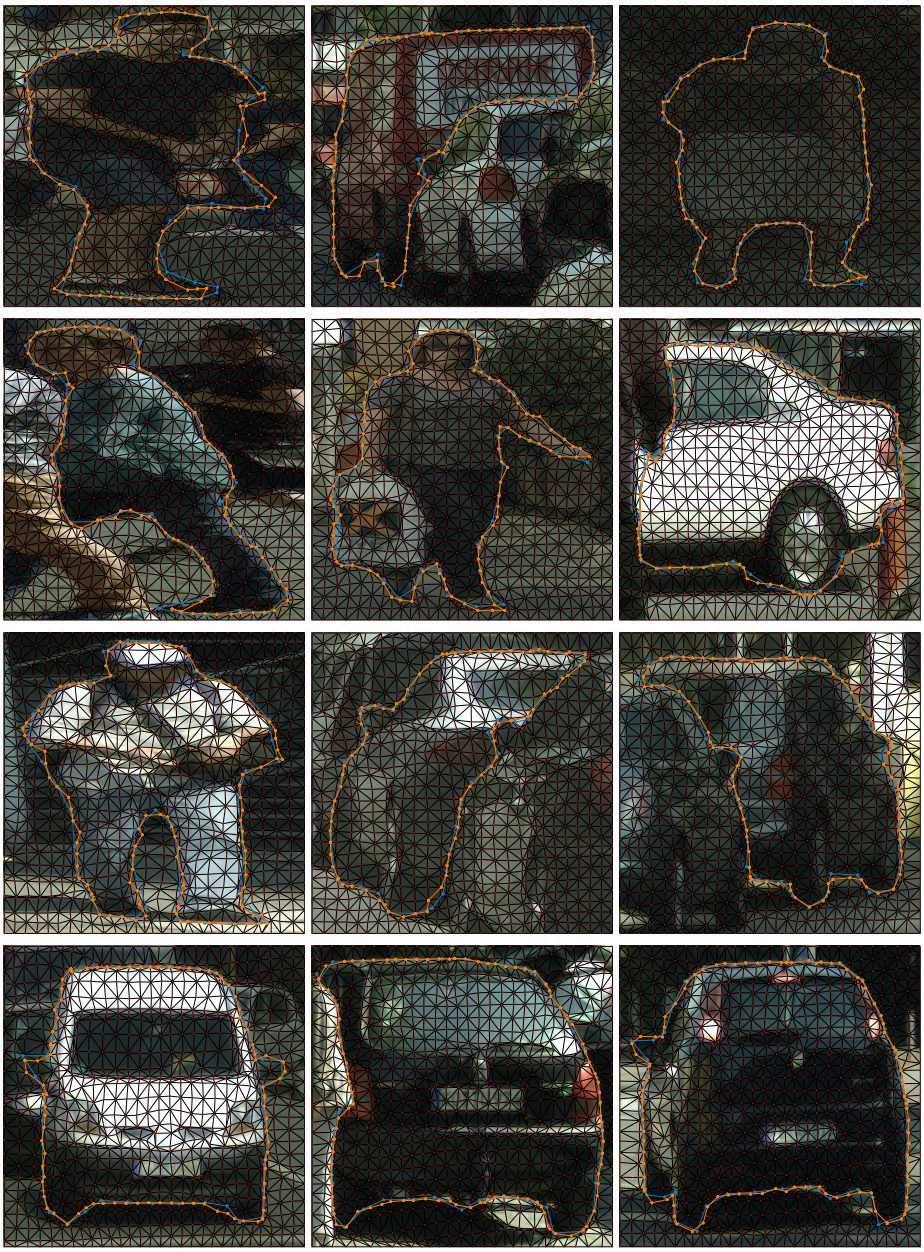

Fig. 5: **Deformed Grid:** We show examples of predicted grids on the Cityscapes dataset. Blue line is CurveGCN's prediction and orange line is searched minimal energy path.

while the baselines tend to predict over-smoothed boundaries. This shows that the geometry-aware property of deformable grid also benefits the feature space.

