AdaFit: Rethinking Learning-based Normal Estimation on Point Clouds

Runsong Zhu[1*] Yuan Liu[2*] Zhen Dong[1†] Yuan Wang[1] Tengping Jiang[1] Wenping Wang[2,3]
Bisheng Yang[1†]

[1]Wuhan University [2]The University of Hong Kong [3]Texas A&M University

1. Architecture details

The detailed architecture is shown in Fig. 3.

2. Training details

To train AdaFit, we use four losses proposed in DeepFit [1], the angle loss between the ground truth normal N_{GT} and the estimation result at the query point N, the offset regularization L_{reg3}, the PointNet's transform matrix regularization L_{reg1} and L_{reg2}, and the consistency loss L_{con}.

$$L_{reg1} = \left| I - A_1 {A_1}^T \right|, \tag{1}$$

$$L_{reg2} = \left| I - A_2 {A_2}^T \right|, \tag{2}$$

$$L_{reg3} = \frac{1}{s} \sum_i |(\Delta x_i, \Delta y_i, \Delta z_i)|_1, \tag{3}$$

$$L_{con} = \frac{1}{s} \left[-\alpha_1 \sum_{j=1}^{s} \log\left(w_j\right) + \alpha_2 \sum_{j=1}^{s} w_j \left| N_{GT,j} \times N_j \right| \right], \tag{4}$$

where $A_1 \in \mathbb{R}^{3\times3}$ is transformation matrix predicted by QSTN and $A_2 \in \mathbb{R}^{3\times3}$ is the transformation matrix predicted by F-STN, $s = 175$ is the smallest neighborhood number used in CSA, $\{w_j | j \in \{1, 2, \ldots, s\}\}$ is the point-wise weight prediction set, $\{N_j | j \in \{1, 2, \ldots, s\}\}$ is the point-wise normal prediction set and $\{N_{GT,j} | j \in \{1, 2, \ldots, s\}\}$ is the corresponding groundtruth normal set for the neighbor points.

Our final loss L is

$$L = |N_{GT} \times N| + L_{con} + \alpha_3 L_{reg1} + \alpha_4 L_{reg2} + \alpha_5 L_{reg3}, \tag{5}$$

where $\alpha_1 = 0.25$, $\alpha_2 = 0.05$, $\alpha_3 = 0.1$, $\alpha_4 = 0.01$ and $\alpha_5 = 0.1$.

We train AdaFit on PCPNet dataset [2] for 600 epochs. We use Adam optimizer with an initial learning rate 5×10^{-4} and a batch size of 256. We decay the learning rate to 0.1 of the original value at epoch 200 and epoch 500.

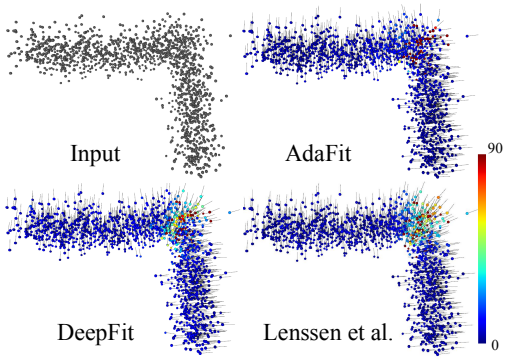

Figure 1. Estimated normals of different methods. The colors of points are encoded by angle error.

3. Evaluation on the SceneNN dataset

To evaluate normal estimation methods on the SceneNN dataset, we randomly sample 1 million points and compute their ground-truth normals from the provided meshes.

4. Analysis on sharp edges

We further provide normals in Fig. 1 on edge regions. The figure shows that AdaFit can predict accurate normals so that the normal directions are sharply changed on the edge. However, due to the noise, it is ambiguous for the network to determine the correct plane for some points very close to the edge, which results in large angle error on these points. In contrast, baseline methods produce normals gradually changing on the edge so that their error are smaller on the edge but much worse in the vicinity.

*Equal contribution
†Corresponding authors: [dongzhenwhu,bshyang]@whu.edu.cn

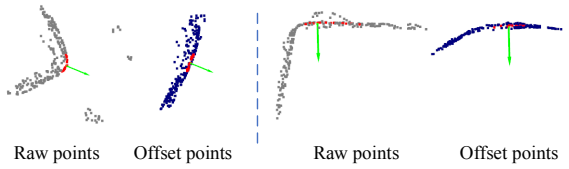

Figure 2. Points before and after being offset.

5. Examples of offset neighboring points

The predicted offsets will change the local geometry of neighboring points to make it easy for the subsequent jet fitting to find an accurate normal for the center point. Especially, as shown in Fig. 2, the points that are not on the same surface will be offset to the surface of the center point while points on the same surface of the center points (red in the figure) will have small offsets. Note we only use the offset neighboring points for the computation of normals but not really use these offset points as the output points. Thus, predicted offsets will not interfere the subsequent surface reconstruction.

