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.

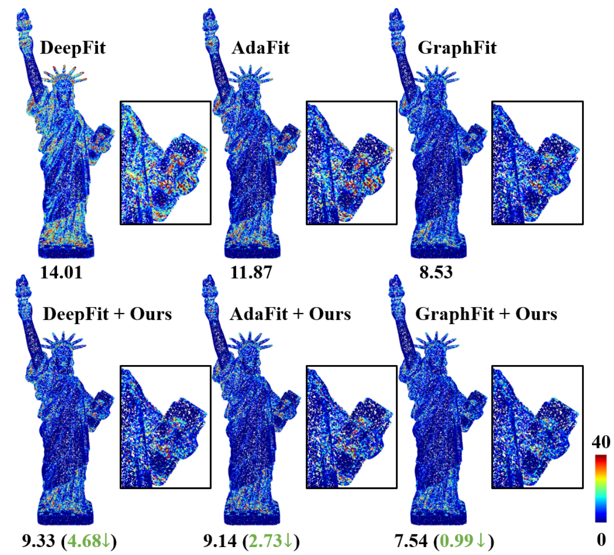


Figure 1. The error heatmap of point cloud normal estimation. The first row is produced by three SOTA surface fitting methods, while the second row shows the results of integrating our method with them. The bottom values indicate the corresponding normal angle root mean square error (RMSE) . Both quantitative and qualitative results demonstrate that our proposed method provides more precise normal estimation.

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

*These authors contributed equally to this work.
†Corresponding author.

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-scale PointNet [28] to process the point clouds with different neighborhood sizes simultaneously, and the patch-wise global features are regressed to the geometric properties consequently. Moreover, Nesti-Net [5] and Zhou *et al*. [39] propose to select the optimal neighborhood size by designing a multi-scale architecture and choose the corresponding sub-network for normal estimation. Although these methods perform better than traditional routine, they lack of generalization ability on unseen data.

Surface fitting based. In contrast to regressing the normal vectors directly, recent approaches [3, 19, 21, 42] aim to integrate deep learning with solving the least-squares surface fitting problem. Among them, lenssen *et al*. [19] propose an iterative estimation method that predicts point weights for plane fitting and iteratively adjusts the weights to a better fitting. Other methods [3, 21, 42] extend the n-jet surface fitting [8] to the learning-based regime. They also employ the neural network to predict point-wise weights which can be regarded as a selection of sample points. The surface normal is calculated from the solution of polynomial surface fitting problem. To improve the precision of normal estimation, AdaFit [42] adjusts the position of neighboring points by regressing the point offsets, and GraphFit [21] designs a more powerful network for learning point-wise weights. In this work, we follow the same routine and study the approximation error of polynomial surface 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. Theoretical Formulation

The truncated Taylor expansion (n-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-jet surface model, and then analyze the approximation error of n-jet fitting problem, which facilitates the next section to present our two basic design principles that pursue a more precise normal estimation.

3.1. Revisiting N-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, given 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:

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

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

$$\hat{\mathbf{n}} = \frac{(-\beta_{1,0}, -\beta_{0,1}, 1)}{\sqrt{\beta_{1,0}^2 + \beta_{0,1}^2 + 1}}. \tag{2}$$

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:

$$J_{\alpha,n} = \arg\min \left\{ \sum_{i=1}^{N_p} \left(J_{\alpha,n}\left(x_i, y_i\right) - f\left(x_i, y_i\right)\right)^2 \right\}, \tag{3}$$

where $J_{\alpha,n}$ is the solution of the least-square polynomial fitting problem, and $i = 1, ..., N_p$ 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:

$$\boldsymbol{\alpha} = \left(\mathbf{M}^{\top}\mathbf{W}\mathbf{M}\right)^{-1}\left(\mathbf{M}^{\top}\mathbf{W}\mathbf{z}\right), \tag{4}$$

