NeAF: Learning Neural Angle Fields for Point Normal Estimation

Shujuan Li[1*], Junsheng Zhou[1*], Baorui Ma[1], Yu-Shen Liu[1†], Zhizhong Han[2]

[1]School of Software, BNRist, Tsinghua University, Beijing, China
[2]Department of Computer Science, Wayne State University, Detroit, USA
lisj22, zhoujs21, mbr18 @mails.tsinghua.edu.cn, liuyushen@tsinghua.edu.cn, h312h@wayne.edu

Abstract

Normal estimation for unstructured point clouds is an important task in 3D computer vision. Current methods achieve encouraging results by mapping local patches to normal vectors or learning local surface fitting using neural networks. However, these methods are not generalized well to unseen scenarios and are sensitive to parameter settings. To resolve these issues, we propose an implicit function to learn an angle field around the normal of each point in the spherical coordinate system, which is dubbed as Neural Angle Fields (NeAF). Instead of directly predicting the normal of an input point, we predict the angle offset between the ground truth normal and a randomly sampled query normal. This strategy pushes the network to observe more diverse samples, which leads to higher prediction accuracy in a more robust manner. To predict normals from the learned angle fields at inference time, we randomly sample query vectors in a unit spherical space and take the vectors with minimal angle values as the predicted normals. To further leverage the prior learned by NeAF, we propose to refine the predicted normal vectors by minimizing the angle offsets. The experimental results with synthetic data and real scans show significant improvements over the state-of-the-art under widely used benchmarks. Project page: https://lisj575.github.io/NeAF/.

Introduction

Estimating normals for unstructured point clouds is a fundamental task in 3D computer vision. The estimated normal vectors can be leveraged in various downstream applications, such as surface reconstruction (Kazhdan, Bolitho, and Hoppe 2006), registration (Pomerleau et al. 2015), and semantic segmentation (Grilli, Menna, and Remondino 2017). Recently, the learning-based methods have achieved promising results by casting the normal estimation task into a regression problem. They resolved this problem by directly learning a mapping from local patches to normal vectors using neural networks. However, these methods fail to observe adequate samples for accurate normal estimation. This leads to poor generalization ability on unseen scenarios such as large-scale scenes.

As a remedy, state-of-the-art methods introduce to predict the normal of a point by fitting a local geometric surface. They learned point-wise weights for neighboring points to fit the surface, and then predict the normal vector from the fitted surface. However, the geometric surface settings (e.g. the constant order of the polynomial surface) are predefined during the training process, which leads to poor fitting results due to the various complexity of the local point patches. For example, when the geometric surface is more complex than the underlying point patches, it results in overfitting problems. Otherwise, the underfitting leads to over-smoothing details. As a result, the performance of surface fitting methods is dramatically limited in cases with complex topology and geometry.

To resolve these issues, we propose an implicit method to learn the Neural Angle Field (NeAF) of local patches in the spherical coordinate system. Specifically, instead of explicitly predicting the normal from the input point clouds, we design a neural network to implicitly predict the angle offset between the ground truth normal and a randomly sampled query vector. The network observes more diverse samples by learning to model the relationship between vectors from various directions and the target normal, leading to an angle field around the target normal. With a well-learned angle field, the network will have the ability to predict accurate and robust normal vectors.

At inference time, to predict normals from the learned angle fields, we randomly sample query vectors in a unit spherical space and take the vectors with minimal angle offsets as the predicted normals. To further leverage the prior learned by NeAF, we propose a novel learning scheme for refining the predicted normal vectors. Unlike existing learning-based methods that can only predict the normal via one forward pass, our proposed method can further optimize the predicted normal, which leads to more accurate normal estimation.

Our contributions are summarized as follows.

- We propose Neural Angle Field (NeAF) for point normal estimation. Unlike previous methods, our implicit function learns an angle field for each point, which implicitly predicts the angle offset of query vectors.
- Our method can conduct further optimization at infer-

[*]These authors contributed equally.
[†]Corresponding author: Yu-Shen Liu

Copyright © 2023, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.