Fig. 6: **Deformed Grid:** We show examples of predicted grids on the ADE dataset. Blue line is CurveGCN's prediction and orange line is searched minimal energy path.

Fig. 7: **Deformed Grid:** We show examples of predicted grids on KITTI. Blue line is CurveGCN's prediction and orange line is searched minimal energy path.

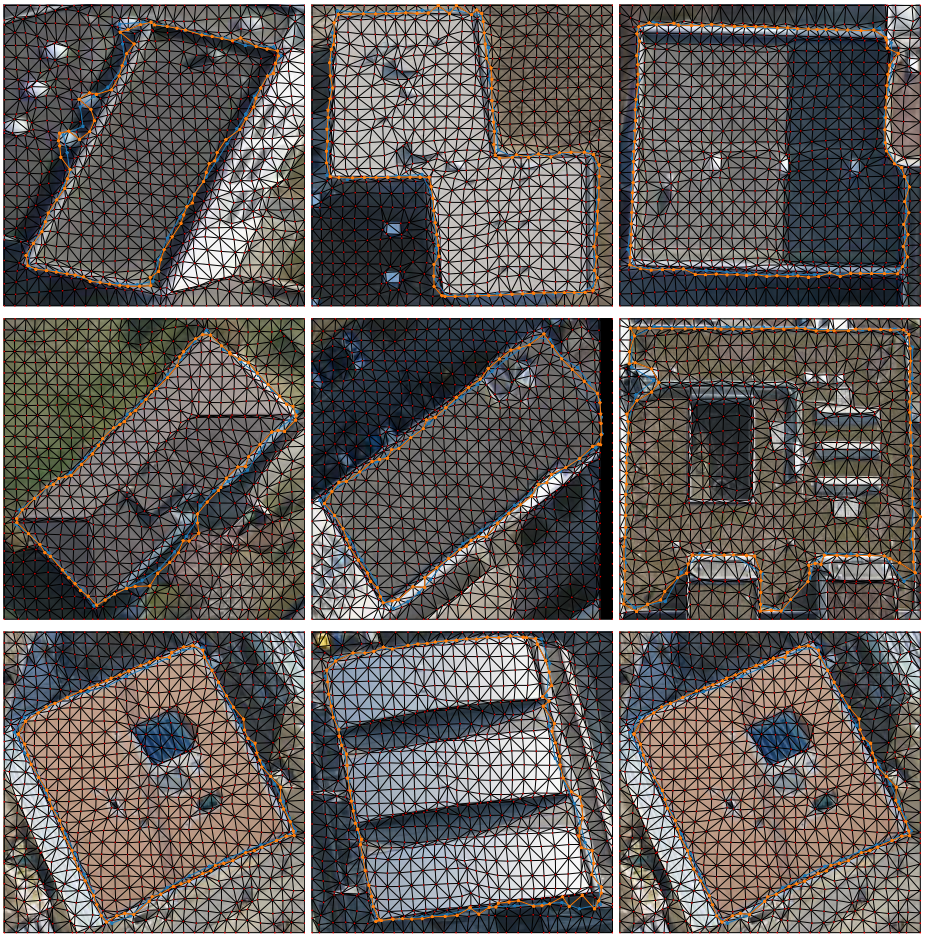

Fig. 8: **Deformed Grid:** We show examples of predicted grids on Rooftop. Blue line is CurveGCN's prediction and orange line is searched minimal energy path.