where $\mathbf{M} = \left(1, x_i, y_i, \ldots, x_i y_i^{n-1}, y_i^n\right)_{i=1,\ldots,N_p} \in \mathbb{R}^{N_p \times N_n}$ is the Vandermonde matrix, $\mathbf{W} = \operatorname{diag}\left(w_1, w_2, \ldots, w_{N_p}\right) \in \mathbb{R}^{N_p \times N_p}$ is the diagonal point-wise weight matrix and $\mathbf{z} = \left(z_1, z_2, \ldots, z_{N_p}\right) \in \mathbb{R}^{N_p}$ 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 $\mathrm{O}(h) = \|(x_i, y_i)\|$, 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 $\mathrm{O}(h^{n-k+1})$:

$$\alpha_{k-j,j} = \beta_{k-j,j} + \mathrm{O}(h^{n-k+1}). \tag{5}$$

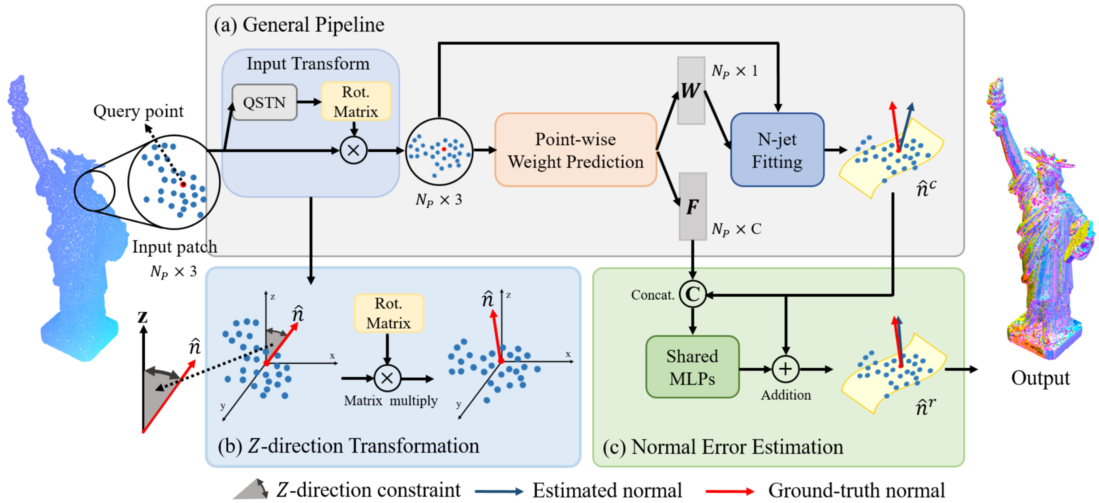


Figure 2. (a) The overall architecture of a general n-jet fitting network. The input patch is fed into the spatial transformation network to learn a quaternion for rotation transformation. Then, point-wise weights are predicted from the rotated points. Finally, the weight predictions are utilized to solve n-jet fitting problem, and the normal is inferred from the fitted surface. (b) The proposed Z**-direction Transformation**: the rotation matrix is explicitly constrained by z-direction transformation, which aims to align the rotated normal to the z axis. (c) The proposed **Normal Error Estimation**: the rough estimated surface normal is further adjusted by learnable error estimation.

Hence, there exists a error term $\mathrm{O}(h^{n-k+1})$ between the coefficients $\beta_{k-j,j}$ of $J_{\beta,n}$ and those of $J_{\alpha,n}$, and a smaller error enables to yield a more precise estimation result.

As proof in [8], *the error estimates are better when the convex hull of the sample points is not too "flat"*. To be specific, $\mathrm{O}\left(h^{n-k+1}\right)$ depends on the supremum of $\left\{\left\|D^{n+1}f(x,y)\right\|; (x,y)\in K\right\}$, where K is the convex hull of the point set $\{(x_i,y_i)\}_{i=1,\ldots,N}$. Let d_{max} be the diameter of K and d_{min} be the supremum of the diameter of disks inscribed in K. A small ratio d_{max}/d_{min} means that the geometry of K is not too "flat", which can reduce the supremum of $\left\|D^{n+1}f(x,y)\right\|$, leading to a better surface fitting with a lower error. To decrease the ratio, one should take a coordinate system as close as possible to the Monge system, where the z axis is aligned with the true normal. So, a good choice is to rotate the point cloud and minimize the angle between the rotated normal and the z axis. By doing so, one can reduce the approximation error and confirm the convergence of the surface fitting problem well.

