Supplementary Material for GraphCSPN: Geometry-Aware Depth Completion via Dynamic GCNs

1 Overview

In this supplementary material, we first present a video demonstration of our model on KITTI Depth Completion dataset [12]. An example of the video frame can be found in Figure 1. We also provide additional analysis on spatial propagation networks from the physical perspective to demonstrate the advantages of our model over previous methods [8, 3, 2, 9]. In addition, more experiments and additional visualization results on indoor and outdoor datasets are also provided in this material.

2 Anisotropic Diffusion Process

Spatial propagation networks originate in anisotropic diffusion which is a physical transport process and can be formulated by *Fick's law*:

$$j = -D \cdot \nabla u \tag{1}$$

This equation describes the equilibration process during which a concentration gradient ∇u causes a flux j to compensate for this gradient. And D denotes the diffusion tensor. The case where j and ∇u are not parallel is called anisotropic diffusion. By imposing continuity restrictions, the diffusion equation can be derived as follows:

$$\partial_t u = \mathrm{div}(D \cdot \nabla u) \tag{2}$$

The simple case where the diffusion tensor D is not changed during the diffusion process is called linear diffusion. Previous methods [8, 3, 2, 9] only models the linear diffusion process, and both the neighbors and affinity matrix are fixed during the spatial propagation. Linear diffusion process dislocates edges and blurs important features [13] which can easily lead to inaccurate predictions. Our graph convolution based spatial propagation network models the nonlinear diffusion process where the diffusion tensor is a function of the differential structure of the evolving image itself. And our method casts the diffusion process in a more realistic manner and can effectively avoid those shortcomings of previous approaches. Furthermore, the spatial propagation of our method is performed between geometrically relevant neighbors in 3D space rather than in 2D plane of previous methods, and can impede the propagation of errors during the iterative refinement, as described in the main paper.

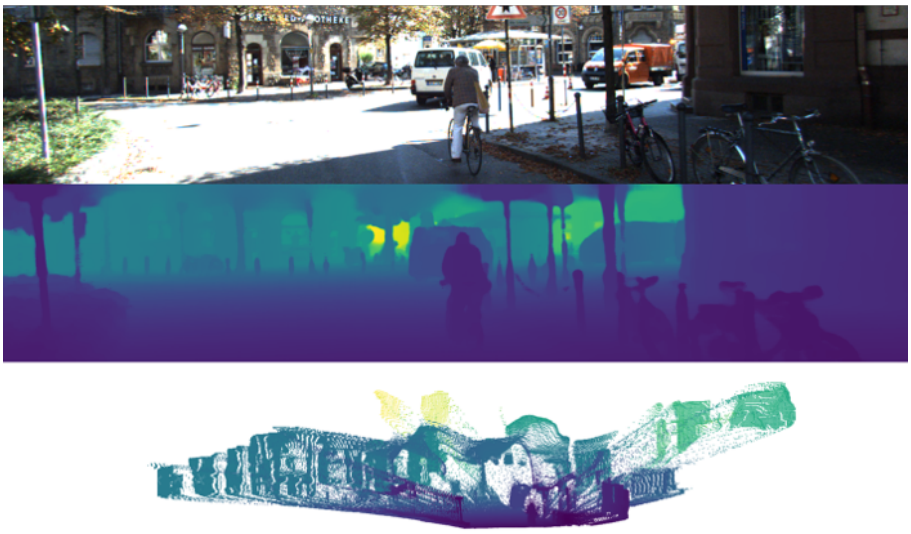


Fig. 1. An example frame of the video demonstration on KITTI Depth Completion dataset. From top to bottom, the frame consists of three parts, which are input RGB image, predicted dense depth map and the corresponding 3D reconstruction. The dense depth map and 3D reconstruction are generated using our method. (Best viewed in color.)

3 Implementation Details

The proposed model is implemented with the Pytorch framework [10] and trained on NVIDIA 3090 GPUs. During training, we adopt the ADAM optimizer [6] with $\beta_1 = 0.9$, $\beta_2 = 0.99$, and the initial learning rate is set to 0.001 which is reduced to half every 10 epochs. We train the model for 50 epochs and use a batch size of 32 and 16 for NYU-Depth-v2 dataset and KITTI dataset, respectively. The weights of the ResNet in the encoding layers are initialized with models pretrained on the ImageNet dataset [4].