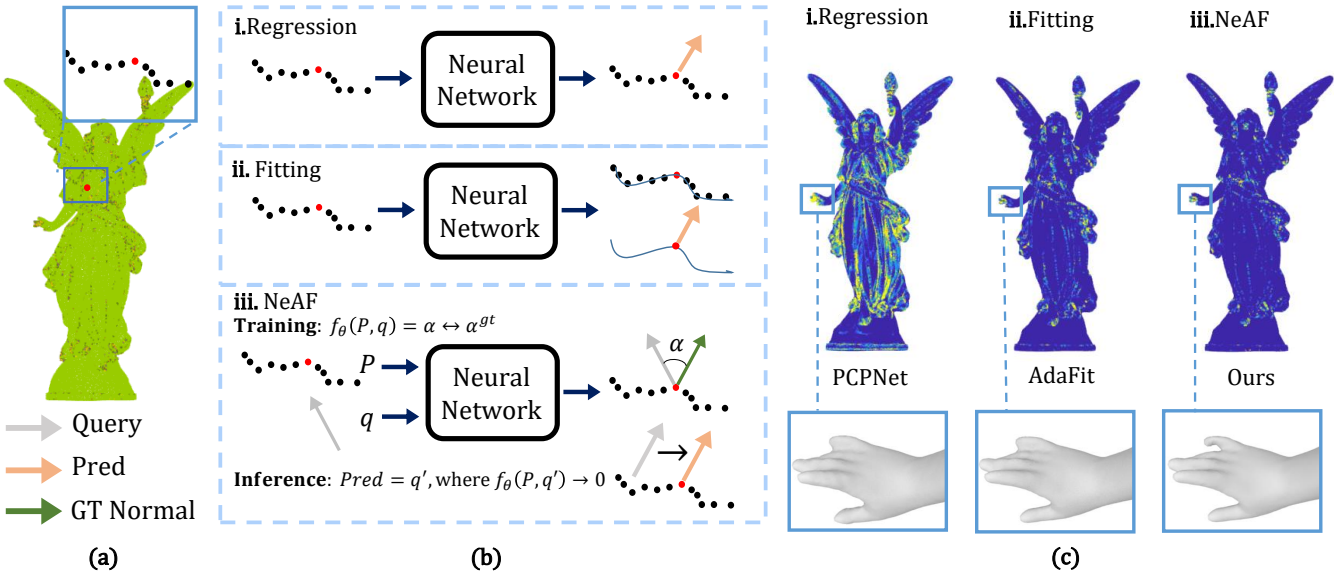

Figure 1: Comparisons between previous learning-based normal estimation approaches and our NeAF. (a) Given a point on the point cloud, we sample its K nearest neighbors as a local patch and output the normal for the point. (b) Existing learning-based normal estimation methods can be roughly divided into (i) regression-based and (ii) fitting-based methods. (i) Regression-based methods directly map the local patch to a 3D vector as the normal. (ii) Surface fitting-based methods estimate the weight for each point in the local patch and compute the normal of the surface fitted with the weighted points. (iii) Our NeAF learns the angle offsets between the random query vectors and the ground truth normal, and outputs the query vector with an angle offset zero as the normal vector. (c) The colors of the shapes indicate normal RMSE errors. The estimated normals can be used for surface reconstruction and significantly affect the quality of the reconstructed surface.

ence time to refine the predicted normals by minimizing the angle offsets for more accurate normal estimation.

- We achieve state-of-the-art results in normal estimation for synthetic data and real scans on widely used benchmarks.

Related Work

Normal Estimation

Traditionally, principal component analysis (PCA) (Hoppe et al. 1992) is a simple and efficient method for normal estimation, which is based on constructing a tangent plane from a fixed-scale local neighborhood and analyzes its normal vector. Variants such as truncated Taylor expansion (Jets) (Cazals and Pouget 2005), moving least squares (MLS) (Levin 1998) and multi-scale kernel methods (Aroudj et al. 2017) were proposed to fit more complex local surfaces. However, these methods tend to choose large scales to ensure the robustness of normal estimation. To preserve more details, the approaches (Amenta et al. 1999; Mérigot et al. 2010) were proposed based on analyzing Voronoi cells. In practice, these methods have to tune their parameters to work with different cases, which dramatically limits their application scenarios.

Recently, with the development of deep learning, learning-based methods were proposed for normal estimation. Boulch and Marlet (2016) proposed to transform the local patches of point clouds into accumulators in a 2D Hough space and estimate normals from the resulting approximate planes. However, the transformation from 3D to 2D resulted in the loss of the structural information contained in the surface. To preserve a complete local shape, Guerrero et al. (2018) proposed the PCPNet with a multi-scale point cloud normal estimation architecture using PointNet (Qi et al. 2017). Based on this method, Zhou et al. (2020) proposed a new scale selection strategy and extra constraints on the feature. In addition, Zhou et al. (2022a) proposed to introduce additional feature representations to refine the input initial normal. Li et al. (2022b) used the networks to learn hyper surfaces and obtained an improved performance.

Recent studies have achieved encouraging performance on normal estimation by predicting point-wise weights to select neighboring points softly. Lenssen et al. (2020) proposed to use a graph neural network to iteratively generate point-wise weights, and then estimate the normal by fitting a moving least squares plane. Ben-Shabat et al. (2020) used the n-order Jet of weighted points to fit local surfaces. Cao et al. (2021) used a differentiable random sample consensus module to predict normals from a latent tangent plane representation constructed from the neighboring points. To learn better point-wise weights, Zhang et al. (2022) proposed to add direct geometric weight guidance which is constructed by distances between points and the tangent plane. Zhu (2021) proposed to predict an offset to adjust the distribution of clouds and designed a cascaded scale aggrega-

tion (CSA) layer adapting to the scale of the local neighborhood, which achieved state-of-the-art results. Li et al. (2022a) proposed to learn graph convolutional feature representation. Although improvements have been made by predicting point-wise weights and fitting surfaces for normal estimation, these methods still suffer from several inherent problems such as the difficulty in determining the form of the fitted surface and the sensitivity to outliers.

Neural Implicit Representation in 3D Reconstruction

Recently, neural implicit representations have gained popularity and are widely used for 3D reconstruction (Zhou et al. 2022b; Ma, Liu, and Han 2022; Chen, Liu, and Han 2022; Ma et al. 2022; Li et al. 2022c). These representations were learned by neural networks to map input 3D coordinates to occupancy probabilities (Chen and Zhang 2019; Mescheder et al. 2019) or distance values (Michalkiewicz et al. 2019; Park et al. 2019), which implicitly represent continuous surfaces of shapes and handle complex shape topology. In addition, Neural Radiance Fields (NeRFs) (Mildenhall et al. 2020) implicitly encode the geometry and color information of the scene in the network.

Inspired by neural implicit representations, we propose to utilize the network to implicitly represent the normals of points. Unlike existing learning-based normal estimation methods that focus only on the relationship between local neighborhoods and normal vectors, we use the network to observe the difference between random vectors and the ground truth normal. This provides the network with more prior knowledge by learning a neural angle field, which helps to predict more accurate and robust normals.

Method

In this section, we present NeAF, an implicit normal estimation method for point clouds.