In addition, *a polynomial fitting of degree n estimates any kth-order differential quantity to accuracy* $\mathrm{O}(h^{n-k+1})$. Here, we take the notations of a curve for simplicity,

$$
\begin{aligned}
&F\left(\alpha_{i=0,\ldots,k}\right) = F\left(\beta_{i=0,\ldots,k} + \mathrm{O}(h^{n-k+1})\right) \\
&= F\left(\beta_{i=0,\ldots,k}\right) + DF_{(\beta_i+\mathrm{O}(h^{n-k+1}))_{i=0,\ldots,k}}\left(\mathrm{O}(h^{n-k+1})\right),
\end{aligned} \tag{6}
$$

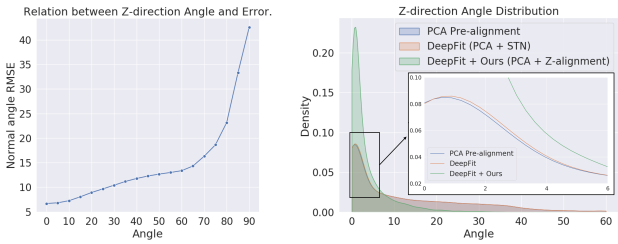
where $F(\alpha_i)$ is the differential quantity (*e.g.*, curvatures or normals) function of the n-jet coefficients $\alpha_{i=0,\ldots,k}$, and DF_p is the differential of F at point p. For normal estimation, *i.e.*, $k=1$, the error term $DF_{(\beta_i+\mathrm{O}(h^n))}\left(\mathrm{O}(h^n)\right)$ denotes the angle between the true normal and the estimated normal. 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



(a) (b)

Figure 3. The analysis of z-direction transformation. (a) The relation between the error of normal estimation and the z-direction angle. A larger z-direction angle usually increases the error. (b) The distribution of the angle between the rotated true normal and the z axis. In the original DeepFit [3], approximate 30% cases exhibit a large angle (≥ 20) with the z axis after transformation.

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 for rotation transformation.

As explained in Sec. 3.2, the accuracy of surface fitting problem can be improved by minimizing the angle between the rotated normal and the z axis. Here, we argue that the error of normal estimation can be decreased when the z-direction angle is minimized. To further support this argument, we conduct a toy experiment on PCPNet [13] test set, and analyze the relation between the z-direction angle and the normal estimation error. In Fig. 3a, we can find the error is positively related to the z-direction angle of rotated normals, which means a larger z-direction angle usually increases the normal error. Moreover, we inspect the angle distribution between the rotated ground-truth normal and the z axis. The angle distribution is shown in Fig. 3b, where the horizontal axis denotes the value of angle and the vertical axis denotes the density. In the original DeepFit model, approximate 30% cases exhibit a large angle (≥ 20) with the z axis after transformation suggesting that the model is unable to learn the optimal transformation for achieving the best surface fitting, Thereby, it fails to obtain a precise normal estimation result.

From the above analysis, we can see the mechanism of z-direction alignment transformation enables to reduce the approximation error and thus improve the accuracy of the surface fitting problem. However, the existing transformation network cannot fulfill this objective well. To address it, we aim to explicitly learn a quaternion spatial transformation $\mathbf{T}$ that sets the rotated ground-truth normal aligned with the axis z, which can be formulated as

$$\hat{\mathbf{z}} = \hat{\mathbf{n}}_i \mathbf{T}, \tag{7}$$

where $\hat{\mathbf{z}} = (0, 0, 1)^\top$, $\hat{\mathbf{n}}_i$ is the ground-truth normal, and $\mathbf{T}$ is the learned transformation. We expect the rotated normal has the minimum angle with the z axis. To accomplish this objective, we design a z-direction transformation loss,

$$\mathcal{L}_{\text{trans}} = \left| \hat{\mathbf{n}}_i \mathbf{T} \times \hat{\mathbf{z}} \right|. \tag{8}$$