4 Complexity Analysis

Since computation overhead is important for real-world applications, we make a comparison of model size and inference time for different methods:

method	CSPN	CSPN++	NLSPN	GraphCSPN
# params (M)	256.08	26.21	25.84	24.90
runtime (s)	1.00	0.20	0.23	0.14

Runtime is taken from the official KITTI benchmark website. Our model can attain high performance with light-weight backbone which largely reduces the model size. As for the propagation part, the computation complexity is irrelevant

Variants	Time (ms)	RMSE ↓	REL ↓	$\delta_{1.25}$ ↑	$\delta_{1.25^2}$ ↑	$\delta_{1.25^3}$ ↑
pixel	103.4	0.114	0.018	99.2	99.8	99.9
patch (2,2)	40.2	0.098	0.015	99.4	99.8	99.9
patch (4,4)	12.5	0.090	0.012	99.6	99.9	100.0
patch (2,8)	14.8	0.095	0.013	99.5	99.9	100.0
patch (8,8)	7.1	0.105	0.016	99.3	99.8	99.9
dense	20.6	0.102	0.016	99.4	99.9	100.0

Table 1. Quantitative evaluation on the NYU-Depth-v2 dataset with different choices of graph construction.

to the number of neighbors, but grows linearly with the number of nodes and feature dimensions. And our patch-wise graph construction greatly decreases the size of graph, thus reducing the overall computation overhead.

5 More Ablation Studies

5.1 Graph Construction

As stated in the main paper, the core of graph construction is to convert affinity maps $\mathbf{A} \in \mathbb{R}^{H\times W\times C}$ into a sequence of features $\mathbf{A}^{seq} \in \mathbb{R}^{N\times L}$. To do so, we first convert $\mathbf{A}$ into a sequence of patches $\mathbf{A}^{patch} \in \mathbb{R}^{N\times P_h\times P_w}$. And (P_h, P_w) is set to $(4, 4)$ in the experiments. In this ablation study, we first change the size of the patch and see how it would affect the performance and inference time of our model. The results are shown in Table 1. When we set $P_h = P_w = 1$, the graph is pixel-wise constructed and the total number of nodes equals to $H\times W$. In this case, the inference process is much slower, because the computation complexity of graph propagation increases quadratically with the number of nodes N. When we increase the patch size, the inference time decreases, and the performance first gets better and then becomes worse if the patch size is too large. By our patch-wise construction, prior knowledge about local correlations can be established, so the performance is improved. However, it fails to capture details of local structures with large patch size, thus leading to poor performance. We also change the shape of the patch to rectangle and find that there is a small decrease of performance. So the default patch size is set to $(4, 4)$. In addition, we study how the performance would change if we construct a dense graph like a Transformer [5], where the neighbors of each node are all the other nodes. In this way, the densely connected graph makes the propagation inefficient, because most of the neighbors are geometrically irrelevant. And as a result, the performance drops as can be seen in Table 1.

5.2 Propagation Steps

In the main paper, experiments are conducted to study the impact of different number of steps on spatial propagation. In this supplementary material, we

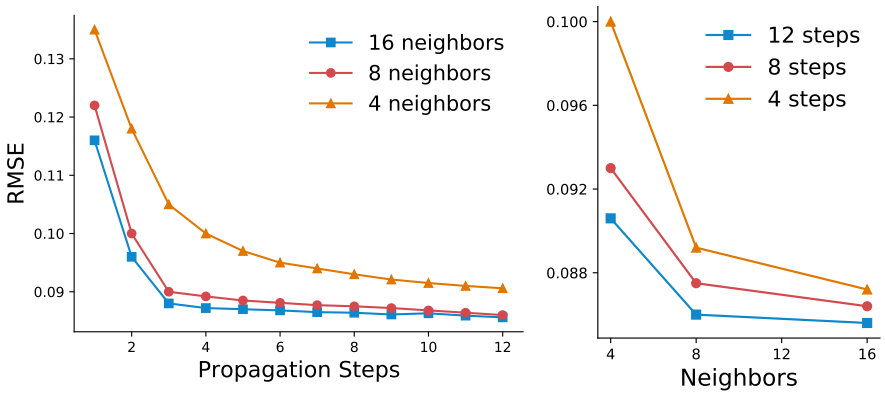
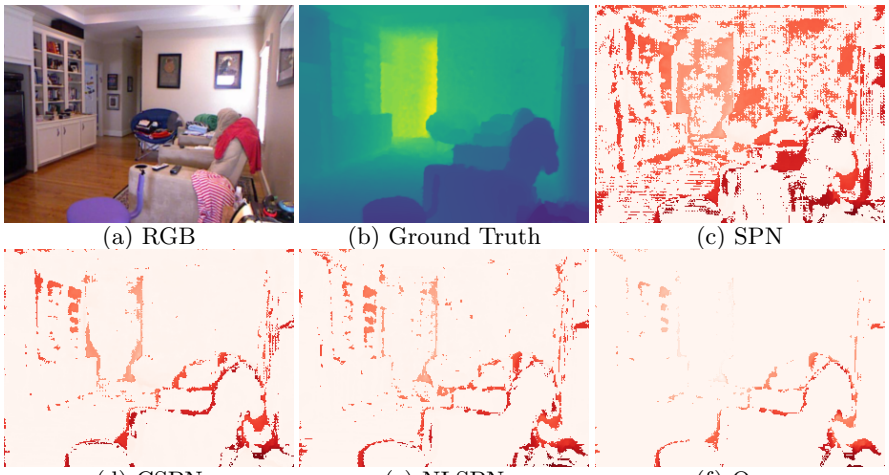


