Rethinking the Approximation Error in 3D Surface Fitting for Point Cloud Normal Estimation

Hang Du*, Xuejun Yan*, Jingjing Wang, Di Xie†, and Shiliang Pu
Hikvision Research Institute, Hangzhou, China
{duhang, yanxuejun, wangjingjing9, xiedi, pushiliang.hri}@hikvision.com

Abstract

Most existing approaches for point cloud normal estimation aim to locally fit a geometric surface and calculate the normal from the fitted surface. Recently, learning-based methods have adopted a routine of predicting point-wise weights to solve the weighted least-squares surface fitting problem. Despite achieving remarkable progress, these methods overlook the approximation error of the fitting problem, resulting in a less accurate fitted surface. In this paper, we first carry out in-depth analysis of the approximation error in the surface fitting problem. Then, in order to bridge the gap between estimated and precise surface normals, we present two basic design principles: 1) applies the Z-direction Transform to rotate local patches for a better surface fitting with a lower approximation error; 2) models the error of the normal estimation as a learnable term. We implement these two principles using deep neural networks, and integrate them with the state-of-the-art (SOTA) normal estimation methods in a plug-and-play manner. Extensive experiments verify our approaches bring benefits to point cloud normal estimation and push the frontier of state-of-the-art performance on both synthetic and real-world datasets. The code is available at https://github.com/hikvision-research/3DVision.

1. Introduction

Surface normal estimation on point clouds can offer additional local geometric information for numerous applications, such as denoising [11, 23, 24], segmentation [28–30], registration [9, 17, 27, 33], and surface reconstruction [10, 15, 18, 25, 41]. However, raw-scanned point clouds tend to be incomplete, noisy, and non-uniform, which poses a challenge in accurately estimating surface normals amidst noise, density variations, and missing structures.

Normal estimation on point clouds is a long-standing research topic. The majority of traditional methods [6, 8, 12, 15, 20] aim to fit a local geometric surface (e.g., plane, jet and spherical) around a specific point, and infer the normal from the fitted surface. However, these methods require to carefully tune the setting of parameters, such as point neighborhood sizes, which is sensitive to noise and outliers. With the power of deep neural networks, many learning-based approaches [4, 5, 13, 14, 31, 37, 40] have been proposed to regress surface normal vectors directly, achieving promising performance improvements over traditional methods. However, these approaches exhibit limited generalization capability when applied to real-world point clouds.

More recently, several approaches [3, 21, 42] have generalized the truncated Taylor expansion (n-jet) surface...
model [8] to the learning-based regime, formulating normal estimation as a weighted least-squares problem with learnable weights. In these methods, the point-wise weights of a local surface patch are predicted by a deep neural network, which can control the importance of neighboring points to the fitted surface and alleviate the sensitivity to outliers and noise. Then, the solution of weighted least-squares fitting problem can be expressed in a closed form, which enables to estimate the geometric surface and infer the surface normal. These methods heavily constrain the solution space and obtain a better result for surface normal estimation. Nevertheless, none of them theoretically analyzes the approximation error in surface fitting, leading to a suboptimal normal estimation performance. In some sense, a smaller approximation error represents a more precise estimation. Therefore, we aim to study how to reduce the approximation error and fit a more accurate surface for normal estimation.

In this paper, we analyze the approximation error in the n-jet surface model, and find the existing gap between estimated and accurate normals in previous methods. Specifically, the truncated Taylor expansion polynomial is expected to be equivalent to the height function of the surface, and the accuracy of the reminder term in Taylor expansion has an impact on the precision of normal estimation. As pointed out in [8], to improve the accuracy, a feasible way is to set up a coordinate system where z direction is aligned (has the minimum angle) with the estimated normal. However, we find the previous methods cannot accomplish this objective well, leading to a large estimation error in most cases. Besides, due to the presence of the reminder term and the imperfect data (inevitably containing outliers and noise), it is impossible to achieve an accurate surface fitting without any approximation error. To solve these problems, we propose two basic design principles. First, we apply the z-direction transformation to rotate local patches for a better surface fitting with a lower approximation error. Second, the error of normal estimation is modeled as a term that can be learned in a data-driven manner. The proposed principles can improve the accuracy of the surface fitting, thereby leading to a more precise estimation of surface normals.

