Supplemental Material for Geometry and Learning Co-supported Normal Estimation for Unstructured Point Cloud

1. Implemental details for MFPS

In this section, we explain a detailed implementation for our multi-scale fitting patch selection.

First, for each given patch Q_j^t with K_t neighboring points located at p_j, three random non-collinear points are chosen to form a candidate plane. This process is repeated M times to generate several candidate planes. We set $M = 150$ empirically, to ensure that there is at least one plane that can describe the local structure. For each candidate plane θ, $E_{Q_j^t}(\theta)$ is computed via Eq. 1 of the submitted manuscript, and the one with maximum value is regarded as the fitting plane $\theta^*_{Q_j^t}$ of Q_j^t.

Having generated all the fitting planes, for each candidate point p_i, we first collect all the candidate patches that contain p_i as set $S_i = \{Q_j^t | p_i \in Q_j^t\}$, and compute its $\mathcal{D}_i$ via Eq. 4 of the submitted manuscript. The patch maximizing $\mathcal{D}_i$ is the selected fitting patch, and the normal of the fitting plane θ^* of this patch is used as the suboptimal normal of p_i. Note that, for a point on the smooth regions, its suboptimal normal is computed simply using PCA, not the above process for more efficiency.

2. Cluster-based approach

Before training, we partition the trainset into K_c clusters via the k-means algorithm. It is performed on the normal components $(\{N_f^i\}_{i=1}^n)$ of MPD to cluster similar samples together. In each cluster, we train a separate NH-Net to map an MPD to its ground-truth counterpart. The network can then focus on a specific kind of geometric features and can intensively refine this kind of normals. In runtime test stage, a single MPD is distributed into one of these clusters if $||N_f^i - C_l|| \leq ||N_f^i - C_k||, \forall k$. Here, C_k is the clustering center of the k-th cluster.

Furthermore, we evaluate the performance of different K_c on our synthetic benchmark dataset, as shown in Fig. 2. We set $K_c = 4$ by default in all our experiment, unless stated otherwise. A larger K_c improves the results slightly but heavily expands the network.

Scheme 1	Scheme 2
$1 \times 7 \times 7$ (input)	$9 \times 7 \times 7$ (input)
Conv(3, 3, 64, P=1) + Relu	Conv(3, 3, 64, P=1) + Relu
Conv(3, 3, 128, P=1) + Relu	Conv(3, 3, 128, P=1) + Relu
Maxpool(3, 3, S=1)	Maxpool(3, 3, S=1)
Conv(3, 3, 64, P=1) + Relu	Conv(3, 3, 64, P=1) + Relu
Maxpool(3, 3, S=1)	Maxpool(3, 3, S=1)
Conv(3, 3, 1, P=1) + Tanh	Conv(3, 3, 1, P=1) + Tanh

Scheme 3
27×1 (input)
FC(20) + Relu
FC(3)

Table 1. Architecture details of ablations.

3. Experiments on different learning schemes

We give a detailed look at the ablations of our architecture and results in Sec. 6.4.

- Scheme 1 - a network based on a single HMP, which feeds one HMP into a CNN architecture and outputs a 3×3 matrix. The corresponding normal is transformed by the output matrix to yield the final normal. We train this network on the suboptimal normals and HMPs constructed on them. See Tab. 1.

- Scheme 2 - an HMP-based network which receives total 9 HMPs in an MPD as input, outputting one 3×3 matrix. Without a gathering module, the yielded matrix is applied on the suboptimal normal in the MPD and outputs the final normal. See Tab. 1.

- Scheme 3 - a single-layer fully connected network without the HMP module. This network receives 9 normals in the MPD and outputs the final normal. See Tab. 1.

We train all the above ablations using the same dataset as our NH-Net. The complete benchmark results are reported in Fig. 1. We can see that by the combination of HMP-based module and gathering module, our proposed scheme outperforms the separate versions, especially for the big noise data.