Fig. 2. Impact of different number of propagation steps and neighbors on the prediction accuracy on NYU dataset.

want to explore the full capacity of our model by applying more propagation steps. However, unlike CNNs, making GCNs deep is not trivial and there are inherent over-fitting and over-smoothing problems with deep GCNs [1]. In order to construct a deeper GCN, we use the methods proposed in [7] and exploit residual connections and dilated convolutions to alleviate those inherent problems. As can be seen in Figure 2, we find the model can converge quickly with a small number of steps and neighbors, which validates our proposed framework is propagation-efficient. Besides, the performance can be steadily improved with increasing number of steps and neighbors, which offers flexible choices given different requirement of performance and budget of computation resources in real-world applications.

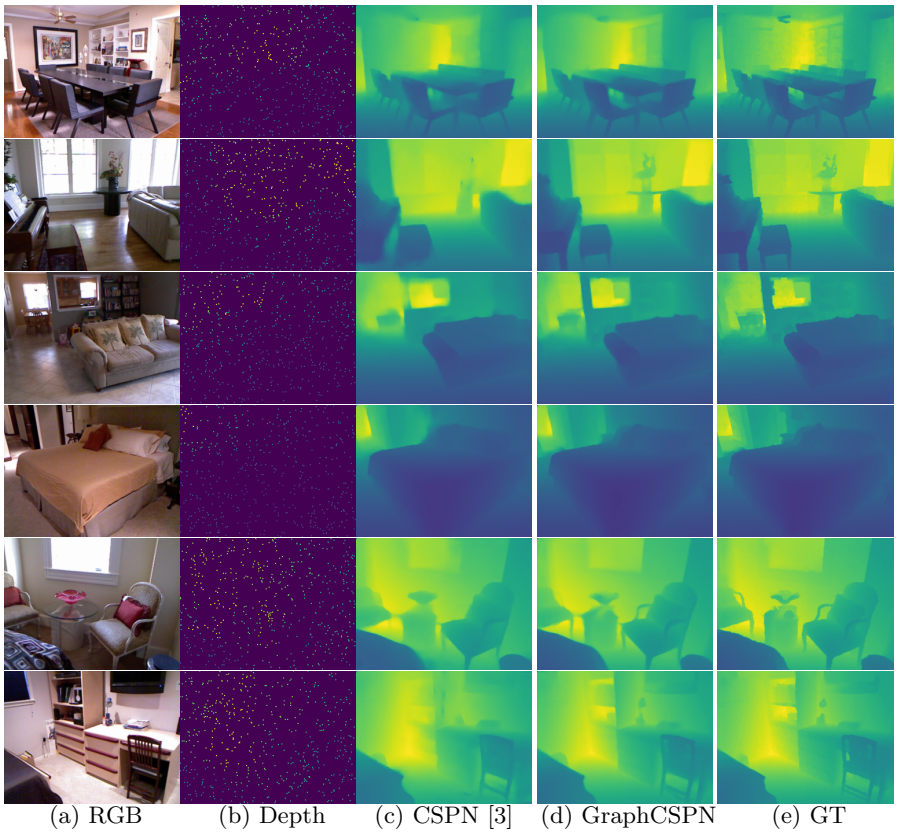
6 Additional Visualization Results



(a) RGB (b) Ground Truth (c) SPN

(d) CSPN (e) NLSPN (f) Ours

Fig. 3. Comparison of different methods using per-pixel error maps. A higher error is visualized with a darker red color.



(a) RGB (b) Depth (c) CSPN [3] (d) GraphCSPN (e) GT

Fig. 4. Depth completion results on the NYU-Depth-v2 dataset [11].