To model the above two principles, we implement them with deep neural networks, and introduce two simple yet effective methods: Z-direction Transformation and Normal Error Estimation. More specifically, the z-direction transformation is fulfilled by adding a constraint on the angle between the rotated normal and the z axis, which aims to align the rotated normal with the z direction. To achieve this learning objective, we also design a graph-convolution based alignment transformation network to fully exploit the local neighborhood information for learning a better point transformation. Then, the rough estimated normal can be inferred by any existing polynomial surface fitting method, such as DeepFit [3] and GraphFit [21]. Finally, we design a normal error estimation module that learns a residual term based on the rough estimated result and thus improves the precision of normal estimation.

We conduct comprehensive experiments to verify the effectiveness of our methods on point cloud normal estimation. The proposed two basic design principles are implemented with the existing polynomial surface fitting methods. The experimental results demonstrate our design principles are beneficial to these methods with few extra burdens. As shown in Fig. 1, an obvious improvement can be achieved by our proposed methods for normal estimation.

The contributions of this paper are summarized as:

- We provide an in-depth analysis of the approximation error in n-jet surface fitting, and introduce two basic design principles to improve the precision of 3D surface fitting.
- We implement the design principles with neural networks and propose two approaches, i.e., z-direction transformation and normal error estimation, which can be flexibly integrated with the current polynomial surface fitting methods for point cloud normal estimation.
- We conduct extensive experiments to show the improvements by the proposed methods. The experimental results demonstrate our methods consistently bring benefits and push the frontier of SOTA performance.

2. Related Work

2.1. Traditional Approaches

Normal Estimation on point clouds has been widely studied. A commonly-used way is the Principal Component Analysis (PCA), which can be utilized to estimate a tangent plane by computing the eigenvector with the smallest eigenvalue of a covariance matrix [15]. Subsequently, some approaches [8, 12, 20] are designed to fit a more complex surface (e.g., jet and spherical) by involving more neighboring points. Although these methods enable to be more robust to the noise and outliers, the shape details are over-smoothed due to the large neighborhood size. In order to preserve the shape details, certain methods employ Voronoi diagram [1, 2, 26] or Hough transform [6] for normal estimation. However, they require careful parameters tuning to handle the input points with different noise levels. The above-mentioned methods are sensitive to the setting of parameters, such as the point neighborhood sizes. There is no universal setting that can meet all the challenges.

2.2. Learning-based Methods

Learning-based methods [3,5,7,13,19,21–23,32,36,39, 42] have better robustness for noise and outliers, which can be roughly divided into regression and surface fitting based methods.
Regression based. The estimation of surface normals can be regressed by deep neural networks directly. A group of methods [7, 23, 37] aim to transform the input points into structured data, such as 2D grid representations, and train a Convolutional Neural Network (CNN) to predict the normal vectors. Another kind of methods [4, 13, 14, 22, 31, 38, 39] takes the advantages of point cloud processing network and directly predicts surface normals from unstructured point clouds. For example, PCPNet [13] adopts a multi-vectors. Another kind of methods [4, 13, 14, 22, 31, 38, 39] Convolutional Neural Network (CNN) to predict the normal structured data, such as 2D grid representations, and train a.

3. Theoretical Formulation

The truncated Taylor expansion \((n\text{-jet})\) surface model has been widely used for estimating geometric quantities, such as normal vectors, and curvatures. In this section, we first revisit the theory of \(n\text{-jet}\) surface model, and then analyze the approximation error of \(n\text{-jet}\) fitting methods. To bridge the gap between estimated and precise surface normals, we propose two basic design principles which can be integrated with the current polynomial fitting methods and improve their performance on surface normal estimation.