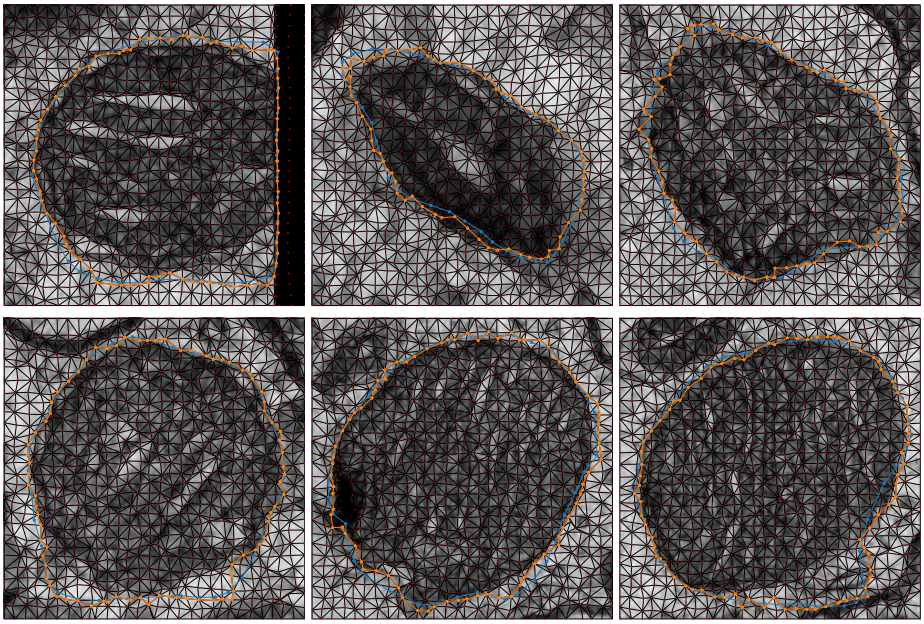

Fig. 9: **Deformed Grid:** We show examples of predicted grids on SSTEM. Blue line is CurveGCN's prediction and orange line is searched minimal energy path.

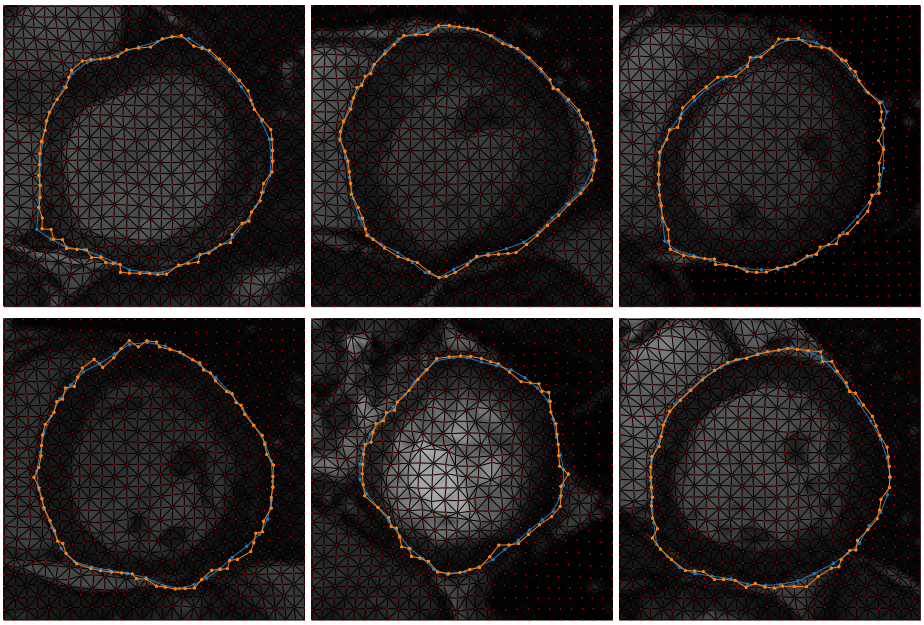

Fig. 10: **Deformed Grid:** We show examples of predicted grids on Card.MR. Blue line is CurveGCN's prediction and orange line is searched minimal energy path.

