SHS-Net: Learning Signed Hyper Surfaces for Oriented Normal Estimation of Point Clouds

Qing Li[1] Huifang Feng[2] Kanle Shi[3] Yue Gao[1] Yi Fang[4]
Yu-Shen Liu[1] * Zhizhong Han[5]
[1]School of Software, BNRist, Tsinghua University, Beijing, China
[2]School of Informatics, Xiamen University, Xiamen, China [3]Kuaishou Technology, Beijing, China
[4]Center for Artificial Intelligence and Robotics, New York University Abu Dhabi, Abu Dhabi, UAE
[5]Department of Computer Science, Wayne State University, Detroit, USA
{leoqli, gaoyue, liuyushen}@tsinghua.edu.cn fenghuifang@stu.xmu.edu.cn
shikanle@kuaishou.com yfang@nyu.edu h312h@wayne.edu

Abstract

We propose a novel method called SHS-Net for oriented normal estimation of point clouds by learning signed hyper surfaces, which can accurately predict normals with global consistent orientation from various point clouds. Almost all existing methods estimate oriented normals through a two-stage pipeline, i.e., unoriented normal estimation and normal orientation, and each step is implemented by a separate algorithm. However, previous methods are sensitive to parameter settings, resulting in poor results from point clouds with noise, density variations and complex geometries. In this work, we introduce signed hyper surfaces (SHS), which are parameterized by multi-layer perceptron (MLP) layers, to learn to estimate oriented normals from point clouds in an end-to-end manner. The signed hyper surfaces are implicitly learned in a high-dimensional feature space where the local and global information is aggregated. Specifically, we introduce a patch encoding module and a shape encoding module to encode a 3D point cloud into a local latent code and a global latent code, respectively. Then, an attention-weighted normal prediction module is proposed as a decoder, which takes the local and global latent codes as input to predict oriented normals. Experimental results show that our SHS-Net outperforms the state-of-the-art methods in both unoriented and oriented normal estimation on the widely used benchmarks. The code, data and pretrained models are available at https://github.com/LeoQLi/SHS-Net.

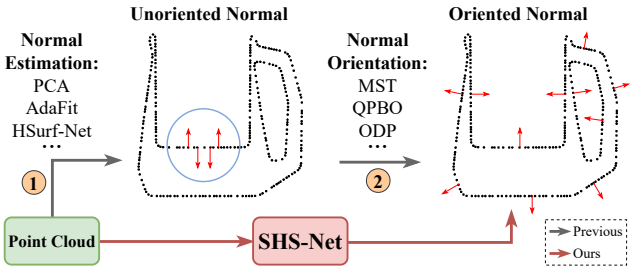

Figure 1. We propose SHS-Net to estimate oriented normals directly from point clouds. In contrast, previous studies usually achieve this process through a two-stage paradigm using different algorithms, *i.e.*, (1) unoriented normal estimation (*e.g.*, PCA [19], AdaFit [59] and HSurf-Net [32]) and (2) normal orientation (*e.g.*, MST [19], QPBO [45] and ODP [38]).

1. Introduction

In computer vision and graphics, estimating normals for point clouds is a prerequisite for many techniques. As an important geometric property of point clouds, normals with consistent orientation, *i.e.*, *oriented normals*, clearly reveal the geometric structures and make significant contributions in downstream applications, such as rendering and surface reconstruction [24–26]. Generally, the estimation of oriented normals requires a two-stage paradigm (see Fig. 1): (1) the unoriented normal estimation from the local neighbors of the query point, (2) the normal orientation to make the normal directions to be globally consistent, *e.g.*, facing outward of the surface. While unoriented normals can be estimated by plane or surface fitting of the local neighborhood, determining whether the normals are facing outward or inward is ambiguous. In recent years, many excellent algorithms [5, 29, 31, 32, 59] have been proposed for unoriented normal estimation, while there are few methods that

*The corresponding author is Yu-Shen Liu. This work was supported by National Key R&D Program of China (2022YFC3800600), the National Natural Science Foundation of China (62272263, 62072268), and in part by Tsinghua-Kuaishou Institute of Future Media Data.

have reliable performance for normal orientation or directly estimating oriented normals. Estimating oriented normals from point clouds with noise, density variations, and complex geometries in an end-to-end manner is still a challenge.