Implicit Angle Functions

We first define the *implicit angle function* of a given normal vector. The implicit angle function takes an arbitrary query vector in a unit spherical space and a condition $\mathcal{C}$ as input and predicts the angle offset between the query vector and the ground truth vector. The corresponding function is defined as follows,

$$f : \mathcal{C} \times \mathbb{R}^3 \rightarrow \mathbb{R}. \tag{1}$$

Our key idea is to estimate the normal of one point using the implicit angle function that is learned by a neural network. Observing that the normal vector is a local property of a surface, we leverage the local neighborhood of the center point as a condition $\mathcal{C}$ indicating the center point, together with the query vector as the input to feed the network, and the output is the angle offset between the ground truth and the query vector. Since we focus on unoriented normal estimation, the angle offset is either the difference from the positive direction or the negative direction of the ground truth, and its value range is $[0, \frac{\pi}{2}]$. The implicit angle function of point p_i can be represented as:

$$f_\theta(P_i, \boldsymbol{n}^q) = \alpha, i \in [1, N], \tag{2}$$

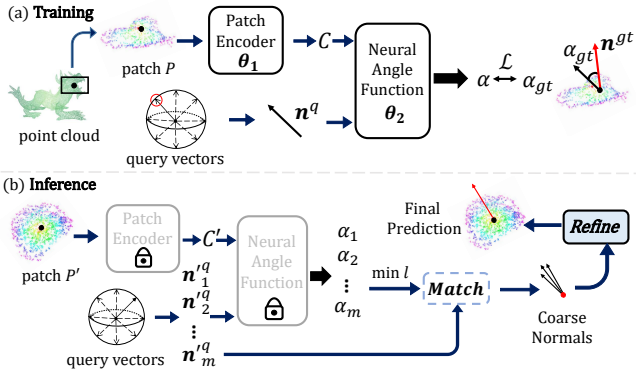

Figure 2: Overview of our NeAF. (a) During training, we randomly sample query vectors in a unit spherical space, together with the local patch as the input to the network, and predict the angle offset α between the query vector $\boldsymbol{n}^q$ and the ground truth normal $\boldsymbol{n}^{gt}$. (b) At inference time, we sample m query vectors at different directions, and then select the query vectors with the minimum l angle offsets as the coarse normals. *Match* means to find the query vectors with the same indices as the minimum l angle offsets. *Refine* means coarse normal refinement in a subsequent section.

where f_θ denotes the network parameterized by θ, $P_i \in \mathbb{R}^{k\times 3}$ is the condition of point p_i, it is the set of k nearest neighbors for the center point p_i, N is the number of points on the point cloud, $\boldsymbol{n}^q \in \mathbb{R}^3$ is the query vector, and $\alpha \in [0, \frac{\pi}{2}]$ is the angle offset of $\boldsymbol{n}^q$ relative to the target normal.

With this definition, the target normal of p_i is implicitly represented as the zero level set of the implicit angle function, which is denoted as $\boldsymbol{n}^t$, i.e. $f_\theta(P_i, \boldsymbol{n}^t) = 0$.

Training for Implicit Angle Functions

To learn the parameters θ of the neural network f_θ, we first randomly sample query vectors $\boldsymbol{n}_j^q, j = 1, ...M$ on the unit sphere. For a given point $p_i, i = 1, ...N$, we use the K-Nearest-Neighbors algorithm to sample the nearest neighbors P_i as the local patch centered at p_i. Next, the network takes the local patch P_i and query vector $\boldsymbol{n}_j^q$ as inputs and predicts the angle offsets between the ground truth normal $\boldsymbol{n}_i^{gt}$ and $\boldsymbol{n}_j^q$. We minimize the difference between the predicted angle offset $f_\theta(P_i, \boldsymbol{n}_j^q)$ and ground truth angle offset α_{ij}^{gt} by the following $L1$ loss,

$$\mathcal{L} = \frac{1}{NM} \sum_{i\in[1,N]} \sum_{j\in[1,M]} |f_\theta(P_i, \boldsymbol{n}_j^q) - \alpha_{ij}^{gt}|, \tag{3}$$

and the α_{ij}^{gt} is computed by,

$$\alpha_{ij}^{gt} = arcsin(||\boldsymbol{n}_i^{gt} \times \boldsymbol{n}_j^q||), \tag{4}$$

where $arcsin(\cdot)$ represents the arcsine function, $||\cdot||$ is the L2-norm, and $\times$ is cross product.

After convergence, the network learns an implicit angle field for each point. In Figure 3, we visualize the optimizing

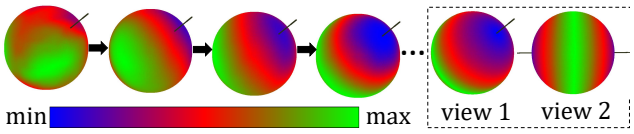

Figure 3: The visualization of learning an implicit angle field. Each point on the unit sphere represents a query vector, and the color map indicates their angle offsets to the ground truth normal represented as the black line. After convergence, an angle field centered on the ground truth normal is formed.

process of the angle field around a single point, where the well-learned angle field can predict correct angle offsets on the whole spherical surface.

The angle field implicitly represents the target vector as the zero level-set of the learned angle field. At inference time, to predict the target vector from the learned field, we introduce a gradient descent based optimization to explore the prior learned by the angle field. An intuitive implementation is to directly optimize some randomly sampled vectors which cover the whole spherical space until their angle offsets approach zero, but it increases the computational burden to optimize a large number of vectors during inference. To resolve this issue, we propose to first predict a few coarse normals and then optimize these predicted coarse normals by a gradient decent based optimization algorithm.

Inference of Angle Fields

Coarse normal prediction. To predict coarse normals at inference time, we discretize the continuous unit spherical space into sections, which are the different query directions. Then a certain density of points are sampled from the discrete sections to form a query vector set $S = \{\boldsymbol{n}_k^{q\prime} | k = 1, 2, ..., m\}$ to cover m different directions. If the query vectors in S are infinitely dense to cover the full spherical space, there is a vector $\boldsymbol{n}^o \in S$ that satisfies $f_\theta(P', \boldsymbol{n}^o) = 0$, then $\boldsymbol{n}^o$ can be viewed as the approximated direction of $\boldsymbol{n}^{gt\prime}$.