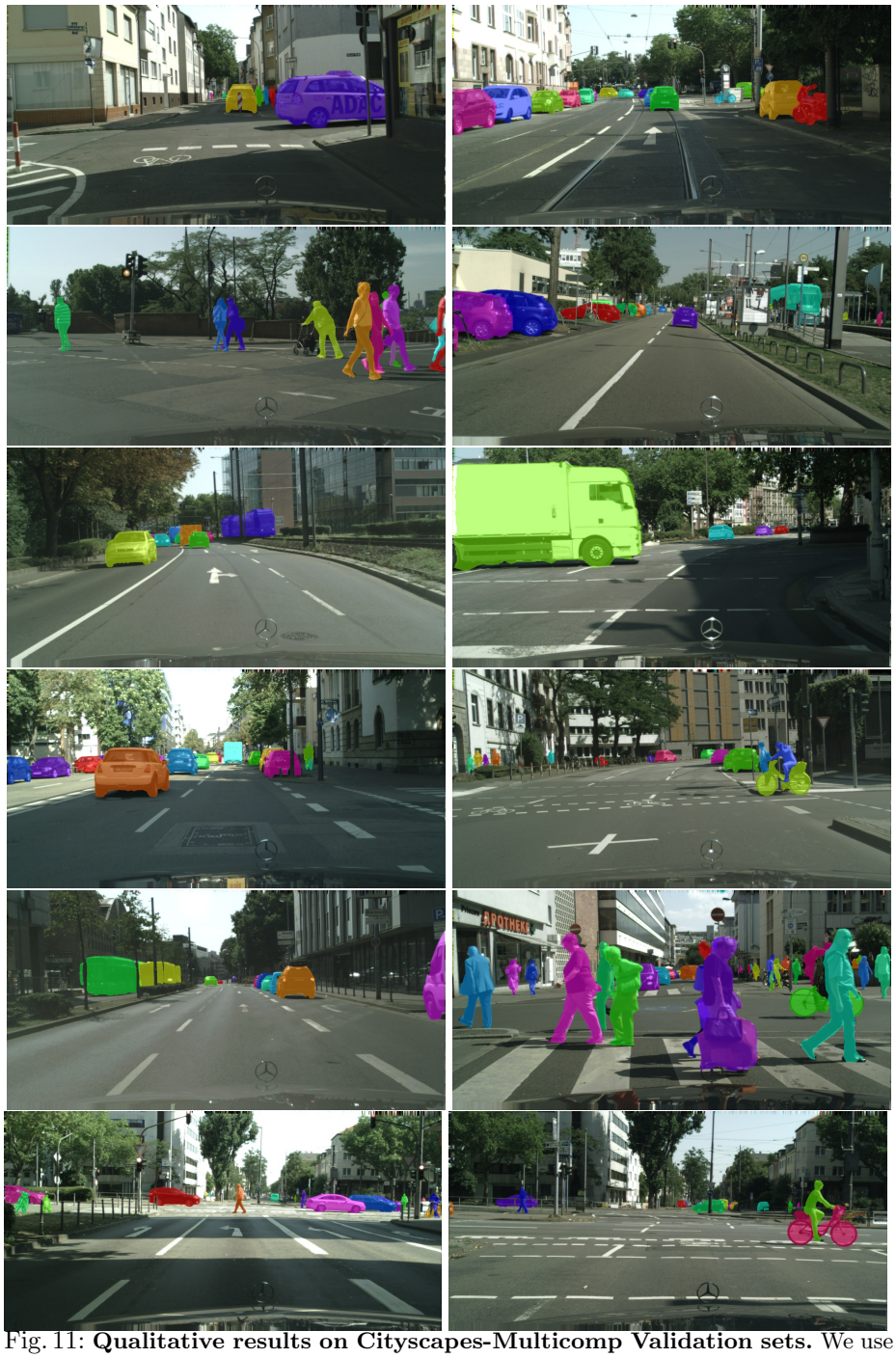

Fig. 11: **Qualitative results on Cityscapes-Multicomp Validation sets.** We use boundary-based deformable grid here.