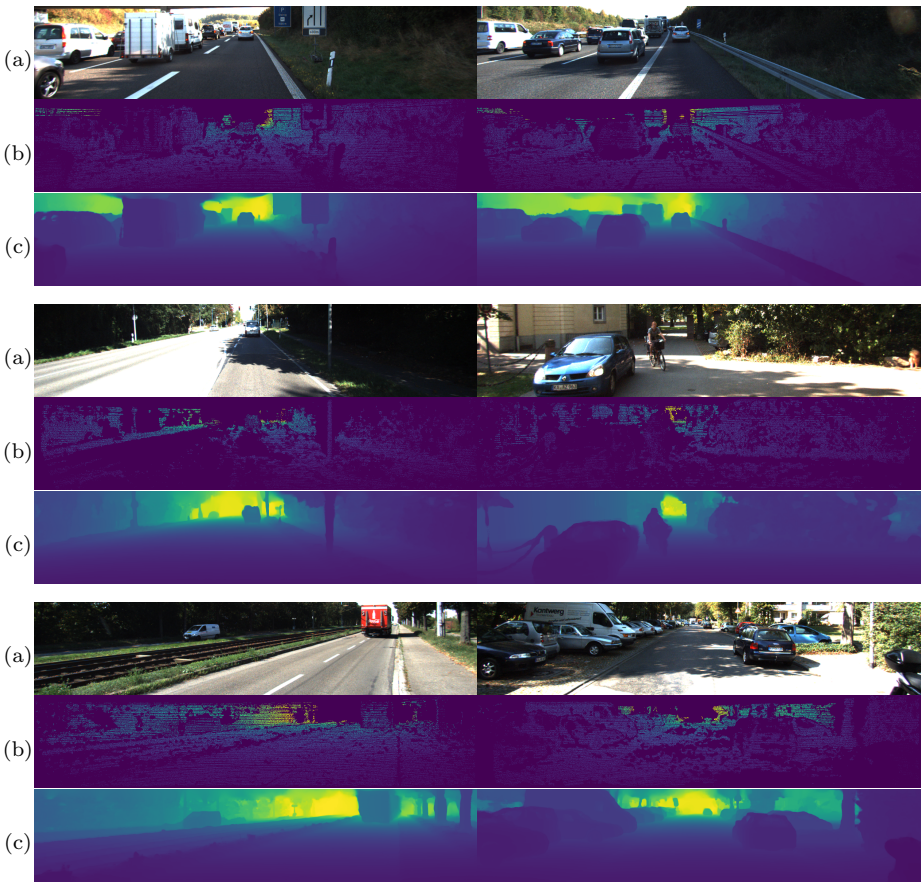


Fig. 5. Depth completion results on the KITTI Depth Completion dataset [12]. (a) RGB, (b) Ground truth, (c) GraphCSPN.

References

1. Chen, D., Lin, Y., Li, W., Li, P., Zhou, J., Sun, X.: Measuring and relieving the over-smoothing problem for graph neural networks from the topological view. In: Proceedings of the AAAI Conference on Artificial Intelligence. pp. 3438–3445 (2020)
2. Cheng, X., Wang, P., Guan, C., Yang, R.: Cspn++: Learning context and resource aware convolutional spatial propagation networks for depth completion. In: Proceedings of the AAAI Conference on Artificial Intelligence. pp. 10615–10622 (2020)
3. Cheng, X., Wang, P., Yang, R.: Depth estimation via affinity learned with convolutional spatial propagation network. In: Proceedings of the European Conference on Computer Vision (ECCV). pp. 103–119 (2018)
4. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet: A large-scale hierarchical image database. In: 2009 IEEE conference on computer vision and pattern recognition. pp. 248–255. Ieee (2009)
5. Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., et al.: An image is worth 16x16 words: Transformers for image recognition at scale. arXiv preprint arXiv:2010.11929 (2020)
6. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
7. Li, G., Muller, M., Thabet, A., Ghanem, B.: Deepgcns: Can gcns go as deep as cnns? In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 9267–9276 (2019)
8. Liu, S., De Mello, S., Gu, J., Zhong, G., Yang, M.H., Kautz, J.: Learning affinity via spatial propagation networks. arXiv preprint arXiv:1710.01020 (2017)
9. Park, J., Joo, K., Hu, Z., Liu, C.K., Kweon, I.S.: Non-local spatial propagation network for depth completion. arXiv preprint arXiv:2007.10042 **3**(8) (2020)
10. Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., et al.: Pytorch: An imperative style, high-performance deep learning library. arXiv preprint arXiv:1912.01703 (2019)
11. Silberman, N., Hoiem, D., Kohli, P., Fergus, R.: Indoor segmentation and support inference from rgbd images. In: European conference on computer vision. pp. 746–760. Springer (2012)
12. Uhrig, J., Schneider, N., Schneider, L., Franke, U., Brox, T., Geiger, A.: Sparsity invariant cnns. In: 2017 international conference on 3D Vision (3DV). pp. 11–20. IEEE (2017)
13. Weickert, J.: Anisotropic diffusion in image processing, vol. 1. Teubner Stuttgart (1998)