Model	PCA [5]	HF [2]	HoughCNN [3]	PCPNet [4]	PCV [6]	Nesti-Net [1]	Ours
Merlion	5.6°	6.6°	4.9°	6.1°	4.6°	4.5°	**3.9**°
Anchor	9.1°	13.6°	11.3°	10.3°	6.0°	5.8°	**5.0**°
Block	18.7°	9.4°	9.2°	18.7°	6.5°	6.8°	**4.5**°
Joint	9.2°	4.2°	4.3°	11.3°	3.4°	2.8°	**2.4**°
Octahedron	7.7°	15.8°	10.5°	8.5°	6.1°	6.5°	**4.1**°
Sharpsphere	8.9°	6.6°	7.5°	8.2°	**6.2**°	7.1°	6.5°
Kitten	**2.4**°	2.5°	2.8°	3.4°	**2.4**°	**2.4**°	**2.4**°
Fertility	5.6°	4.5°	7.0°	5.9°	4.6°	4.7°	**4.4**°

Table 2. Statistics of angular errors of several selected models from our benchmark.

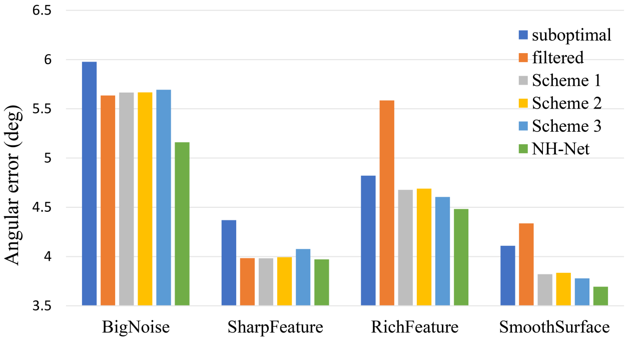

Figure 1. Comparisons of different learning schemes on our synthetic benchmark dataset.

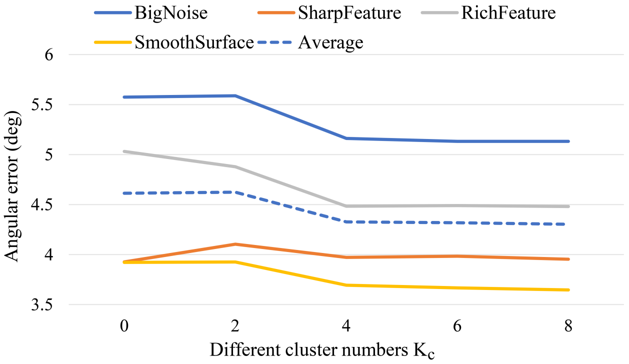

Figure 2. Comparisons of different numbers of training clusters.

4. Additional experiment results

Additionally, we show more visual comparisons (see from Fig. 3) and error comparisons (see from Tab. 2) of partial models in our collected benchmark dataset.

References

[1] Yizhak Ben-Shabat, Michael Lindenbaum, and Anath Fischer. Nesti-net: Normal estimation for unstructured 3d point clouds using convolutional neural networks. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pages 10112–10120, 2019. 2, 3

[2] Alexandre Boulch and Renaud Marlet. Fast and robust normal estimation for point clouds with sharp features. In *Computer graphics forum*, volume 31, pages 1765–1774. Wiley Online Library, 2012. 2, 3

[3] Alexandre Boulch and Renaud Marlet. Deep learning for robust normal estimation in unstructured point clouds. In *Computer Graphics Forum*, volume 35, pages 281–290. Wiley Online Library, 2016. 2, 3

[4] Paul Guerrero, Yanir Kleiman, Maks Ovsjanikov, and Niloy J. Mitra. PCPNet: Learning local shape properties from raw point clouds. *Computer Graphics Forum*, 37(2):75–85, 2018. 2, 3

[5] Hugues Hoppe, Tony DeRose, Tom Duchamp, John McDonald, and Werner Stuetzle. *Surface reconstruction from unorganized points*, volume 26. ACM, 1992. 2, 3

[6] Jie Zhang, Junjie Cao, Xiuping Liu, He Chen, Bo Li, and Ligang Liu. Multi-normal estimation via pair consistency voting. *IEEE transactions on visualization and computer graphics*, 25(4):1693–1706, 2018. 2, 3