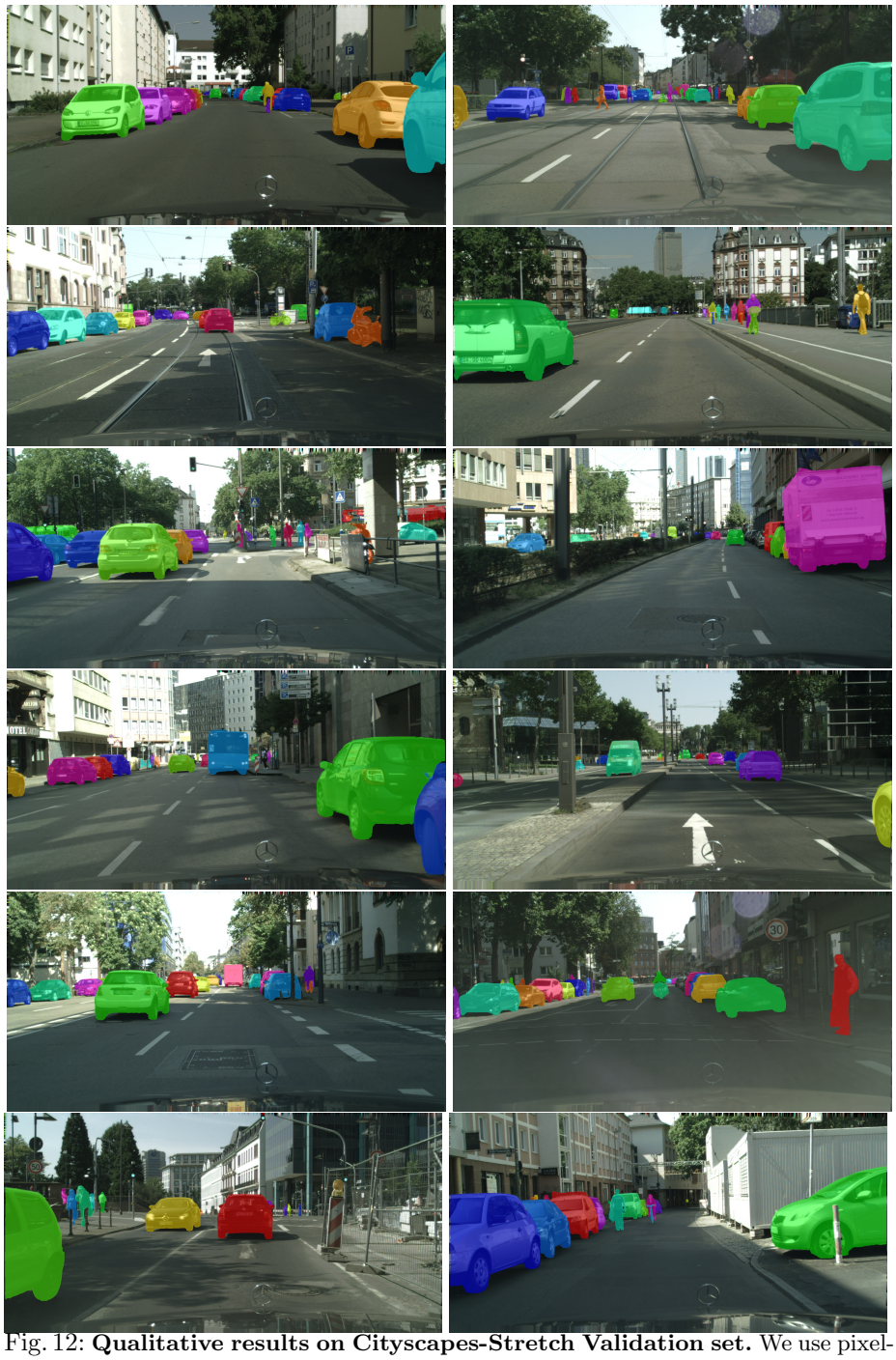

Fig. 12: **Qualitative results on Cityscapes-Stretch Validation set.** We use pixel-based deformable grid here.

Fig. 13: **Qualitative results on cross domain datasets:** From left to right column: Medical, KITTI, Rooftop, ADE. We use boundary-based deformable grid here.