3.1. Revisiting \(N\text{-jet}\) Surface Fitting

Jet surface model [8] represents a polynomial function that mapping points \((x, y)\) of \(\mathbb{R}^2\) to their height \(z \in \mathbb{R}\) over a surface, where any \(z\) axis is not in the tangent space. In other words, a point \((x, y)\), \(z\) on the surface can be obtained by the height function \(f(x, y)\). Then, an order \(n\) Taylor expansion of the height function over a surface is defined as:

\[
f(x, y) = J_{\beta,n}(x, y) + O\left(||(x, y)||^{n+1}\right),
\]

where the truncated Taylor expansion \(J_{\beta,n}(x, y) = \sum_{k=0}^{n} \sum_{j=0}^{k} \beta_{k-j,j} x^k y^j\) is called a degree \(n\) jet, or \(n\)-jet, and \(O(||(x, y)||^{n+1})\) denotes the remainder term of Taylor expansion. Then, the surface normal given by Eq. 1 is

\[
\hat{n} = \frac{(-\beta_{1,0}, -\beta_{0,1}, 1)}{\sqrt{\beta_{1,0}^2 + \beta_{0,1}^2 + 1}}.
\]

In the approximation case, the number of point \(N_p\) is larger than that of the coefficients \(N_n = (n+1)(n+2)/2\). The \(n\)-jet surface model aims to find an approximation result on the coefficients of the height function, which can be expressed as:

\[
J_{\alpha,n} = \arg\min \left\{ \sum_{i=1}^{N_p} (J_{\alpha,n}(x_i, y_i) - f(x_i, y_i))^2 \right\},
\]

where \(J_{\alpha,n}\) is the solution of the least-square polynomial fitting problem, and \(i = 1, \ldots, N_p\) is the index of a set of points. Moreover, considering the noise and outliers have a large impact on the fitting accuracy, a widely-used way is to extend Eq. 3 to a weighted least-square problem, and thus the solution can be expressed as:

\[
\alpha = (M^TWM)^{-1}(M^TWz),
\]

where \(M = (1, x_i, y_i, \ldots, x_iy_i^{n-1}, y_i^n)_{i=1,\ldots,N_p} \in \mathbb{R}^{N_p \times N_n}\) is the Vandermonde matrix, \(W = \text{diag}(w_1, w_2, \ldots, w_{N_p}) \in \mathbb{R}^{N_p \times N_p}\) is the diagonal point-wise weight matrix and \(z = (z_1, z_2, \ldots, z_{N_p}) \in \mathbb{R}^{N_p}\) is the \(N_p\)-vector of coordinates.

3.2. Analysis of Approximation Error

Recent surface fitting methods [3, 21, 42] have generalized the \(n\)-jet surface model to the learning-based regime, which predict the point-wise weights resorting to the deep neural network. They obtain the coefficients of \(J_{\alpha,n}\) by solving the weighted least-squares polynomial fitting problem, and expect that the coefficients of \(J_{\alpha,n}\) are approximated to those of \(J_{\beta,n}\). In this way, the surface normal can be calculated by Eq. 2.

However, as presented in [8], assuming the convergence rate of approximation is given by the value of the exponent of parameter \(h\) and \(O(h) = ||(x, y)||\), the coefficients \(\beta_{k-j,j}\) of \(J_{\beta,n}(x, y)\) are estimated by those of \(J_{\alpha,n}(x, y)\) up to accuracy \(O(h^{n-k+1})\):

\[
\alpha_{k-j,j} = \beta_{k-j,j} + O(h^{n-k+1}).
\]
DF denotes the angle between the true normal and the estimated normal, e.g., $\hat{n}$, and those of $J_\alpha,n$ and $J_\beta,n$. The error term $\Delta F_h$ at point $p$ for the $n$-jet differential $F$ is defined as $\Delta F_h = \frac{\partial F_h}{\partial h}$. Let $\hat{h}$ be the normal estimated from the fitted surface. Due to the presence of this error term and imperfect data, obtaining precise estimations remains a challenge. Thereby, the normal error estimation is another good choice to improve the precision of surface normal estimation.

4. The Proposed Approach

Motivated by the theoretical formulation outlined above, we introduce two basic design principles to reduce the approximation error and improve the precision of surface normal estimation: 1) we aim to explicitly learn a alignment transformation that rotates input patches for a better surface fitting with a lower approximation error; 2) we model the error of normal estimation as a learnable term which compensates the rough estimated surface normal and yields a more accurate result. Based on these two basic design principles, we propose two simple yet effective methods, i.e., $z$-direction transformation and normal error estimation, which can be flexibly integrated with the current polynomial surface fitting model for further improvements. In the following, we first provide the overview of $n$-jet surface normal estimation network, and then elaborate the implementation details of the proposed methods.