This loss term can explicitly constrain the learned alignment transformation towards the z direction. However, the expected transformation is non-trivial by simply using a chain of Multi-layer Perceptrons (MLPs) in [3, 13, 42]. To this end, we design a graph-convolution (GCN) based spatial transformation network by fully exploiting the local neighborhood information. Specifically, three EdgeConv [34] layers with two adaptive graph pooling [35] are utilized to learn multi-scale point features. Then, a combination of max and average pooling operations is adopted to generate a global feature which is further reduced to a quaternion for z-direction transformation. As shown in Fig. 3b, more angles tend to zero under the green curve, which means our z-direction transformation indeed aligns the normal to the z axis, and thereby leads to a more accurate surface fitting.

4.3. Normal Error Estimation

Although we can reduce the approximation error via z-alignment transformation, it is still challenging to yield precise normal estimations, due to the presence of error term $DF_{(\beta_i + \mathrm{O}(h^n))}\left(\mathrm{O}(h^n)\right)$ 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 $\hat{\mathbf{n}}_i^c$ of the fitted surface should be updated by learning a residual term,

$$\Delta(\hat{\mathbf{n}}_i^c) = \phi(\mathrm{Concat}(x_i, \hat{\mathbf{n}}_i^c)), \tag{9}$$

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 $\hat{\mathbf{n}}_i^r$ by adding $\Delta(\hat{\mathbf{n}}_i^c)$ on the rough estimated normal,

$$\hat{\mathbf{n}}_i^r = \hat{\mathbf{n}}_i^c + \Delta(\hat{\mathbf{n}}_i^c). \tag{10}$$

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.

Aug.	GrapFit + Ours	AdaFit + Ours	DeepFit + Ours	GraphFit [21]	AdaFit [42]	DeepFit [3]	IterNet [19]	Nesti-Net [5]	PCPNet [13]	Jet [8]	PCA [15]
No Noise	**4.11**	4.71	4.90	4.45	5.19	6.51	6.72	6.99	9.62	12.25	12.29
Noise ($\sigma = 0.125\%$)	**8.66**	8.75	8.91	8.74	9.05	9.21	9.95	10.11	11.37	12.84	12.87
Noise ($\sigma = 0.6\%$)	**16.02**	16.31	16.61	16.05	16.44	16.72	17.18	17.63	18.87	18.33	18.38
Noise ($\sigma = 1.2\%$)	**21.57**	21.64	22.87	21.64	21.94	23.12	21.96	22.28	23.28	27.68	27.50
Density (Gradient)	**4.83**	5.51	5.52	5.22	5.90	7.31	7.73	9.00	11.70	13.13	12.81
Density (Striped)	**4.89**	5.48	5.70	5.48	6.01	7.92	7.51	8.47	11.16	13.39	13.66
Average	**10.01** (**0.25↓**)	10.40 (**0.36↓**)	10.75 (**1.05↓**)	10.26	10.76	11.80	11.84	12.41	14.34	16.29	16.25

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 $\hat{\mathbf{n}}_i^c$ and refined normal $\hat{\mathbf{n}}_i^r$,

$$\mathcal{L}_{\text{normal}} = |\hat{\mathbf{n}}_i \times \hat{\mathbf{n}}_i^c| + |\hat{\mathbf{n}}_i \times \hat{\mathbf{n}}_i^r| \,. \tag{11}$$

We also utilize the neighborhood consistency loss $\mathcal{L}_{\text{con}}$ and transformation regularization loss $\mathcal{L}_{\text{reg}}$ in DeepFit. Moreover, as above presented, we add a penalty term on transformation matrix $\mathcal{L}_{\text{trans}}$. Thus, the total training loss is

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{normal}} + \lambda_1 \mathcal{L}_{\text{con}} + \lambda_2 \mathcal{L}_{\text{reg}} + \lambda_3 \mathcal{L}_{\text{trans}}, \tag{12}$$

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 α).

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.