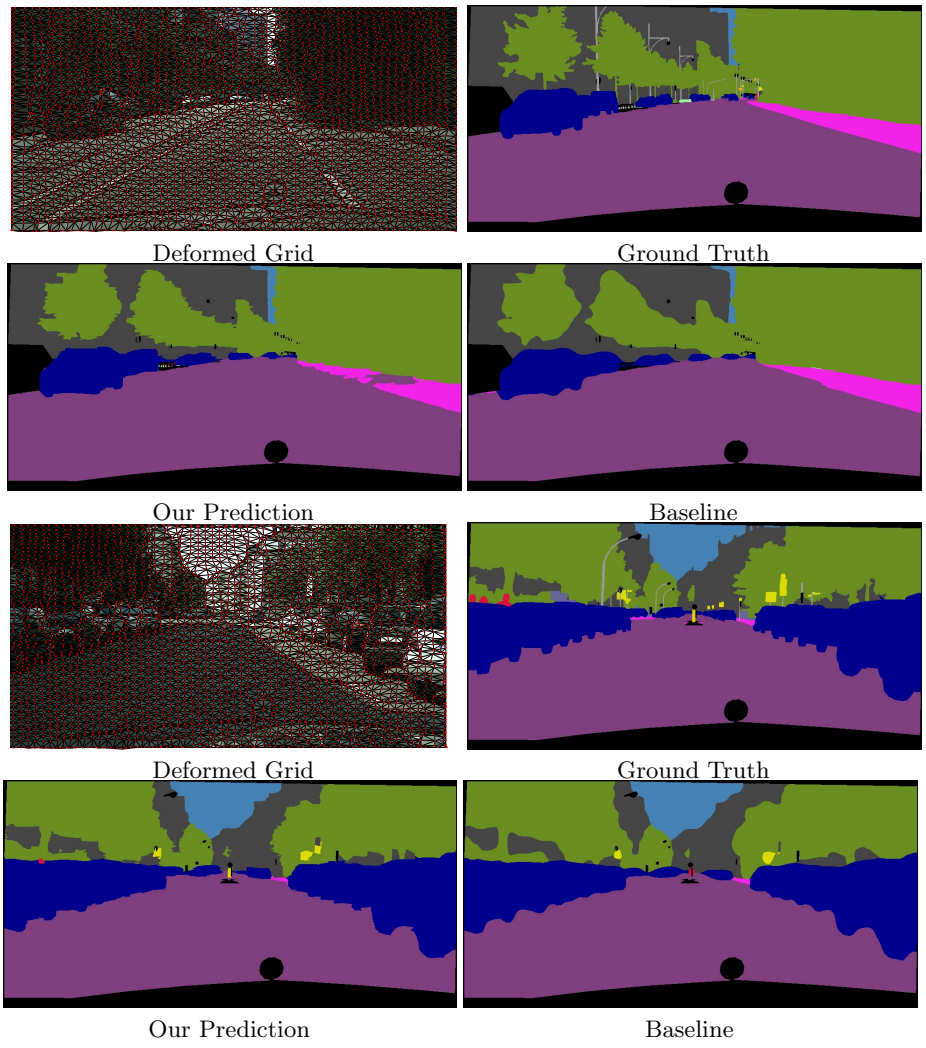

Fig. 14: Qualitative comparison of Grid Average Pooling and Average Pooling.