4.1. Overview

As shown in Fig. 2, a general $n$-jet surface fitting network consists of three components, including input transformation, point-wise weight prediction, and normal estimation ($n$-jet fitting). For input transformation, we propose to explicitly constrain the learned transformation matrix, which aims to narrow the angle between the rotated normal.
normal and the $z$ axis. Then, the point-wise weights prediction module can be any existing polynomial surface fitting network, such as DeepFit [3], AdaFit [42], and GraphFit [21]. The weights are utilized for solving the weighted least-square polynomial fitting problem in Eq. 4, and the rough normal can be calculated from the fitted surface. Finally, we learn a normal error term and add it on the rough estimated result to yield a more precise estimation.

### 4.2. Z-direction Transformation

In order to learn transformation invariant features, PointNet [28] adopts a spatial transformation network (STN) to align the input points to a canonical view. In terms of surface normal estimation, rotation transformation is a more proper choice, since it can stabilize the convergence of network [13]. Thus, previous methods [3, 13, 21, 42] tend to learn a quaternion spatial transformation, it is still challenging to yield precise normal estimations, due to the presence of error term $DF(\theta_\beta + O(h^n)) (O(h^n))$ in the differential quantity function (Eq. 6) and imperfect data which inevitably contains the noise and outliers. In the pursuit of a more precise normal estimation, we propose to estimate the error of normal estimation in a data-driven manner. To be specific, we consider the rough estimated normal $\hat{n}^e_i$ of the fitted surface should be updated by learning a residual term,$$
abla(\hat{n}^e_i) = \phi(\text{Concat}(x_i, \hat{n}^e_i)),$$

where $x_i$ is the point-wise feature, and $\phi(\cdot)$ is a mapping function, i.e., MLPs. Then, we compute the final output normal $\hat{n}^e_i$ by adding $\Delta(\hat{n}^e_i)$ on the rough estimated normal,$$
abla_i = \hat{n}^e_i + \Delta(\hat{n}^e_i).$$

In this way, the network also enables to adjust the inaccurate surface fitting brought by noise and outliers. Finally, we can reduce the error of surface normal estimation and thereby yield a more accurate estimation result.
Table 1. Normal angle RMSE of our methods and baseline models on PCPNet dataset. After being integrated with our methods, the state-of-the-art surface fitting methods obtain significant improvements on point cloud normal estimation.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>No Noise</td>
<td>4.11 (0.25)</td>
<td>4.71 (0.36)</td>
<td>4.90 (1.05)</td>
<td>4.45</td>
<td>5.19</td>
<td>6.51</td>
<td>6.72</td>
<td>6.99</td>
<td>9.62</td>
<td>12.25</td>
<td>12.29</td>
</tr>
<tr>
<td>Noise ($\sigma = 0.125%$)</td>
<td>8.66</td>
<td>8.75</td>
<td>8.91</td>
<td>8.74</td>
<td>9.05</td>
<td>9.21</td>
<td>9.95</td>
<td>10.11</td>
<td>11.37</td>
<td>12.84</td>
<td>12.87</td>
</tr>
<tr>
<td>Noise ($\sigma = 0.6%$)</td>
<td>16.02</td>
<td>16.31</td>
<td>16.61</td>
<td>16.05</td>
<td>16.44</td>
<td>16.72</td>
<td>17.18</td>
<td>17.63</td>
<td>18.87</td>
<td>18.33</td>
<td>18.38</td>
</tr>
<tr>
<td>Noise ($\sigma = 1.2%$)</td>
<td>21.57</td>
<td>21.64</td>
<td>22.87</td>
<td>21.64</td>
<td>21.94</td>
<td>23.12</td>
<td>21.96</td>
<td>22.28</td>
<td>23.28</td>
<td>27.68</td>
<td>27.50</td>
</tr>
<tr>
<td>Density (Gradient)</td>
<td>4.83</td>
<td>5.51</td>
<td>5.52</td>
<td>5.22</td>
<td>5.90</td>
<td>7.31</td>
<td>7.73</td>
<td>9.00</td>
<td>11.70</td>
<td>13.13</td>
<td>12.81</td>
</tr>
<tr>
<td>Density (Striped)</td>
<td>4.89</td>
<td>5.48</td>
<td>5.70</td>
<td>5.48</td>
<td>6.01</td>
<td>7.92</td>
<td>7.51</td>
<td>8.47</td>
<td>11.16</td>
<td>13.39</td>
<td>13.66</td>
</tr>
<tr>
<td>Average</td>
<td>10.01</td>
<td>10.40</td>
<td>10.75</td>
<td>10.26</td>
<td>10.76</td>
<td>11.80</td>
<td>11.84</td>
<td>12.41</td>
<td>14.34</td>
<td>16.29</td>
<td>16.25</td>
</tr>
</tbody>
</table>