However, it is impossible to cover the full spherical space during testing since the discrete query vectors approximate the continuous spherical space and it is also time-consuming to infer highly dense query vectors. Therefore, we propose to predict a few coarse normals from the learned angle fields first.

In practice, given a test point p_i', the trained network f_θ takes the local neighborhood P_i' around p_i' and the query vectors in S as input, and predicts the angle offsets $A = \{\alpha_{ik} | k = 1, 2, ..., m\}$, that is $f_\theta(P_i', \boldsymbol{n}_k^{q\prime}) = \alpha_{ik}$. We select the query vectors $\{\boldsymbol{n}_s^c | s = 1, ..., l\}$ from S with the minimum l angle offsets as the predicted coarse normals. The coarse normals will be further refined in the following.

Coarse normal refinement. The analysis of normal prediction by the learned angle fields shows that the random sampling of discrete spherical points cannot perfectly cover the entire continuous spherical space, thus the forward pass prediction cannot make full use of the learned angle fields.

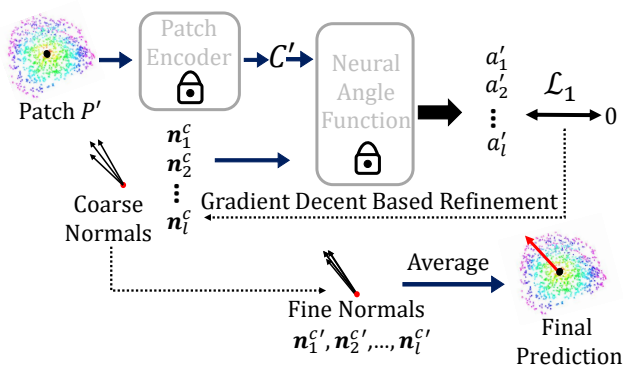

Figure 4: Coarse normal refinement. For the coarse normal $\boldsymbol{n}_s^c, s = 1...l$, we use the trained network f_θ with fixed parameters to estimate the angle offset α_s', then we update the vector $\boldsymbol{n}_s^c$ to make α_s' closer to zero. When the further optimization is done, we average all the coarse normals to output the final predicted normal.

To further improve the accuracy of the predicted normals at inference time, we propose a novel learning scheme for refining the predicted normal vectors by a gradient decent based optimization algorithm. Unlike existing learning-based methods that can only predict normals via a single forward pass during inference, our proposed method can fully leverage the prior learned by NeAF to adjust the predicted normals, and can further optimize the predicted normal at inference time, thus leading to a more accurate estimation.

Specifically, the proposed NeAF represents the relationship between the target normal and an arbitrary vector in the neural network by measuring the angle offset. Thus we can adopt further refinement given the predicted coarse normals by minimizing the angle offsets of these normals.

At inference time, we utilize the network f_θ with fixed parameters to output an angle offset α_{is}' for the coarse normal $\boldsymbol{n}_s^c$, and then update $\boldsymbol{n}_s^c$ to make α_{is}' as close as possible to zero following the formula,

$$\boldsymbol{n}_s^{c\prime} = \arg\min_{\boldsymbol{n}_s^c} \mathcal{L}_1(f_\theta(P_i', \boldsymbol{n}_s^c) - 0), \tag{5}$$

where $\mathcal{L}_1$ is the $L1$ loss, and P_i' is the local patch for the testing point p_i'.

After coarse normal refinement, $\boldsymbol{n}_s^c$ is optimized to approximate the real zero level-set of the learned angle field.

To ensure the robustness of the results, we propose to average these refined normals $\boldsymbol{n}_s^{c\prime}, s \in [1, l]$ as the final estimation $\boldsymbol{n}_{pred}$ by the following formula,

$$\boldsymbol{n}_{pred} = \frac{1}{l} \sum_{s \in [1,l]} \boldsymbol{n}_s^{c\prime}. \tag{6}$$

Since the correct normal vector direction can be easily estimated by several methods (Mullen et al. 2010; Wu et al. 2015; Huang et al. 2019; Metzer et al. 2021), we mainly focus on *unoriented normal estimation*. However, the mean of the refined normals described above may lead to a large error due to the uncertain signs. For example, both the vector with

Methods	Density		Noise				average
	Stripes	Gradients	No noise	Low Noise	Med Noise	High Noise	
Jets (Cazals and Pouget 2005)	13.39	13.13	12.23	12.84	18.33	27.68	16.29
PCA (Hoppe et al. 1992)	13.66	12.81	12.29	12.87	18.38	27.5	16.25
HoughCNN (Boulch et al. 2016)	12.47	11.02	10.23	11.62	22.66	33.39	16.90
PCPNet (Guerrero et al. 2018)	11.74	13.42	9.66	11.46	18.26	22.8	14.56
Nesti-Net (Ben-Shabat et al. 2019)	8.47	9.00	6.99	10.11	17.63	22.28	12.41
IterNet (Lenssen et al. 2020)	7.73	7.51	6.72	9.95	17.18	21.96	11.84
DeepFit (Ben-Shabat et al. 2020)	7.31	7.92	6.51	9.21	16.72	23.12	11.8
Refine-Net (Zhou et al. 2022a)	6.61	7.02	6.27	9.18	16.59	22.57	11.37
AdaFit (Zhu et al. 2021)	6.04	5.90	5.19	**9.05**	16.44	21.94	10.76
NeAF (Ours)	**4.89**	**4.88**	**4.20**	9.25	**16.35**	**21.74**	**10.22**

Table 1: Comparison of the angle RMSE with the state-of-the-art methods on PCPNet dataset.