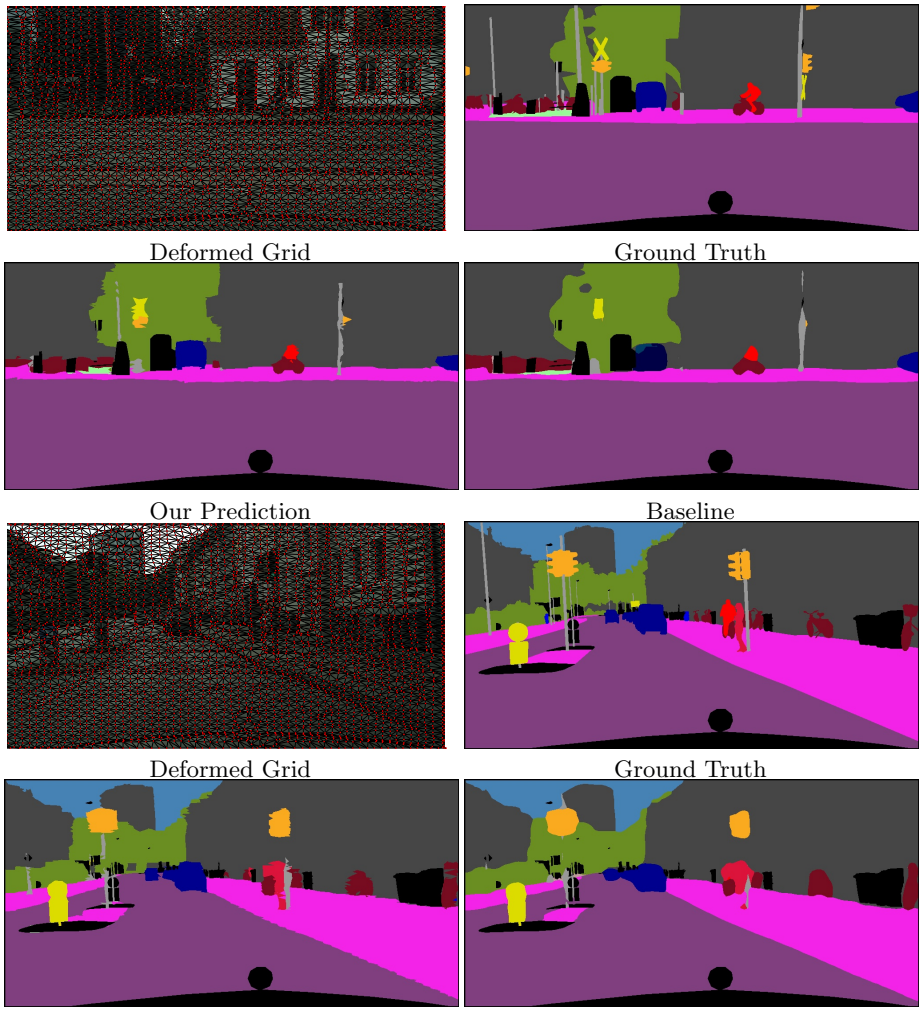

Fig. 15: Qualitative comparison of Grid Max Pooling and Max Pooling.

References

1. Arbelaez, P., Maire, M., Fowlkes, C., Malik, J.: Contour detection and hierarchical image segmentation. IEEE transactions on pattern analysis and machine intelligence **33**(5), 898–916 (2010)
2. Cordts, M., Omran, M., Ramos, S., Rehfeld, T., Enzweiler, M., Benenson, R., Franke, U., Roth, S., Schiele, B.: The cityscapes dataset for semantic urban scene understanding. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 3213–3223 (2016)
3. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 770–778 (2016)
4. Jampani, V., Sun, D., Liu, M.Y., Yang, M.H., Kautz, J.: Superpixel samping networks. In: ECCV (2018)
5. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
6. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. arXiv preprint arXiv:1609.02907 (2016)
7. Lin, T.Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L.: Microsoft coco: Common objects in context. In: ECCV (2014)
8. Ling, H., Gao, J., Kar, A., Chen, W., Fidler, S.: Fast interactive object annotation with curve-gcn. In: CVPR. pp. 5257–5266 (2019)
9. Maninis, K.K., Caelles, S., Pont-Tuset, J., Van Gool, L.: Deep extreme cut: From extreme points to object segmentation. In: CVPR (2018)
10. Tu, W.C., Liu, M.Y., Jampani, V., Sun, D., Chien, S.Y., Yang, M.H., Kautz, J.: Learning superpixels with segmentation-aware affinity loss. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 568–576 (2018)
11. Wang, Z., Acuna, D., Ling, H., Kar, A., Fidler, S.: Object instance annotation with deep extreme level set evolution. In: CVPR (2019)
12. Zhao, H., Shi, J., Qi, X., Wang, X., Jia, J.: Pyramid scene parsing network. In: CVPR (2017)