The classic normal orientation methods rely on simple greedy propagation, which selects a seed point as the start and diffuses its normal orientation to the adjacent points via a minimum spanning tree (MST) [19, 27]. These methods are limited by error accumulation, where an incorrect orientation may degenerate all subsequent steps during the iterative propagation. Furthermore, they heavily rely on a smooth and clean assumption, which makes them easily fail in the presence of sharp edges or corners, density variations and noise. Meanwhile, their accuracy is sensitive to the neighborhood size of propagation. For example, a large size is usually used to smooth out outliers and noise, but can also erroneously include nearby surfaces. Considering that local information is usually not sufficient to guarantee robust orientation, some improved methods [22, 38, 45, 46, 49, 53] try to formulate the propagation process as a global energy optimization by introducing various constraints. Since their constraints are mainly derived from local consistency, the defects are inevitably inherited, and they also suffer from cumulative errors. Moreover, their data-specific parameters are difficult to generalize to new input types and topologies.

Different from the propagation-based methods, which only consider the adjacent orientation, the volume-based approaches exploit volumetric representation, such as signed distance functions [35, 40] and variational formulations [2, 21, 48]. They aim to divide the space into interior/exterior and determine whether point normals are facing inward or outward. Despite improvements in accuracy and robustness, these methods cannot scale to large point clouds due to their computational complexity. In general, propagation-based methods have difficulty with sharp features, while volume-based methods have difficulty with open surfaces. Furthermore, the above-mentioned methods are usually complex and require a two-stage operation, their performance heavily depends on the parameter tuning in each separated stage. Recently, several learning-based methods [17, 18, 50] have been proposed to deliver oriented normals from point clouds and have exhibited promising performance. Since they focus on learning an accurate local feature descriptor and do not fully explore the relationship between the surface normal orientation and the underlying surface, their performance cannot be guaranteed across different noise levels and geometric structures.

In this work, we propose to estimate oriented normals from point clouds by implicitly learning *signed hyper surfaces*, which are represented by MLP layers to interpret the geometric property in a high-dimensional feature space. We learn this new geometry representation from both local and global shape properties to directly estimate normals with consistent orientation in an end-to-end manner. The insight of our method is that determining a globally consistent normal orientation should require global context to eliminate the ambiguity from local since orientation is not a local property. We evaluate our method by conducting a series of qualitative and quantitative experiments on a range of point clouds with different sampling densities, noise levels, thin and sharp structures.

Our main contributions can be summarized as follows.

- We introduce a new technique to represent point cloud geometric properties as signed hyper surfaces in a high-dimensional feature space.
- We show that the signed hyper surfaces can be used to estimate normals with consistent orientations directly from point clouds, rather than through a two-stage paradigm.
- We experimentally demonstrate that our method is able to estimate normals with high accuracy and achieves the state-of-the-art results in both unoriented and oriented normal estimation.