an actual angle offset of $\hat{\alpha}$ and the one with an angle offset of $(\pi - \hat{\alpha})$ are with the same angle offset $\hat{\alpha}$, but the mean of the two normals can be far from the ground truth normal by an offset of $\pi/2$. To solve this problem, we normalize the signs of these vectors before averaging them together. We choose one of the refined vectors $\boldsymbol{n}_r^{c\prime}$ as the reference and re-direct the other vectors $\boldsymbol{n}_s^{c\prime}$, $s \in [1, l]$, according to the dot product between $\boldsymbol{n}_r^{c\prime}$ and $\boldsymbol{n}_s^{c\prime}$ below

$$\boldsymbol{n}_s^{c\prime} = sign(\boldsymbol{n}_r^{c\prime T} \boldsymbol{n}_s^{c\prime}) \boldsymbol{n}_s^{c\prime}, \tag{7}$$

where $sign$ represents the sign function with an output in $\{-1, 1\}$.

Implementation Details

For generating training data, we adopt the method proposed by Muller (1959) to randomly and uniformly sample $M = 5000$ query vectors in the unit spherical space for training, and the ground truth angle offsets are calculated by Eq. (4). As for inference data, we use the same method to sample $m = 10000$ query vectors for extracting coarse normals. During training, we randomly select 400 query vectors from the training set as a batch to train the network. At inference time, we select to extract $l = 10$ coarse normals and optimize them simultaneously in 5 epochs for coarse normal refinement. We use the same strategy as AdaFit (Zhu et al. 2021) for preprocessing local patches, including centering and normalizing.

We employ an encoder similar to AdaFit to learn features of local patches, which is mainly based on the PointNet (Qi et al. 2017) architecture, and we retain the CSA layer. The local patch size and parameter settings of our networks are also the same as AdaFit. Besides, we adopt a neural network similar to DeepSDF (Park et al. 2019) to decode the angle offsets, which is composed of 8 fully connected layers with a residual connection.

Experimental Results

Evaluation on Synthetic Dataset

Dataset and metrics. For the experiments on synthetic shapes, we adopt the PCPNet dataset provided by Guerrero et al. (2018). We use the same train/test settings and data augmentation strategies. PCPNet samples 100k points on the mesh of each shape to obtain a point cloud. The training set contains 8 shapes, and each shape includes a noise-free point cloud and three point clouds containing Gaussian noise with a standard deviation of 0.125% (Low), 0.65% (Med) and 1.2% (High) of the length of the bounding box diagonal of the shape. In addition to the three noise variants, two additional point clouds with varying densities (Stripes and Gradients) are added to the test set. For training, we use the Adam optimizer with an initial learning rate of 1×10^{-3}, and adopt a cosine learning rate decay strategy with warm-up. The model is trained on 2 GTX 1080Ti. In coarse normal refinement, we use the Adam optimizer with an initial learning rate of 0.005. We use the angle root mean square error (RMSE) between the predicted normal and the ground truth normal as a quantification metric, and compute the final result with 5000 points subset sampled by PCPNet.

Comparisons. We make a comparison with traditional normal estimation methods, including PCA (Hoppe et al. 1992), Jets (Cazals and Pouget 2005), and learning-based methods HoughCNN (Boulch et al. 2016), PCPNet (Guerrero et al. 2018), Nesti-Net (Ben-Shabat et al. 2019), IterNet (Lenssen et al. 2020), DeepFit (Ben-Shabat et al. 2020), Refine-Net (Zhou et al. 2022a), AdaFit (Zhu et al. 2021). A comparison of the angle RMSEs is shown in Table 1. The results show that our method significantly outperforms both the traditional methods and the state-of-the-art learning-based methods. Especially on the point clouds with density variations, the latest fitting-based method Adafit fails dramatically due to the point sparsity. This leads to a large discrepancy between the explicitly fitted surface and the underlying surface, while our method is not affected by point density and can make an accurate estimation.

We perform a visual comparison with the PCPNet (Guerrero et al. 2018), DeepFit (Ben-Shabat et al. 2020), RefineNet (Zhou et al. 2022a), and AdaFit (Zhu et al. 2021) in Figure 5, where the numbers represent the angle RMSE between the predicted normals and the real normals. The color of the shape indicates errors, and the closer to yellow the larger the error, the closer to the blue the smaller the error. The results show that our method outperforms others with more details, and can better handle complex regions such as sharp corners and sharp edges.

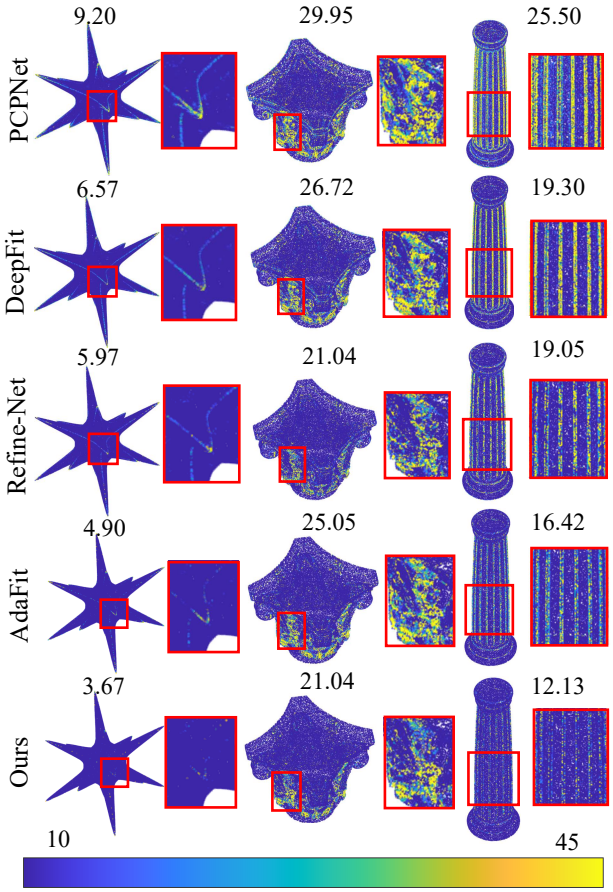

Figure 5: Errors of normal estimation on the PCPNet dataset.