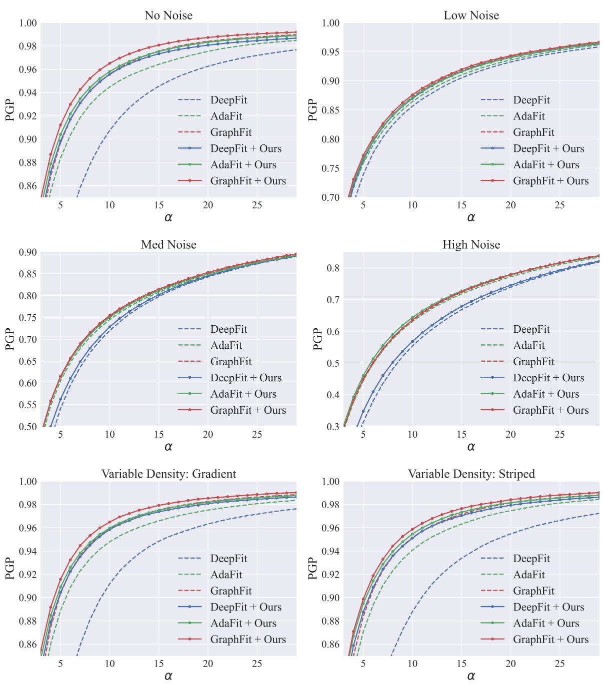


Figure 4. Normal error AUC results of state-of-the-art models with or without our methods on PCPNet dataset. The x axis represents the threshold of error, and the y axis represents the percentage of good points which have the lower error than a given threshold.

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

Method	DeepFit		AdaFit		GraphFit	
	PGP5	PGP10	PGP5	PGP10	PGP5	PGP10
Baseline	80.03	90.72	88.24	94.36	89.73	95.66
+ Ours	89.83	95.59	90.40	95.82	**91.28**	**96.59**

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.

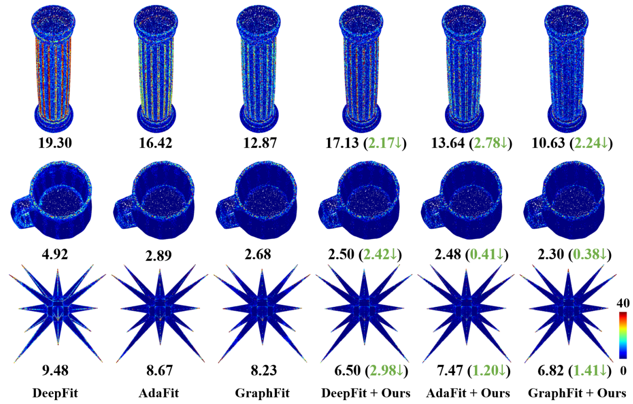


Figure 5. The error heatmap of normal estimation on PCPNet dataset. The bottom values represents the average error. The angle errors are mapped to a range from 0° to 60°.

Table 3. Normal angle RMSE on SceneNN dataset.

Method	DeepFit	AdaFit	GraphFit
Baseline	17.13	15.49	14.79
+ Ours	14.57 (**2.56↓**)	14.45 (**1.04↓**)	14.51 (**0.28↓**)

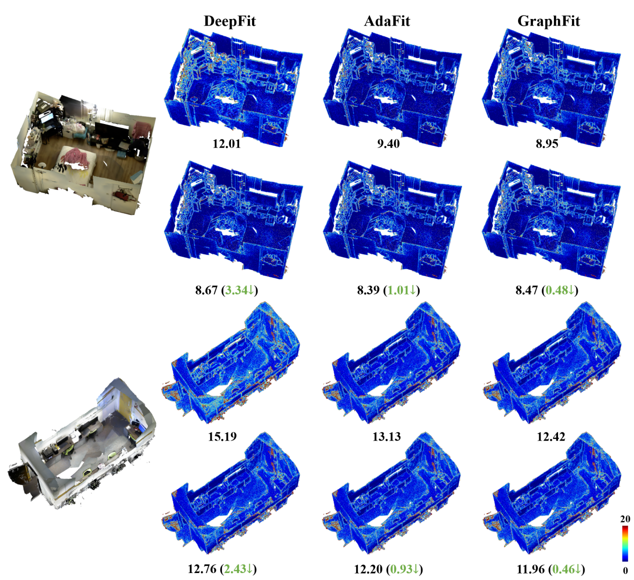


Figure 6. The error heatmap of normal estimation on sceneNN dataset. The bottom values denotes the average error. For each scene, the first line is produced by the baseline models, and the second line shows the results of applying our methods on the corresponding baseline.

