GraphFit: Learning Multi-scale Graph-Convolutional Representation for Point Cloud Normal Estimation

Keqiang Li$^{\star 1,3}$, Mingyang Zhao$^{\star 1,2}$, Huaiyu Wu $^{\star\star 1}$, Dong-Ming Yan^{1,3},
Zhen Shen^{1}, Fei-Yue Wang^{1}, and Gang Xiong^{1}

[1] SKLMCCS/NLPR, Institute of Automation, Chinese Academy of Sciences
[2] Beijing Academy of Artificial Intelligence
[3] School of Artificial Intelligence, University of Chinese Academy of Sciences
{likeqiang2020, huaiyu.wu, zhen.shen, feiyue.wang, gang.xiong }@ia.ac.cn
myzhao@baai.ac.cn, yandongming@gmail.com

In this document, we provide *additional details*, *experiments* and *applications* to support the original paper. Below is a summary of the contents:

- A detailed description of the adopted datasets and training settings, as well as the architecture of the proposed network are provided.
- We demonstrate the practical applications of our method for *point cloud denoising* and *3D surface reconstruction* based on normal estimation.
- More qualitative results, including angle RMSE and PGP5/10, are reported.

1 Implementation Details

We use the benchmark dataset PCPNet [1] to train the proposed network and test it. The training set contains eight shapes: four CAD objects (boxunion, cup, fandisk and flower) and four high-quality scans of figurines (armadillo, bunny, dragon and turtle). All shapes are modeled as triangular meshes and densely sampled with 100 k points. To augment the training set, we add Gaussian noise with zero mean and varying standard deviation $\sigma \in \{0.012, 0.006, 0.00125\}$, with respect to the diagonal length of the bounding box for each model. Then we attain 32 point clouds for training. The test set has 19 shapes, including CAD objects, figurines, and analytic shapes.

The proposed network is trained and implemented by the Pytorch framework on a Nvidia Tesla v100 GPU, using the Adam optimizer[4]. The batch size and initial learning rate are equal to 256 and 1e-3, respectively. We train the network 600 epochs in total, and the learning rate decays to 10% of the initial value at epoch 200 and 500. The overall training loss is defined as

$$\mathcal{L}_{\text{tol}} = |\mathbf{n}_{gt} \times \hat{\mathbf{n}}| + \mathcal{L}_{\text{con}} \ + \lambda_3 \mathcal{L}_{\text{reg1}} + \lambda_4 \mathcal{L}_{\text{reg2}}, \tag{1}$$

* The authors contribute equally in this work.

** Corresponding author.

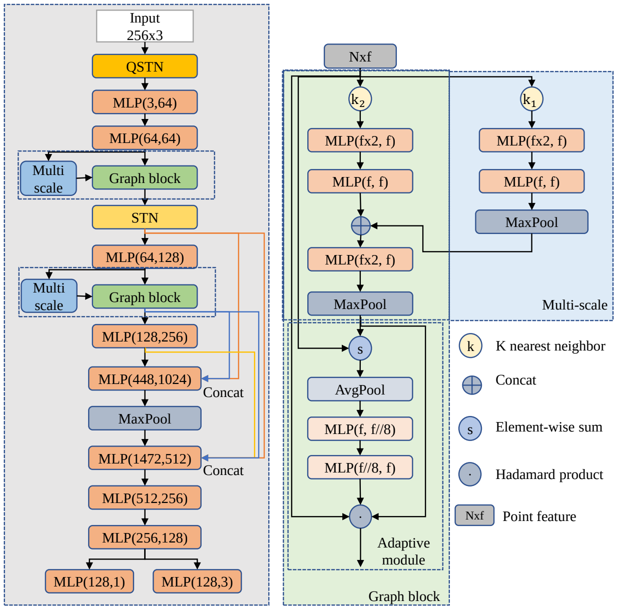


Fig. 1: Architecture details of our proposed network GraphFit.