Note that the proposed normal error estimation is parallel to the point offset learning in AdaFit [42]. In the following experiments, we can achieve a further improvement on AdaFit with our methods. Moreover, compared with predicting the normal directly, residual error prediction is much easier and stable for the network.

4.4. Implementation Details

Network Architecture. In this study, we propose two basic design principles and implement them with DeepFit [3], AdaFit [42], and GraphFit [21]. So, we adopt their original network architecture with two displacements. First, the spatial transformation network is replaced with our GCN-based transformation network, and the learned transformation matrix is constrained by Eq. 8. Second, we add two layers of MLPs to regress the residual terms for adjusting the rough estimated surface normal. More details on network architecture can be found in the supplementary materials.

Training Supervision. To train the network, we employ the same loss functions in DeepFit [3] for both rough estimated normal $\hat{n}_i$ and refined normal $\hat{n}_i'$.\[ L_{normal} = |\hat{n}_i \times \hat{n}_i'| + |\hat{n}_i \times \hat{n}_i'|. \] (11)

We also utilize the neighborhood consistency loss $L_{con}$ and transformation regularization loss $L_{reg}$ in DeepFit. Moreover, as above presented, we add a penalty term on transformation matrix $L_{trans}$. Thus, the total training loss is\[ L_{total} = L_{normal} + \lambda_1 L_{con} + \lambda_2 L_{reg} + \lambda_3 L_{trans}, \] (12)

where we empirically set $\lambda_1 = 0.25$, $\lambda_2 = 0.1$, and $\lambda_3 = 2$ in the experiments.

5. Experiment

5.1. Datasets and Experimental Settings

Datasets. We follow the same configuration of previous works that adopt synthetic PCPNet dataset for training, which includes four CAD objects and four high quality scans of figurines with total 3.2M training examples. Then, the trained models are evaluated on PCPNet test set with six categories, including four sets with different levels of noise, i.e., no noise, low noise ($\sigma = 0.125\%$), med noise ($\sigma = 0.6\%$), and high noise ($\sigma = 1.2\%$), and two sets with varying sampling density (gradient and striped pattern). To verify the generalization ability, we also employ a real-world dataset, SceneNN [16], for both quantitative and qualitative evaluation.

Training. The polynomial order $n$ for the surface fitting is 3. Adam algorithm is used for model optimization. Our models are trained for 700 epochs with a batch size of 256. The learning rate begins at 0.001 and drops by a decay rate of 0.1 at 300 and 550 epochs.

Evaluation. We select three recent state-of-the-art methods as baseline, including DeepFit [3], AdaFit [42] and GraphFit [21]. Besides, we also compare with traditional methods and learning-based normal regression methods. The root-mean-squared error (RMSE) of angles between the estimated and the ground-truth normals to evaluate the performance. Moreover, we also report the percentage of good points with a threshold of error (PGP $\alpha$).