Table 4. Normal angle RMSE of SOTA models with or without the proposed methods on PCPNet dataset.

Aug.	DeepFit (size = 256)				AdaFit (size = 700)				GraphFit (size = 500)			
Z-direction Trans.		✓		✓		✓		✓		✓		✓
Error Estimation			✓	✓			✓	✓			✓	✓
No Noise	6.51	6.27	5.01	4.90	5.19	4.93	4.72	4.71	4.45	4.27	4.36	4.11
Low Noise	9.21	9.10	9.09	8.91	9.05	8.94	8.81	8.75	8.74	8.79	8.71	8.66
Med Noise	16.72	16.68	16.66	16.61	16.44	16.39	16.34	16.31	16.05	16.02	16.05	16.02
High Noise	23.12	22.98	22.94	22.87	21.94	21.56	21.80	21.64	21.64	21.66	21.60	21.57
Gradient	7.31	7.17	5.69	5.52	5.90	5.63	5.59	5.51	5.22	4.98	5.06	4.83
Striped	7.92	7.73	5.85	5.70	6.01	5.89	5.62	5.48	5.48	5.10	5.18	4.89
Average	11.80	11.66	10.87	**10.75**	10.76	10.56	10.48	**10.40**	10.26	10.14	10.16	**10.01**

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

Order	1		2		3	
Baseline	✓		✓		✓	
+ Ours		✓		✓		✓
No Noise	5.13	5.03	5.16	4.91	5.17	4.87
Low Noise	9.23	8.89	9.11	9.08	9.17	9.02
Med Noise	16.76	16.71	16.71	16.73	16.71	16.72
High Noise	22.92	22.83	22.91	22.94	23.02	22.87
Gradient	5.87	5.73	5.79	5.56	6.03	5.72
Striped	6.01	5.82	5.92	5.71	6.00	5.79
Average	10.99	10.84	10.93	10.82	11.02	10.83

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

Method	Params (M)	Model size (MB)	Time (ms)	Avg. error
PCPNet [13]	21.30	85.41	-	14.56
Nesti-Net [5]	170.10	2,010.00	-	12.41
DeepFit [3]	**3.44**	**13.53**	**0.47**	11.80
AdaFit [42]	4.07	16.14	0.49	10.76
GraphFit [21]	4.16	16.38	2.07	10.26
DeepFit + Ours	3.64	14.26	0.56	10.75
AdaFit + Ours	4.28	16.85	0.58	10.40
GraphFit + Ours	4.36	17.09	2.19	**10.01**

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 later 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

[1] Pierre Alliez, David Cohen-Steiner, Yiying Tong, and Mathieu Desbrun. Voronoi-based variational reconstruction of unoriented point sets. In *Symposium on Geometry processing*, volume 7, pages 39–48, 2007. 2

[2] Nina Amenta and Marshall Bern. Surface reconstruction by voronoi filtering. In *Proceedings of the fourteenth annual symposium on Computational geometry*, pages 39–48, 1998. 2

[3] 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, 3, 5, 6, 8

[4] Yizhak Ben-Shabat, Michael Lindenbaum, and Anath Fischer. 3dmfv: Three-dimensional point cloud classification in real-time using convolutional neural networks. *IEEE Robotics and Automation Letters*, 3(4):3145–3152, 2018. 1, 3

[5] Yizhak Ben-Shabat, Michael Lindenbaum, and Anath Fischer. Nesti-net: Normal estimation for unstructured 3d point clouds using convolutional neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10112–10120, 2019. 1, 2, 3, 6, 8