$$\mathcal{L}_{\text{con}} = \frac{1}{N_{p_i}} \left[-\lambda_1 \sum_{j=1}^{N_{p_i}} \log\left(w_j\right) + \lambda_2 \sum_{j=1}^{N_{p_i}} w_j \left|\mathbf{n}_{gt,j} \times \hat{\mathbf{n}}_j\right| \right], \tag{2}$$

$$\mathcal{L}_{\text{reg1}} = \left|I - A_1 A_1^T\right|, \tag{3}$$

$$\mathcal{L}_{\text{reg2}} = \left|I - A_2 A_2^T\right|, \tag{4}$$

where we use the default settings $\lambda_1 = 0.05$, $\lambda_2 = 0.25$, $\lambda_3 = 0.1$, and $\lambda_4 = 0.01$.

2 Details of Our Proposed Network Architecture

Fig. 1 presents the details of our proposed network architecture. An MLP unit consists of a 1×1 Conv, a BatchNorm layer and a ReLU in the left of Fig. 1. For MLP in the Graph block and the multi-scale layer, we use the Leaky ReLU (LReLU) as its activation function.

3 Results on the SceneNN Dataset

The $PGP\alpha$ results are shown in Fig. 2, our method significantly outperforms all competitors and achieves the state-of-the-art performance.

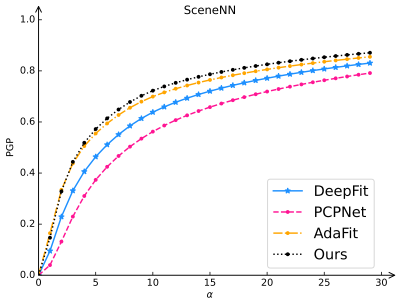


Fig. 2: Comparison of $PGP\alpha$ for unoriented normal estimation on the SceneNN dataset [2].

Table 1: Investigation on the influence of different number of graph blocks on normal estimation.

Aug.	1 Graph-block	2 Graph-blocks	3 Graph-blocks
No Noise	4.90	4.56	4.43
σ = 0.125%	9.06	8.87	8.88
σ = 0.6%	16.58	16.57	16.53
σ = 1.2%	22.87	22.77	22.74
Gradient	5.52	5.33	5.22
Strip	5.61	5.38	5.31
Average	10.76	10.58	10.52

4 Ablation Study of Different Graph Blocks

To demonstrate the effectiveness of different graph blocks in the proposed network, we further implement ablation study and report the average RMSE in Tab. 1, in which the neighborhood size and the jet order are equal to 256 and 3, respectively. As can be seen, the network achieves higher accuracy for both noise and varying point density by adding graph blocks, where the average RMSE decreases from 10.76 to 10.52. Nevertheless, with the number of graph blocks increasing, such as from two to three, the angle RMSE of unoriented normal vectors does not reduce significantly, thereby we adopt one graph block for use.

5 Practical Applications

We also deploy the proposed normal estimation method for point cloud denoising and 3D surface reconstruction.

Point cloud denoising. It is an important task in the 3D vision field to remove noise disturbance for point clouds. We combine the proposed normal estimation

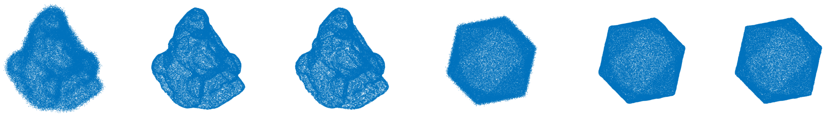

(a) Input (b) Ours (c) GT (d) Input (e) Ours (f) GT

Fig. 3: Point cloud denoising using our proposed normal estimator.

Table 2: Comparison of the surface reconstruction precision (RMSE) of our proposed method and AdaFit, DeepFit, PCPNet.