5.2. SOTA Results on Synthetic Data

Quantitative results. Table 1 reports the RMSE comparison of the exiting methods on PCPNet dataset. The results imply that the proposed two basic ideas can be flexibly integrated with these polynomial surface fitting methods, and obtain evident improvements over the baseline counterparts. Moreover, as shown in Fig. 4, we further provide normal error Area Under the Curve (AUC) results of SOTA polynomial fitting models with or without our methods. Table 2 also gives the quantitative results of PGP5 and PGP10 under no noise setting. From the results, we consistently improve the baseline models under different error thresholds, especially on the point clouds with density variations and low noise. The reason behind has two folds. First, the z-direction transformation enables to achieve a better surface fitting with a lower approximation error. Second, the normal error estimation can further update the rough estimated normal to a more accurate result.
Figure 4. Normal error AUC results of state-of-the-art models with or without our methods on PCPNet dataset. The x-axis represents the threshold of error, and the y-axis represents the percentage of good points which have the lower error than a given threshold.

Table 2. Comparison of percentage of good points PGP5 and PGP10 on PCPNet dataset under no noise. Higher is better.

<table>
<thead>
<tr>
<th>Method</th>
<th>DeepFit PGP5</th>
<th>DeepFit PGP10</th>
<th>AdaFit PGP5</th>
<th>AdaFit PGP10</th>
<th>GraphFit PGP5</th>
<th>GraphFit PGP10</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline</td>
<td>80.03</td>
<td>90.72</td>
<td>88.24</td>
<td>94.36</td>
<td>89.73</td>
<td>95.66</td>
</tr>
<tr>
<td>+ Ours</td>
<td>89.83</td>
<td>95.59</td>
<td>90.40</td>
<td>95.82</td>
<td>91.28</td>
<td>96.59</td>
</tr>
</tbody>
</table>

**Qualitative results.** As shown in Fig. 5, we visualize the angle errors for the baseline models with or without the proposed methods. From the results, we can observe our methods improve the robustness of baseline models on all the areas, such as curved regions and sharp edges. More visualization results can be found in the supplementary material.

### 5.3. Comparison on Real-world Data

In order to validate the proposed methods on real-world scenarios, we choose SceneNN [16] dataset for evaluation, which contains 76 scenes captured by a depth camera. We follow the settings of AdaFit [42] to obtain the sampled point clouds and ground-truth normals from provided ground-truth reconstructed meshes. The models trained on PCPNet dataset are directly utilized for evaluation. Table 3 gives the quantitative results on all the scenes. The results show that our approaches also bring benefits to the baseline models on real-world data. Moreover, as shown in Fig. 6, we randomly choose several scenes to visualize the normal errors. We can observe the real-world data is incomplete with many outliers and noise, which is a more challenging than synthetic data. Nevertheless, we can consistently improve the performance of SOTA models. Both quantitative and qualitative experiments demonstrate the good generalization ability of our methods on real-world data.

<table>
<thead>
<tr>
<th>Method</th>
<th>DeepFit</th>
<th>AdaFit</th>
<th>GraphFit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline</td>
<td>17.13</td>
<td>15.49</td>
<td>14.79</td>
</tr>
<tr>
<td>+ Ours</td>
<td>14.57 (2.56)</td>
<td>14.45 (1.04)</td>
<td>14.51 (0.28)</td>
</tr>
</tbody>
</table>
Table 4. Normal angle RMSE of SOTA models with or without the proposed methods on PCPNet dataset.

<table>
<thead>
<tr>
<th>Aug.</th>
<th>DeepFit (size = 256)</th>
<th>AdaFit (size = 700)</th>
<th>GraphFit (size = 500)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Z-direction Trans.</td>
<td>✓ ✓</td>
<td>✓ ✓</td>
<td>✓ ✓</td>
</tr>
<tr>
<td>No Noise</td>
<td>6.51 6.27 5.01 4.90</td>
<td>5.19 4.93 4.72 4.71</td>
<td>4.45 4.27 4.36 4.11</td>
</tr>
<tr>
<td>Low Noise</td>
<td>9.21 9.10 9.09 8.91</td>
<td>9.05 8.94 8.81 8.75</td>
<td>8.74 8.79 8.71 8.66</td>
</tr>
<tr>
<td>Gradient</td>
<td>7.31 7.17 5.69 5.52</td>
<td>5.90 5.63 5.59 5.51</td>
<td>5.22 4.98 5.06 4.83</td>
</tr>
<tr>
<td>Striped</td>
<td>7.92 7.73 5.85 5.70</td>
<td>6.01 5.89 5.62 5.48</td>
<td>5.48 5.10 5.18 4.89</td>
</tr>
<tr>
<td>Average</td>
<td>11.80 11.66 10.87 <strong>10.75</strong></td>
<td>10.76 10.56 10.48 <strong>10.40</strong></td>
<td>10.26 10.14 10.16 <strong>10.01</strong></td>
</tr>
</tbody>
</table>

Table 5. Normal angle RMSE with different Jet order n on PCPNet dataset. We choose single-scale AdaFit with 256 neighborhood size as baseline.