Evaluation on Real Scanned Dataset

We employ the model pretrained on PCPNet dataset to report our results on the scanned dataset.

SceneNN dataset. The SceneNN (Hua et al. 2016) dataset provides indoor scenes in the form of reconstructed meshes. We randomly sample 1 million points on the mesh of the shape to obtain a point cloud, and calculate the normal of each point. We randomly select 40% of all points to calculate RMSE. The results of the baseline methods are obtained by testing on the dataset using the models provided by (Ben-Shabat et al. 2020) and (Zhu et al. 2021) under the same experimental conditions. The angle RMSE comparison between NeAF, AdaFit (Zhu et al. 2021) and DeepFit (Ben-Shabat et al. 2020) is given in Table 2, and the qualitative results are shown in Figure 6. The numerical and visual comparisons show that NeAF achieves the best performance.

Semantic3D dataset. The Semantic3D (Hackel et al. 2017) dataset provides 30 non-overlapping outdoor scenes acquired with the Terrestrial Laser Scanner in the form of point clouds. This dataset does not provide reconstructed meshes, so the ground truth normals are not available, and we mainly report visual comparisons on this dataset. The comparison between NeAF and baseline methods is shown in Figure 7. The results show that the proposed method can estimate normals at sharp edges more accurately than baseline methods.

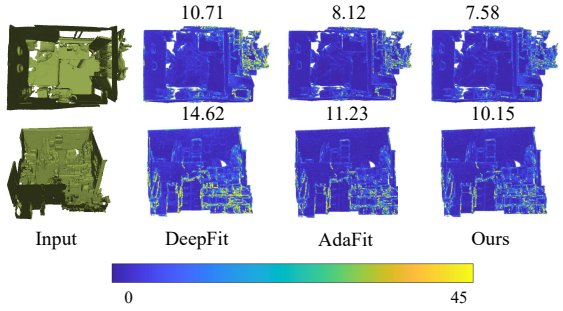

Figure 6: Error maps of estimated normals on the SceneNN dataset.

	NeAF	AdaFit	DeepFit
RMSE	**9.50**	10.19	12.56

Table 2: Normal RMSE of NeAF, AdaFit (Zhu et al. 2021) and DeepFit (Ben-Shabat et al. 2020) on the SceneNN dataset.

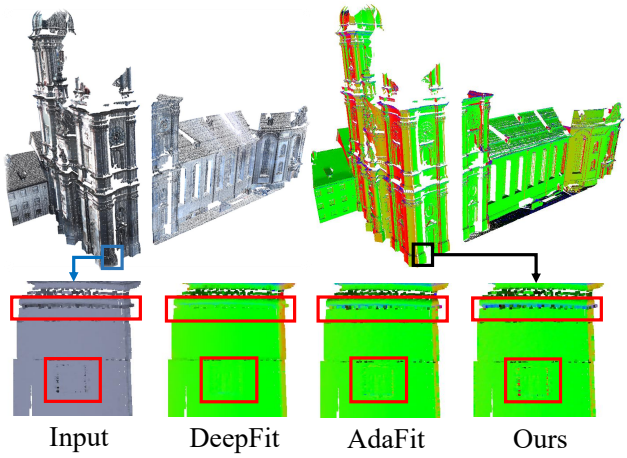

Figure 7: Estimated normals on the semantic3D dataset. The point normal vectors are mapped to RGB colors.

Surface Reconstruction

As an important property of local surfaces, normals are often used as the input for 3D surface reconstruction tasks, and accurate normal vectors play a key role in Poisson reconstruction (Kazhdan, Bolitho, and Hoppe 2006). We use the estimated normals for surface reconstruction, and the comparison with the baseline methods is shown in Figure 8. The results show that the normals estimated by NeAF are more accurate, which helps to reconstruct surfaces with higher accuracy.

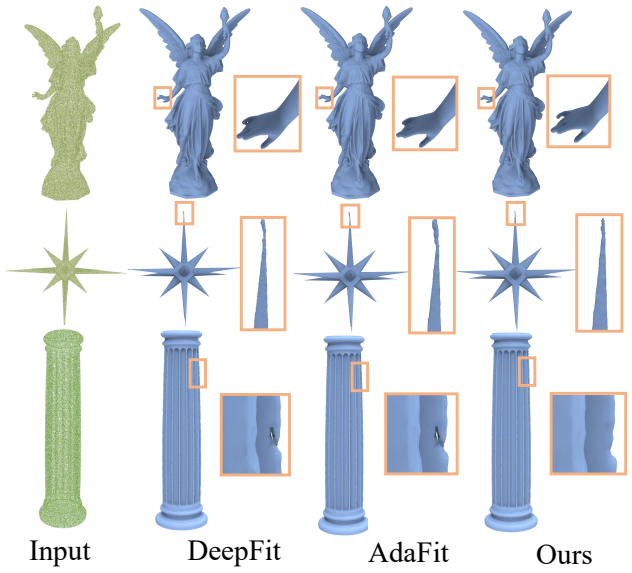

Figure 8: The comparison of the Poisson surface reconstruction using the estimated normals from NeAF, DeepFit (Ben-Shabat et al. 2020), and AdaFit (Zhu et al. 2021).