2. Related Work

Unoriented Normal Estimation. Over the past few decades, many algorithms have been proposed for point cloud normal estimation, such as the classic Principle Component Analysis (PCA) [19] and its improvements [1, 20, 28, 39, 43], Voronoi-based paradigms [2, 3, 13, 36], and variants based on complex surface [4, 10, 16, 30, 41]. These methods are usually sensitive to noise and have limited accuracy even with heavy fine-tuned parameters. More recently, learning-based methods have been proposed to improve performance in this area and can be mainly divided into two categories: regression-based and surface fitting-based. The regression-based methods try to directly predict normals from structured data [8, 34, 44] or raw point clouds [6, 17, 18, 32, 33, 55–57] in a data-driven manner. Among them, HSurf-Net [32] achieves good performance by learning hyper surfaces from local patches, but the learned surfaces have no sign and cannot determine the normal orientation. The surface fitting-based methods integrate the traditional surface fitting techniques, such as plane fitting [9,29] and jet fitting [5,31,54,58,59], into the end of the learning pipeline. They usually carefully design a network to predict point-wise weights, and then use a weighted surface formulation to solve the fitted surface normal. The normals estimated by the above methods randomly face both sides of the surface and cannot be used in many downstream applications without normal orientation.

Consistent Normal Orientation. To make the unoriented normals have consistent orientations, early approaches mainly focus on local consistency and use the orientation propagation strategy upon a minimum spanning tree (MST) to let the adjacent points have the same orientations, such

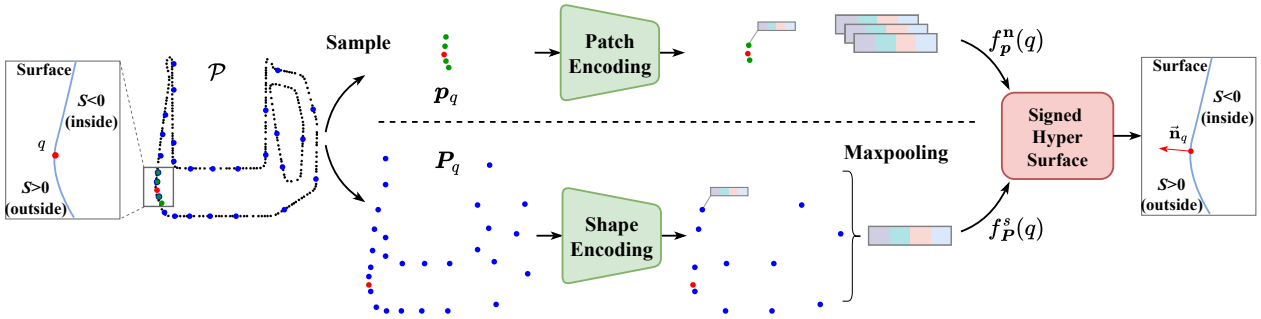

Figure 2. The learning pipeline of the signed hyper surfaces for oriented normal estimation.

as the pioneering work of [19] and its improved methods [22, 45, 46, 49, 53]. These methods have many limitations in real applications as we introduced earlier. ODP [38] aims to achieve global consistency by introducing a dipole propagation strategy across the partitioned patches, but its robustness may suffer from the patch partition. On the contrary, some alternative approaches propose to solve the consistent normal orientation through volumetric representation. They are usually developed for reconstructing surfaces from unoriented points by various techniques, such as signed distance functions [35, 40], variational formulations [2, 21, 48], visibility [11, 23], isovalue constraints [51] and active contours [52]. Recently, a few works [17, 18, 50] explore to predict oriented normals through end-to-end deep networks. These methods focus on learning a general mapping from point clouds to normals and neglect the underlying surface distribution for normal orientation, leading to a sub-optimal solution.

3. Preliminary

In mathematics, an explicit representation in Euclidean space expresses the z coordinate of a point p in terms of the x and y, *i.e.*, $z = f(x, y)$. Such a surface is called an explicit surface, also called a height field. Another symmetric representation is $F(x, y, z) = 0$, where F implicitly defines a locus called an implicit surface, also called a scalar field [7]. The implicit surface is a zero iso-surface of F, *i.e.*, the point set $\{p \in \mathbb{R}^3 : F(p) = 0\}$ is a surface implicitly defined by F. The explicit surface is usually used in surface fitting-based normal estimation, such as jet fitting [10], while the implicit surface is widely used in surface reconstruction. Generally, an explicit surface, *i.e.*, $z = f(x, y)$, can always be rewritten as an implicit surface, *i.e.*, $F(x, y, z) = z - f(x, y) = 0$. These two surface representations have the same tangent plane at a given point, where the normal is defined. See supplementary for details.