	Ours	AdaFit	DeepFit	PCPNet
Liberty	**0.00039**	0.00064	0.00084	0.00165
Column	**0.00592**	0.00885	0.01077	0.01667
Netsuke	**0.00495**	0.00501	0.00993	0.01573
Average	**0.00375**	0.00483	0.00718	0.01135

method with the *modified edge recovery algorithm* in [5] to update the point positions $\mathbf{p}_i$. The new position $\mathbf{p}'_i$ is calculated by

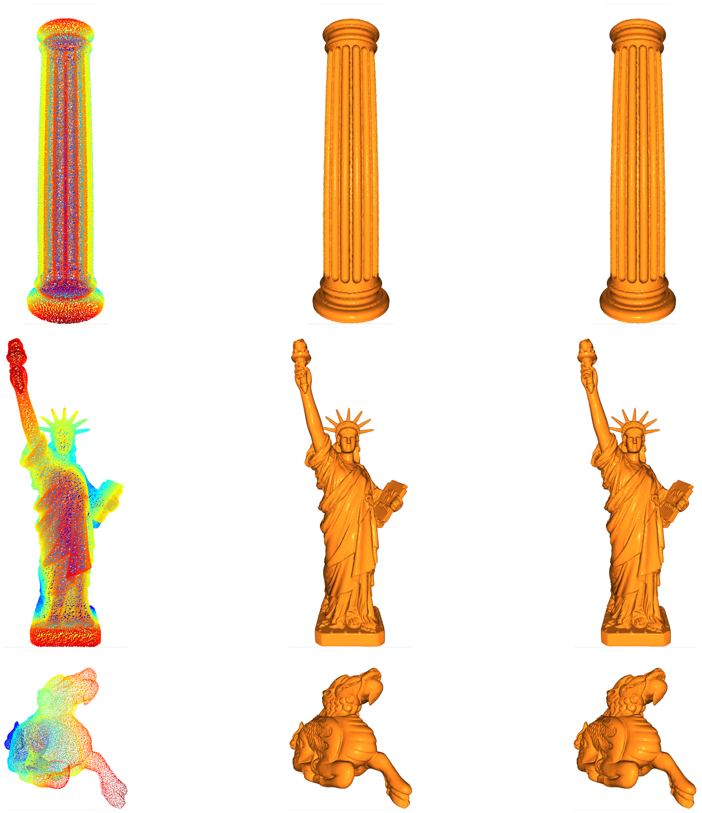
$$\mathbf{p}'_i = \mathbf{p}_i + \gamma_i \sum_{j \in \mathcal{N}_i} \left(\mathbf{p}_j - \mathbf{p}_i\right) \left(\mathbf{n}_i^T \mathbf{n}_i + \mathbf{n}_j^T \mathbf{n}_j\right) \tag{5}$$

where $\mathcal{N}_i$ are the neighboring points of $\mathbf{p}_i$, $\mathbf{n}_i, \mathbf{n}_j$ are the estimated normals. Fig. 3 shows sample denosing results. As observed, our method attains highly promising denosing performance compared with the ground truths, which are recovered via the ground truth normals.

3D surface reconstruction. One common application of normal estimation for point clouds is to reconstruct the potential surface. We apply the proposed normal estimation method and adopt *Poisson reconstruction* [3] for this purpose. Tab. 2 reports the RMSE of all compared approaches in the sense of *Hausdorff distance*. Results demonstrate that our method achieves high-quality reconstruction on all test cases. It is more accurate than baseline competitors. We present several reconstructed surface in Fig. 4.