6. Derivation of proposition in Section 3.2

The solution to the weighted least square is:

$$\beta = (M^\intercal W M)^{-1} M^\intercal W z. \tag{6}$$

From Eq. 6, we have:

$$\begin{aligned} \mathrm{d}\beta &= -(M^\intercal W M)^{-1}\mathrm{d}(M^\intercal W M)(M^\intercal W M)^{-1}(M^\intercal W z) \\ &\quad + (M^\intercal W M)^{-1}\mathrm{d}(M^\intercal W z) \\ &= -(M^\intercal W M)^{-1}\mathrm{d}(M^\intercal W M)\beta + (M^\intercal W M)^{-1}\mathrm{d}(M^\intercal W z), \end{aligned} \tag{7}$$

and

$$\mathrm{d}(M^\intercal W M) = \mathrm{d}(w_1 M_1^\intercal M_1 + w_2 M_2^\intercal M_2 + \cdots + w_s M_s^\intercal M_s). \tag{8}$$

Based on Eq. 7 and Eq. 8, we have

$$\begin{aligned} \frac{\mathrm{d}\beta}{\mathrm{d}w_i} &= -(M^\intercal W M)^{-1} M_i^\intercal M_i \beta + (M^\intercal W M)^{-1} M_i^\intercal z_i \\ &= (M^\intercal W M)^{-1} M_i^\intercal (z_i - z_i^{'}). \end{aligned} \tag{9}$$

7. More qualitative results

More error maps on the PCPNet dataset are shown in Fig. 4.

References

[1] Yizhak Ben-Shabat and Stephen Gould. Deepfit: 3d surface fitting via neural network weighted least squares. In *European Conference on Computer Vision*, pages 20–34. Springer, 2020. 1

[2] Paul Guerrero, Yanir Kleiman, Maks Ovsjanikov, and Niloy J Mitra. Pcpnet learning local shape properties from raw point clouds. In *Computer Graphics Forum*, volume 37, pages 75–85. Wiley Online Library, 2018. 1