	No CNP	No CNR	Min	NeAF
Stripes	37.23	5.05	4.94	**4.89**
Gradients	37.03	5.04	4.93	**4.88**
No Noise	37.11	4.41	4.26	**4.20**
Low Noise	37.97	9.34	9.27	**9.25**
Med Noise	40.10	16.40	16.36	**16.35**
High Noise	41.92	21.76	**21.73**	21.74
Average	38.56	10.33	10.25	**10.22**

Table 3: Effect of framework design.

Ablation Studies

Framework design. We demonstrate the effectiveness of each design of our framework in Table 3. We first skip predicting the coarse normals and instead directly refine the random vectors the same number of times. As shown by "No CNP" , the performance degenerates dramatically. We also predict normals without coarse normal refinement, and the result demonstrates that the coarse normal refinement can effectively improve the accuracy of the predicted normals in all settings as shown by "No CNR". We replace averaging coarse normals with selecting the coarse normals with the minimum angle offsets after refinement as shown by "Min" and find a drop in performance.

Density of query vectors. We explore the effect of the sampling number M of query vectors on the angle field learned by f_θ. We report the performance of different $M = [2.5k, 5k, 7.5k, 10k]$ in Table 4. The query vector set that is too sparse ("2.5k") cannot provide enough sample vectors for the network to learn the angle field, while the sets that are too dense ("7.5k", "10k") make the implicit angle function more complicated and make it difficult for the network to learn the correct angle offsets. We found "5k" is a proper trade-off.

M	2.5k	5k	7.5k	10k
Stripes	4.94	**4.89**	4.95	4.96
Gradients	4.98	**4.88**	4.98	5.03
No Noise	4.29	**4.20**	4.24	4.37
Low Noise	9.17	9.25	9.19	**9.16**
Med Noise	16.41	**16.35**	16.49	16.41
High Noise	21.83	21.74	21.81	**21.69**
Average	10.27	**10.22**	10.28	10.27

Table 4: Effect of query vector number M.

l	1	5	10	20	50
Stripes	4.95	4.90	4.89	4.89	4.89
Gradients	4.93	4.90	4.88	4.88	4.88
No Noise	4.26	4.21	4.20	4.21	4.21
Low Noise	9.28	9.25	9.25	9.25	9.25
Med Noise	16.37	16.35	16.35	16.35	16.34
High Noise	21.76	21.74	21.74	21.72	21.72
Average	10.26	10.23	**10.22**	10.22	10.22

Table 5: Effect of coarse normal number l.

Number of coarse normals. In Table 5, we conduct experiments on the PCPNet dataset to explore how the coarse normal number l affects coarse normal refinement at the inference time. We report the performance of different $l = [1, 5, 10, 20, 50]$, where we find that optimizing a single coarse normal leads to the degradation of results, and the best accuracy is achieved for the first time with 10 coarse normals. Optimizing more coarse normals requires longer time and more memory without improving the accuracy.

Conclusion

In this paper, we proposed NeAF to estimate point normals implicitly. We randomly sample the query vectors in a unit spherical space, estimate their angle offsets to ground truth normals, and output the query vector with the smallest angle offset as the estimated normal. To fully leverage the prior learned by NeAF, we refine the predicted normal vector by minimizing the estimated angle offset for more accurate normal estimation. NeAF achieves state-of-the-art performance on the synthetic dataset PCPNet and exhibits good generalization on real scans in SceneNN and Semantic3D. Furthermore, the promising results in the surface reconstruction task with normals estimated by NeAF justify our effectiveness in real applications.

Acknowledgments

The corresponding author is Yu-Shen Liu. This work was supported by National Key R&D Program of China (2022YFC3800600, 2020YFF0304100), the National Natural Science Foundation of China (62272263, 62072268),

and in part by Tsinghua-Kuaishou Institute of Future Media Data.

References

Amenta; et al. 1999. Surface reconstruction by Voronoi filtering. *Discrete & Computational Geometry*, 22(4): 481–504.

Aroudj, S.; Seemann, P.; Langguth, F.; Guthe, S.; and Goesele, M. 2017. Visibility-consistent thin surface reconstruction using multi-scale kernels. *ACM Transactions on Graphics (TOG)*, 36(6): 1–13.

Ben-Shabat, Y.; et al. 2019. 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*, 10112–10120.

Ben-Shabat, Y.; et al. 2020. DeepFit: 3D surface fitting via neural network weighted least squares. In *European Conference on Computer Vision*, 20–34. Springer.

Boulch, A.; et al. 2016. Deep learning for robust normal estimation in unstructured point clouds. In *Computer Graphics Forum*, volume 35, 281–290. Wiley Online Library.

Cao, J.; Zhu, H.; Bai, Y.; Zhou, J.; Pan, J.; and Su, Z. 2021. Latent tangent space representation for normal estimation. *IEEE Transactions on Industrial Electronics*, 69(1): 921–929.

Cazals, F.; and Pouget, M. 2005. Estimating differential quantities using polynomial fitting of osculating jets. *Computer Aided Geometric Design*, 22(2): 121–146.

Chen, C.; Liu, Y.-S.; and Han, Z. 2022. Latent partition implicit with surface codes for 3D representation. In *European Conference on Computer Vision*, 322–343. Springer.

Chen, Z.; and Zhang, H. 2019. Learning implicit fields for generative shape modeling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 5939–5948.

Grilli, E.; Menna, F.; and Remondino, F. 2017. A review of point clouds segmentation and classification algorithms. *The International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences*, 42: 339.

Guerrero, P.; et al. 2018. PCPNet: Learning Local Shape Properties from Raw Point Clouds. *Computer Graphics Forum*, 37(2): 75–85.

Hackel, T.; Savinov, N.; Ladicky, L.; Wegner, J. D.; Schindler, K.; and Pollefeys, M. 2017. SEMANTIC3D.NET: A new large-scale point cloud classification benchmark. In *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, volume IV-1-W1, 91–98.