[6] 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. 1, 2

[7] 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

[8] Frédéric Cazals and Marc Pouget. Estimating differential quantities using polynomial fitting of osculating jets. *Computer Aided Geometric Design*, 22(2):121–146, 2005. 1, 2, 3, 4, 6

[9] Weidong Dai, Xuejun Yan, Jingjing Wang, Di Xie, and Shiliang Pu. Mdr-mfi:multi-branch decoupled regression and multi-scale feature interaction for partial-to-partial cloud registration. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2023. 1

[10] Hang Du, Xuejun Yan, Jingjing Wang, Di Xie, and Shiliang Pu. Point cloud upsampling via cascaded refinement network. In *Asian Conference on Computer Vision*, 2022. 1

[11] Dasith de Silva Edirimuni, Xuequan Lu, Gang Li, and Antonio Robles-Kelly. Contrastive learning for joint normal estimation and point cloud filtering. *arXiv preprint arXiv:2208.06811*, 2022. 1

[12] Gaël Guennebaud and Markus Gross. Algebraic point set surfaces. In *ACM siggraph 2007 papers*, pages 23–es. 2007. 1, 2

[13] 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, 2, 3, 5, 6, 8

[14] Taisuke Hashimoto and Masaki Saito. Normal estimation for accurate 3d mesh reconstruction with point cloud model incorporating spatial structure. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition workshops*, volume 1, 2019. 1, 3

[15] Hugues Hoppe, Tony DeRose, Tom Duchamp, John McDonald, and Werner Stuetzle. Surface reconstruction from unorganized points. In *Proceedings of the 19th Annual conference on Computer Graphics and Interactive Techniques*, pages 71–78, 1992. 1, 2, 6

[16] Binh-Son Hua, Quang-Hieu Pham, Duc Thanh Nguyen, Minh-Khoi Tran, Lap-Fai Yu, and Sai-Kit Yeung. Scenenn: A scene meshes dataset with annotations. In *International Conference on 3D Vision (3DV)*, 2016. 6, 7

[17] Shengyu Huang, Zan Gojcic, Mikhail Usvyatsov, Andreas Wieser, and Konrad Schindler. Predator: Registration of 3d point clouds with low overlap. In *Proceedings of the IEEE/CVF Conference on computer vision and pattern recognition*, pages 4267–4276, 2021. 1

[18] Michael Kazhdan, Matthew Bolitho, and Hugues Hoppe. Poisson surface reconstruction. In *Proceedings of the fourth Eurographics symposium on Geometry processing*, volume 7, 2006. 1

[19] Jan Eric Lenssen, Christian Osendorfer, and Jonathan Masci. Deep iterative surface normal estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11247–11256, 2020. 2, 3, 6

[20] David Levin. The approximation power of moving least-squares. *Mathematics of computation*, 67(224):1517–1531, 1998. 1, 2

[21] Keqiang Li, Mingyang Zhao, Huaiyu Wu, Dong-Ming Yan, Zhen Shen, Fei-Yue Wang, and Gang Xiong. Graphfit: Learning multi-scale graph-convolutional representation for point cloud normal estimation. In *European Conference on Computer Vision*, pages 651–667. Springer, 2022. 1, 2, 3, 5, 6, 8

[22] Qing Li, Yu-Shen Liu, Jin-San Cheng, Cheng Wang, Yi Fang, and Zhizhong Han. Hsurf-net: Normal estimation for 3d point clouds by learning hyper surfaces. In *Advances in Neural Information Processing Systems*, 2022. 2, 3

[23] Dening Lu, Xuequan Lu, Yangxing Sun, and Jun Wang. Deep feature-preserving normal estimation for point cloud filtering. *Computer-Aided Design*, 125:102860, 2020. 1, 2, 3

[24] Xuequan Lu, Scott Schaefer, Jun Luo, Lizhuang Ma, and Ying He. Low rank matrix approximation for 3d geometry filtering. *IEEE Transactions on Visualization and Computer Graphics*, 2020. 1

[25] Baorui Ma, Zhizhong Han, Yu-Shen Liu, and Matthias Zwicker. Neural-pull: Learning signed distance functions from point clouds by learning to pull space onto surfaces. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139, 2021. 1