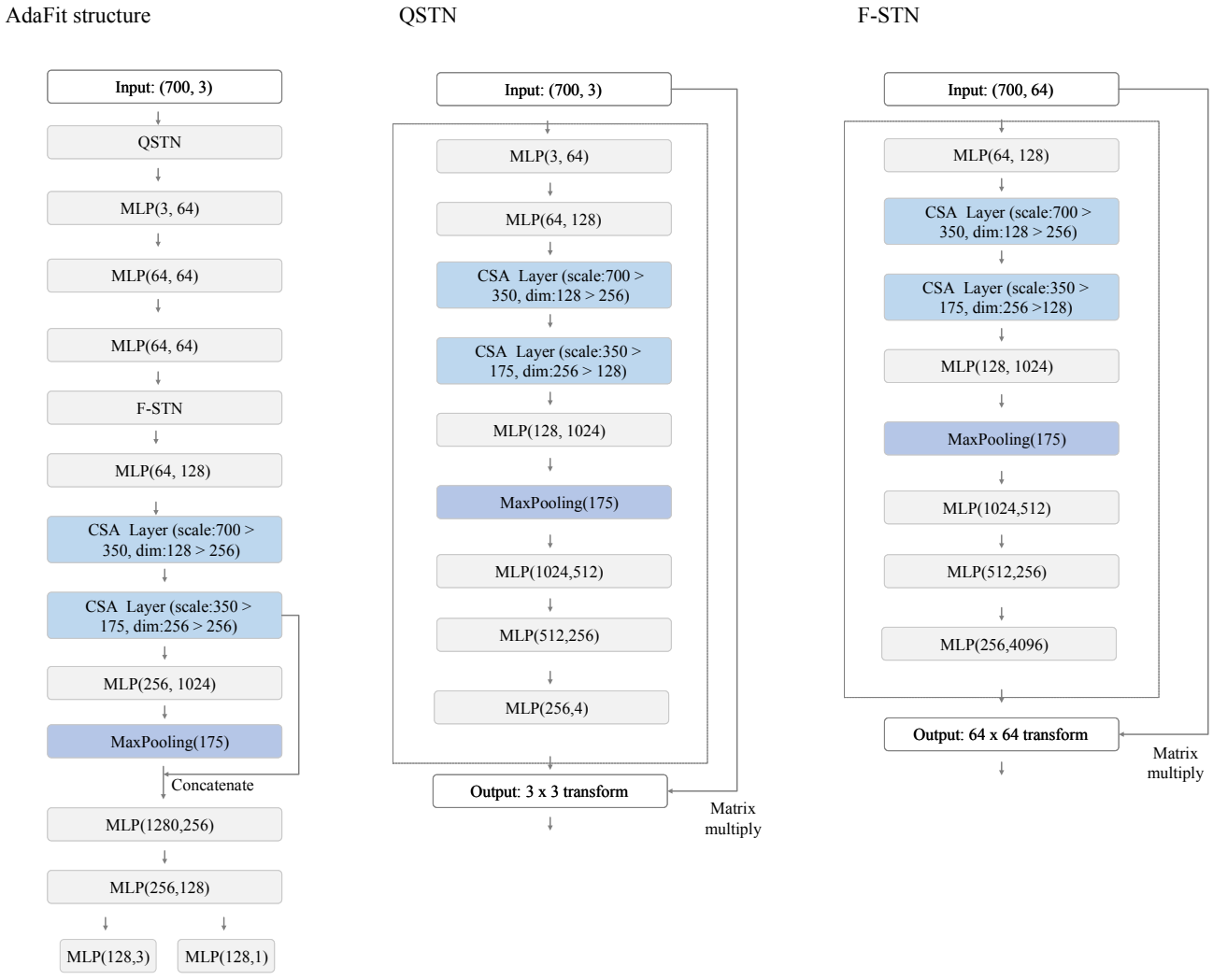

Figure 3. The Architecture details of AdaFit

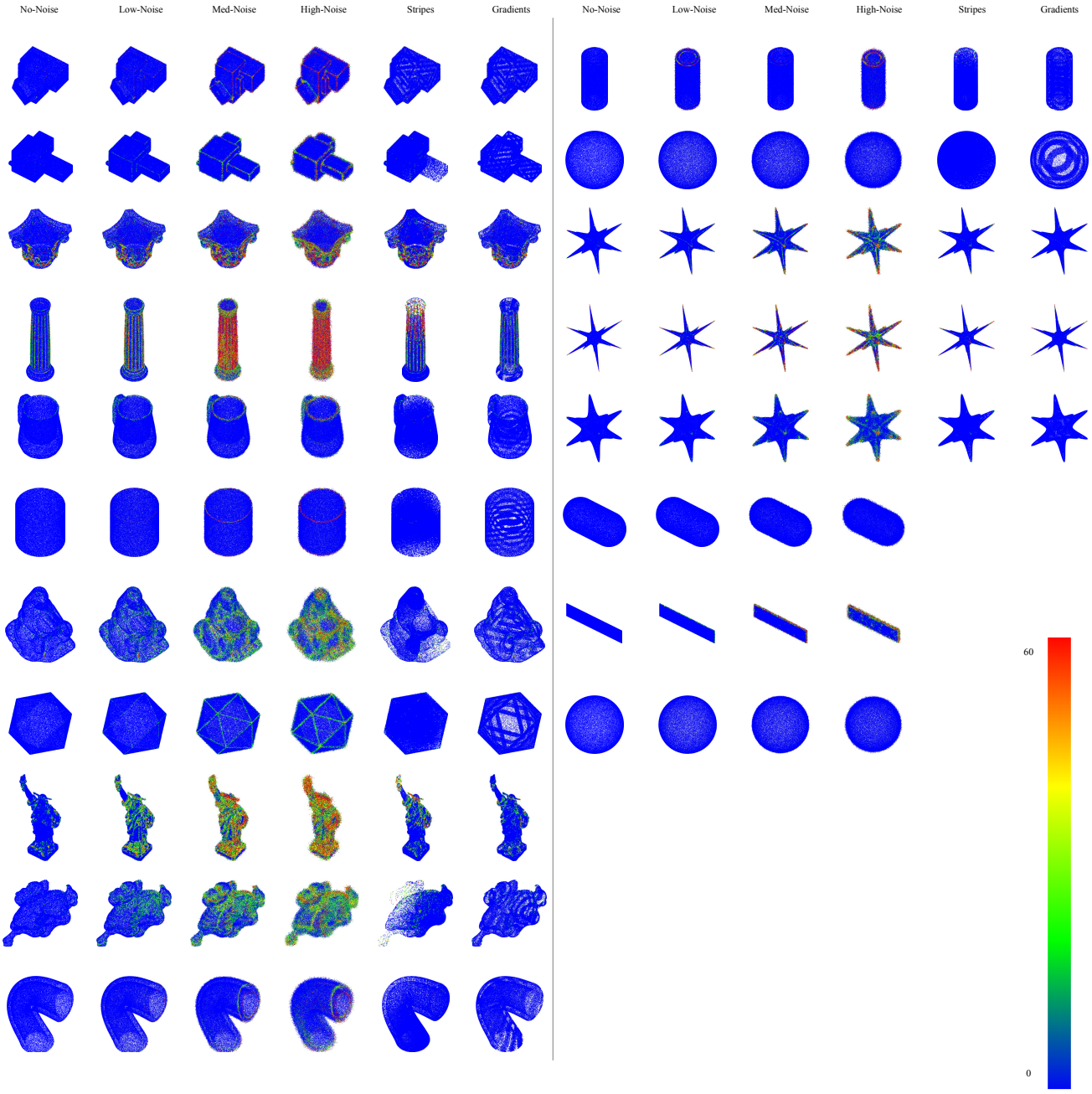

Figure 4. Error maps of normal estimation on the PCPNet testset.