<table>
<thead>
<tr>
<th>Order</th>
<th>Baseline</th>
<th>Ours</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>2</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>3</td>
<td>✓</td>
<td>✓</td>
</tr>
</tbody>
</table>

Table 6. Model complexity comparison. The inference time per point is tested on a NVIDIA TITAN X.

<table>
<thead>
<tr>
<th>Method</th>
<th>Params (M)</th>
<th>Model size (MB)</th>
<th>Time (ms)</th>
<th>Avg. error</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nesti-Net [5]</td>
<td>170.10</td>
<td>2,010.00</td>
<td>-</td>
<td>12.41</td>
</tr>
<tr>
<td>DeepFit [3]</td>
<td><strong>3.44</strong></td>
<td><strong>13.53</strong></td>
<td><strong>0.47</strong></td>
<td>11.80</td>
</tr>
<tr>
<td>AdaFit [42]</td>
<td>4.07</td>
<td>16.14</td>
<td>0.49</td>
<td>10.76</td>
</tr>
<tr>
<td>GraphFit [21]</td>
<td>4.16</td>
<td>16.38</td>
<td>2.07</td>
<td>10.26</td>
</tr>
<tr>
<td>DeepFit + Ours</td>
<td>3.64</td>
<td>14.26</td>
<td>0.56</td>
<td>10.75</td>
</tr>
<tr>
<td>AdaFit + Ours</td>
<td>4.28</td>
<td>16.85</td>
<td>0.58</td>
<td>10.40</td>
</tr>
<tr>
<td>GraphFit + Ours</td>
<td>4.36</td>
<td>17.09</td>
<td>2.19</td>
<td><strong>10.01</strong></td>
</tr>
</tbody>
</table>

5.4. Ablation Study

To verify the effectiveness of our methods, we conduct extensive ablation studies on PCPNet dataset.

**Influence of model components.** Firstly, we check the performance gain on DeepFit, AdaFit, and GraphFit, by integrating the proposed two methods with them. Table 4 reports the ablation results. As we can see, all baseline models consistently obtain performance improvement after integrating with a solo version of our methods (z-direction transformation or normal error estimation). Moreover, they can achieve the best normal estimation results with the combination of two model components. The results imply that our methods enable to reduce the approximation error of the fitting problem and improve the precision of surface normal. Note that we conduct above experiments under their optimal setting of input neighborhood size. So, the experimental results can also verify that our methods are robust to the neighborhood size, which can bring benefits to all baseline models under different settings.

**Robustness against the Jet orders.** Moreover, we conduct experiments to verify the robustness of our methods against the polynomial order n. Considering the training consumption and computing resource, we choose the single-scale AdaFit as baseline, and the input neighborhood size is 256. The results in Table 5 show all the models have similar performance, indicating our methods can also work well under the different polynomial order n.

5.5. Model Complexity Analysis

In addition, we make a comparison on model complexity. As given in Table 6, we only increase few extra burdens in terms of model complexity and bring evident improvement, indicating that the proposed methods are light-weight yet effective. For instance, after integrating with our methods, DeepFit is able to outperform the original AdaFit with fewer parameters. Thus, we can achieve a better practical implementation for the balance between the model accuracy and efficiency.

6. Conclusion

In this work, we study the approximation error in existing 3D surface fitting methods, and find there exists gaps between estimated and precise surface normals. To handle this problem, we propose two basic design principles, i.e., z-direction transformation and normal error estimation. The former is able to provide a better surface fitting with a lower approximation error, and the latter can adjust the rough estimated normal to a more precise result. The proposed two principles can be flexibly integrated with the current SOTA polynomial surface fitting methods in a plug-and-play manner, and achieve significant improvements on both synthetic and real-world datasets.
References