[26] Quentin Mérigot, Maks Ovsjanikov, and Leonidas J Guibas. Voronoi-based curvature and feature estimation from point clouds. *IEEE Transactions on Visualization and Computer Graphics*, 17(6):743–756, 2010. 2

[27] François Pomerleau, Francis Colas, Roland Siegwart, et al. A review of point cloud registration algorithms for mobile robotics. *Foundations and Trends® in Robotics*, 4(1):1–104, 2015. 1

[28] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 652–660, 2017. 1, 3, 5

[29] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In *Advances in Neural Information Processing Systems*, volume 30, 2017. 1

[30] Guocheng Qian, Yuchen Li, Houwen Peng, Jinjie Mai, Hasan Abed Al Kader Hammoud, Mohamed Elhoseiny, and Bernard Ghanem. Pointnext: Revisiting pointnet++ with improved training and scaling strategies. In *Advances in Neural Information Processing Systems*, 2022. 1

[31] Yue Qian, Junhui Hou, Sam Kwong, and Ying He. Pugeonet: A geometry-centric network for 3d point cloud upsampling. In *European Conference on Computer Vision*, pages 752–769. Springer, 2020. 1, 3

[32] Shiyao Wang, Xiuping Liu, Jian Liu, Shuhua Li, and Junjie Cao. Deep patch-based global normal orientation. *Computer-Aided Design*, page 103281, 2022. 2

[33] Yue Wang and Justin M Solomon. Deep closest point: Learning representations for point cloud registration. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 3523–3532, 2019. 1

[34] Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E Sarma, Michael M Bronstein, and Justin M Solomon. Dynamic graph cnn for learning on point clouds. *Acm Transactions On Graphics (tog)*, 38(5):1–12, 2019. 5

[35] Xuejun Yan, Hongyu Yan, Jingjing Wang, Hang Du, Zhihong Wu, Di Xie, Shiliang Pu, and Li Lu. Fbnet: Feedback network for point cloud completion. In *European Conference on Computer Vision*, pages 676–693. Springer, 2022. 5

[36] Jie Zhang, Jun-Jie Cao, Hai-Rui Zhu, Dong-Ming Yan, and Xiu-Ping Liu. Geometry guided deep surface normal estimation. *Computer-Aided Design*, 142:103119, 2022. 2

[37] Haoran Zhou, Honghua Chen, Yidan Feng, Qiong Wang, Jing Qin, Haoran Xie, Fu Lee Wang, Mingqiang Wei, and Jun Wang. Geometry and learning co-supported normal estimation for unstructured point cloud. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13238–13247, 2020. 1, 3

[38] Haoran Zhou, Honghua Chen, Yingkui Zhang, Mingqiang Wei, Haoran Xie, Jun Wang, Tong Lu, Jing Qin, and Xiao-Ping Zhang. Refine-net: Normal refinement neural network for noisy point clouds. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2022. 3

[39] Jun Zhou, Hua Huang, Bin Liu, and Xiuping Liu. Normal estimation for 3d point clouds via local plane constraint and multi-scale selection. *Computer-Aided Design*, 129:102916, 2020. 2, 3

[40] Jun Zhou, Wei Jin, Mingjie Wang, Xiuping Liu, Zhiyang Li, and Zhaobin Liu. Fast and accurate normal estimation for point clouds via patch stitching. *Computer-Aided Design*, 142:103121, 2022. 1

[41] Runsong Zhu, Di Kang, Ka-Hei Hui, Yue Qian, Xuefei Zhe, Zhen Dong, Linchao Bao, and Chi-Wing Fu. Semi-signed neural fitting for surface reconstruction from unoriented point clouds. *arXiv preprint arXiv:2206.06715*, 2022. 1

[42] Runsong Zhu, Yuan Liu, Zhen Dong, Yuan Wang, Tengping Jiang, Wenping Wang, and Bisheng Yang. Adafit: Rethinking learning-based normal estimation on point clouds. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6118–6127, 2021. 1, 2, 3, 5, 6, 7, 8