Hoppe, H.; DeRose, T.; Duchamp, T.; McDonald, J.; and Stuetzle, W. 1992. Surface reconstruction from unorganized points. In *Proceedings of the 19th annual conference on Computer graphics and interactive techniques*, 71–78.

Hua, B.-S.; Pham, Q.-H.; Nguyen, D. T.; Tran, M.-K.; Yu, L.-F.; and Yeung, S.-K. 2016. Scenenn: A scene meshes dataset with annotations. In *2016 fourth international conference on 3D vision (3DV)*, 92–101. Ieee.

Huang, X.; Liang, Z.; Zhou, X.; Xie, Y.; Guibas, L. J.; and Huang, Q. 2019. Learning transformation synchronization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 8082–8091.

Kazhdan, M.; Bolitho, M.; and Hoppe, H. 2006. Poisson surface reconstruction. In *Proceedings of the fourth Eurographics symposium on Geometry processing*, volume 7, 61–70.

Lenssen, J. E.; et al. 2020. Deep iterative surface normal estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 11247–11256.

Levin, D. 1998. The approximation power of moving least-squares. *Mathematics of Computation*, 67(224): 1517–1531.

Li, K.; Zhao, M.; Wu, H.; Yan, D.-M.; Shen, Z.; Wang, F.-Y.; and Xiong, G. 2022a. GraphFit: Learning Multi-scale Graph-Convolutional Representation for Point Cloud Normal Estimation. In *European Conference on Computer Vision*, 651–667. Springer.

Li, Q.; Liu, Y.-S.; Cheng, J.-S.; Wang, C.; Fang, Y.; and Han, Z. 2022b. HSurf-Net: Normal Estimation for 3D Point Clouds by Learning Hyper Surfaces. *Advances in Neural Information Processing Systems (NeurIPS)*.

Li, T.; Wen, X.; Liu, Y.-S.; Su, H.; and Han, Z. 2022c. Learning deep implicit functions for 3D shapes with dynamic code clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 12840–12850.

Ma, B.; Liu, Y.-S.; and Han, Z. 2022. Reconstructing surfaces for sparse point clouds with on-surface priors. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 6315–6325.

Ma, B.; Liu, Y.-S.; Zwicker, M.; and Han, Z. 2022. Surface reconstruction from point clouds by learning predictive context priors. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 6326–6337.

Mérigot; et al. 2010. Voronoi-based curvature and feature estimation from point clouds. *IEEE Transactions on Visualization and Computer Graphics*, 17(6): 743–756.

Mescheder, L.; Oechsle, M.; Niemeyer, M.; Nowozin, S.; and Geiger, A. 2019. Occupancy networks: Learning 3D reconstruction in function space. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 4460–4470.

Metzer, G.; Hanocka, R.; Zorin, D.; Giryes, R.; Panozzo, D.; and Cohen-Or, D. 2021. Orienting point clouds with dipole propagation. *ACM Transactions on Graphics (TOG)*, 40(4): 1–14.

Michalkiewicz, M.; Pontes, J. K.; Jack, D.; Baktashmotlagh, M.; and Eriksson, A. 2019. Implicit surface representations as layers in neural networks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 4743–4752.

Mildenhall, B.; Srinivasan, P. P.; Tancik, M.; Barron, J. T.; Ramamoorthi, R.; and Ng, R. 2020. NeRF: Representing scenes as neural radiance fields for view synthesis. In *European Conference on Computer Vision*, 405–421. Springer.

Mullen, P.; De Goes, F.; Desbrun, M.; Cohen-Steiner, D.; and Alliez, P. 2010. Signing the unsigned: Robust surface reconstruction from raw pointsets. In *Computer Graphics Forum*, volume 29, 1733–1741. Wiley Online Library.

Muller, M. E. 1959. A note on a method for generating points uniformly on n-dimensional spheres. *Communications of the ACM*, 2(4): 19–20.

Park, J. J.; Florence, P.; Straub, J.; Newcombe, R.; and Lovegrove, S. 2019. DeepSDF: Learning continuous signed distance functions for shape representation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 165–174.

Pomerleau, F.; Colas, F.; Siegwart, R.; et al. 2015. A review of point cloud registration algorithms for mobile robotics. *Foundations and Trends® in Robotics*, 4(1): 1–104.

Qi, C. R.; et al. 2017. PointNet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 652–660.

Wu, S.; Huang, H.; Gong, M.; Zwicker, M.; and Cohen-Or, D. 2015. Deep points consolidation. *ACM Transactions on Graphics (ToG)*, 34(6): 1–13.

Zhang, J.; Cao, J.-J.; Zhu, H.-R.; Yan, D.-M.; and Liu, X.-P. 2022. Geometry Guided Deep Surface Normal Estimation. *Computer-Aided Design*, 142: 103119.

Zhou, H.; Chen, H.; Zhang, Y.; Wei, M.; Xie, H.; Wang, J.; Lu, T.; Qin, J.; and Zhang, X.-P. 2022a. Refine-Net: Normal Refinement Neural Network for Noisy Point Clouds. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, 1–1.

Zhou, J.; Ma, B.; Yu-Shen, L.; Yi, F.; and Zhizhong, H. 2022b. Learning Consistency-Aware Unsigned Distance Functions Progressively from Raw Point Clouds. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Zhou, J.; et al. 2020. Normal estimation for 3D point clouds via local plane constraint and multi-scale selection. *Computer-Aided Design*, 129: 102916.

Zhu, R.; Liu, Y.; Dong, Z.; Wang, Y.; Jiang, T.; Wang, W.; and Yang, B. 2021. AdaFit: Rethinking Learning-based Normal Estimation on Point Clouds. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 6118–6127.