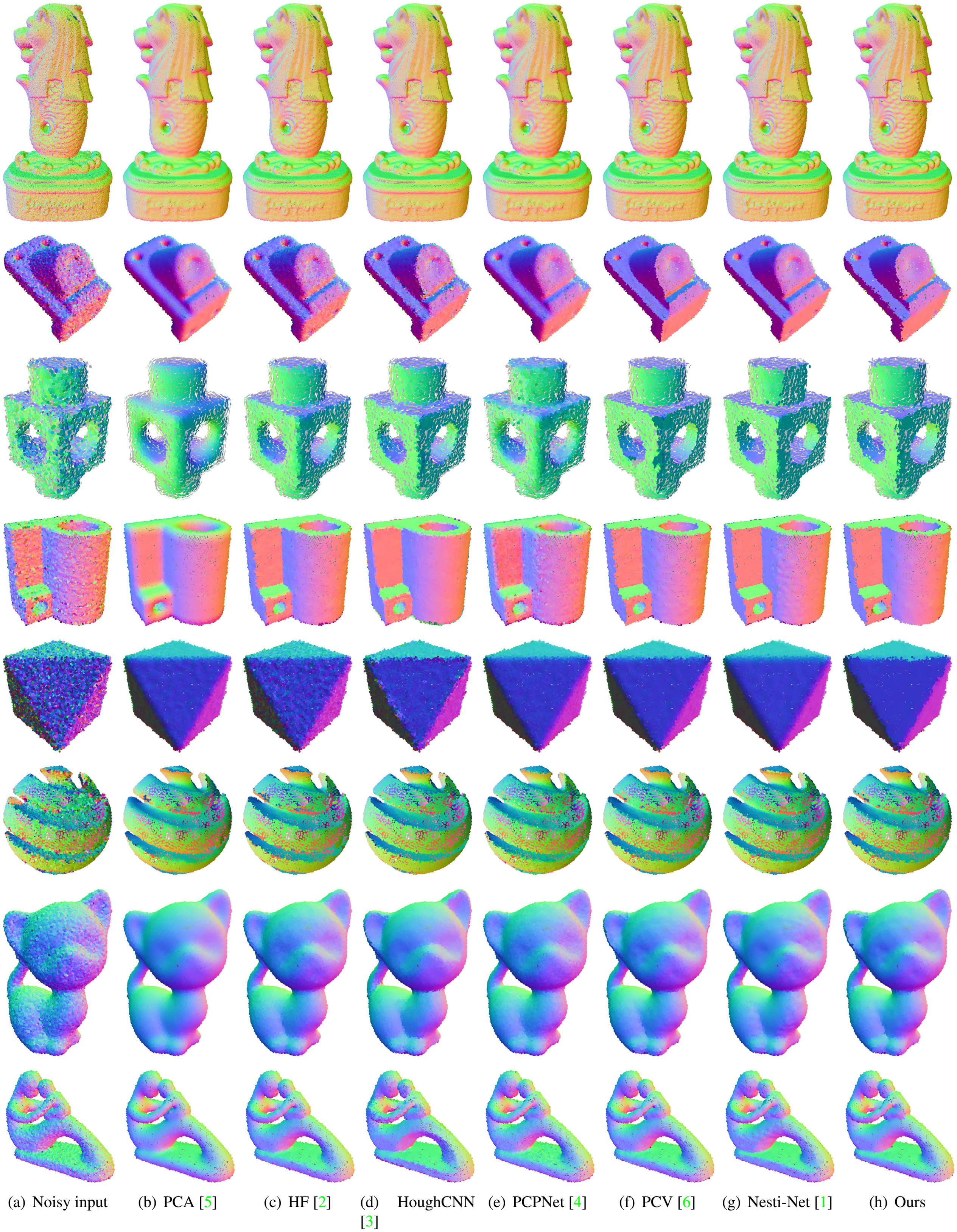
(a) Noisy input (b) PCA [5] (c) HF [2] (d) HoughCNN [3] (e) PCPNet [4] (f) PCV [6] (g) Nesti-Net [1] (h) Ours

Figure 3. Visual comparisons of estimated normals from benchmark dataset results. Point colors are normal vectors mapped to RGB.