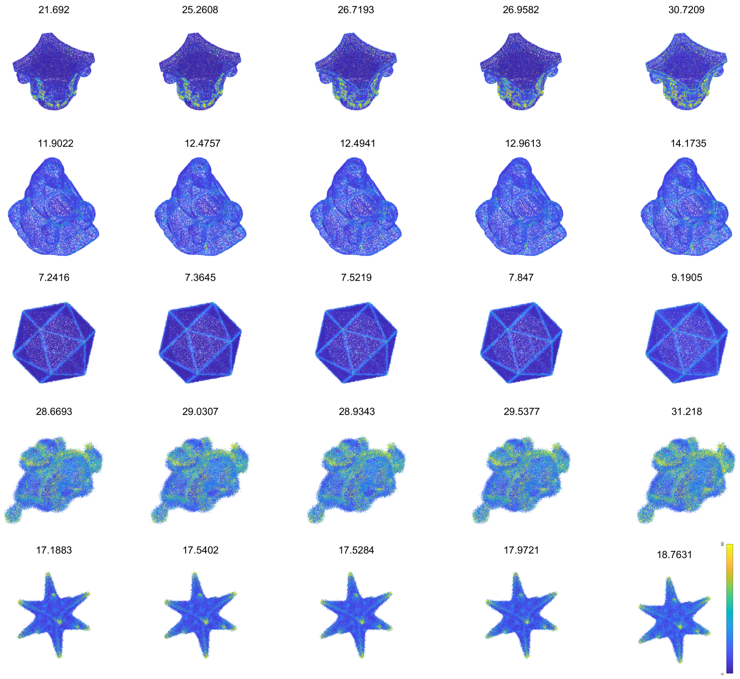
6 More Qualitative Results

We present more test results in Fig. 5 and Fig. 6 to demonstrate the better performance of our proposed normal estimator.



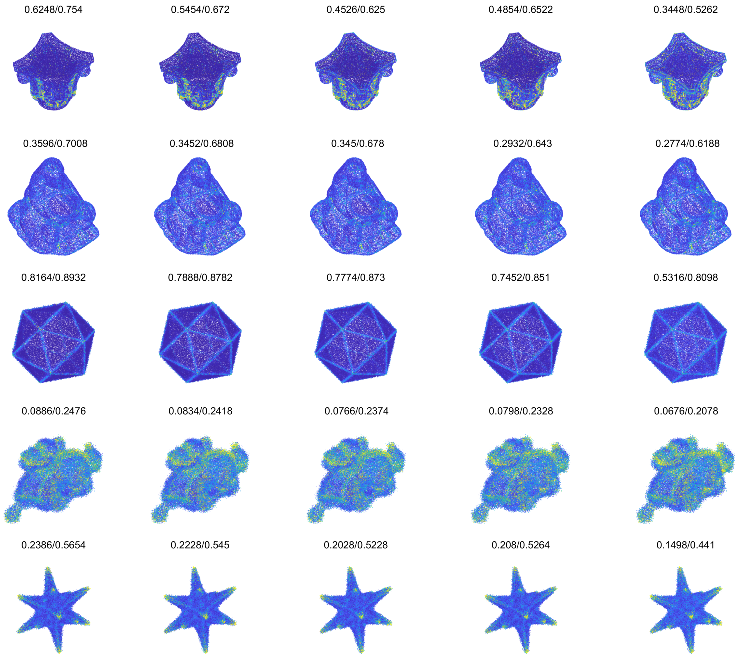
(a) Input (b) Ours (c) Groundtruth

Fig. 4: Poisson surface reconstruction via the proposed normal estimator.



(a) Ours (b) AdaFit (c) DeepFit (d) NestiNet (e) PCPNet

Fig. 5: Illustration of the normal estimation errors. The errors are mapped to a heatmap ranging from 0° to 60°. Values above the models are the corresponding RMSE. Our method achieves higher accuracy.



(a) Ours (b) AdaFit (c) DeepFit (d) NestiNet (e) PCPNet

Fig. 6: Qualitative results with respect to PGP(5) and PGP(10). Values above the models are the corresponding PGP5/10 errors. Our method attains more accurate normal estimation.

References

1. Guerrero, P., Kleiman, Y., Ovsjanikov, M., Mitra, N.J.: Pcpnet learning local shape properties from raw point clouds. Computer Graphics Forum. **37**(2), 75–85 (2018)
2. Hua, B.S., Tran, M.K., Yeung, S.K.: Pointwise convolutional neural networks. In: Proceedings of the IEEE/CVF conference on Computer Vision and Pattern Recognition. pp. 984–993 (2018)
3. Kazhdan, M., Bolitho, M., Hoppe, H.: Poisson surface reconstruction. In: Proceedings of the 4th Eurographics Symposium on Geometry Processing. vol. 7 (2006)
4. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. In: Proceedings of the International Conference on Learning Representations. (2015)
5. Lu, X., Schaefer, S., Luo, J., Ma, L., He, Y.: Low rank matrix approximation for 3D geometry filtering. IEEE Transactions on Visualization and Computer Graphics. **28**(04), 1835–1847 (2022)