Explicit Surface Fitting. We employ the widely used n-jet surface model [10] to briefly review the explicit surface fitting for normal estimation. It represents the surface by a polynomial function $J_n : \mathbb{R}^2 \to \mathbb{R}$, which maps a coordinate

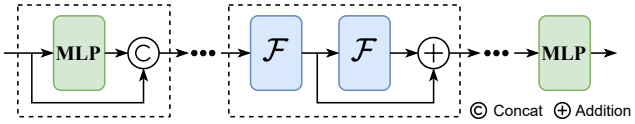

Figure 3. Feature encoding network in patch and shape encoding.

(x, y) to its height z in the tangent space by

$$z = J_{\alpha,n}(x, y) = \sum_{k=0}^{n} \sum_{j=0}^{k} \alpha_{k-j,j} x^{k-j} y^j, \tag{1}$$

where α is the coefficient vector that defines the surface function. In order to find the optimal solution, the least squares approximation strategy is usually adopted to minimize the sum of the square errors between the (ground-truth) height and the jet value over a point set $\{p_i\}_{i=1}^N$,

$$J^*_{\alpha,n} = \underset{\alpha}{\operatorname{argmin}} \sum_{i=1}^{N} \|z_i - J_{\alpha,n}(x_i, y_i)\|^2. \tag{2}$$

If $\alpha = (\alpha_{0,0}, \alpha_{1,0}, \alpha_{0,1}, \cdots, \alpha_{0,n})$ is solved, then the normal at point p on the fitted surface is computed by

$$\mathbf{n}_p = h(\alpha) = (-\alpha_{1,0}, -\alpha_{0,1}, 1)/\sqrt{1 + \alpha_{1,0}^2 + \alpha_{0,1}^2}\ . \tag{3}$$

Implicit Surface Learning. In recent years, many learning-based approaches have been proposed to represent surfaces by implicit functions, such as signed distance function (SDF) [42] and occupancy function [37]. The signed (or oriented) distance function is the shortest distance of a given point $p = (x_0, y_0, z_0)$ to the closest surface $\boldsymbol{S}$ in a metric space, with the sign determined by whether the point is inside ($F(p) < 0$) or outside ($F(p) > 0$) of the surface. The underlying surface is implicitly represented by the iso-surface of $F(p) = 0$. In the surface reconstruction task, a deep network is usually adopted to encode a 3D shape into a latent code, which is fed into a decoder together with query points to predict signed distances. If an implicit surface function is continuous and differentiable, the formula of tangent plane at a regular point p (gradient is non-null) is $F_x(p)(x - x_0) + F_y(p)(y - y_0) + F_z(p)(z - z_0) = 0$ and its normal (*i.e.*, perpendicular) is $\mathbf{n}_p = \nabla F(p)/\|\nabla F(p)\|$.

4. Method

As shown in Fig. 2, we propose to implicitly learn signed hyper surfaces in the feature space for estimating oriented normals. In the following sections, we first introduce the representation of signed hyper surfaces by combining the characteristics of the above two surface representations. Then, we design an attention-weighted normal prediction module to solve the oriented normals of query points from signed hyper surfaces. Finally, we introduce how to learn this new surface representation from patch encoding and shape encoding using our designed loss functions.

4.1. Signed Hyper Surface

We formulate the surface function $F(x, y, z) = z - f(x, y) = 0$ as a more general format $F(z_1, z_2) = z_1 - z_2 = 0$. Similarly, the signed hyper surface is implicitly learned by taking the latent encodings of point clouds as inputs and outputting an approximation of the surface in feature space,

$$f_{\boldsymbol{S}}(\mathcal{X}) \approx \mathcal{E}_\theta(\mathcal{X}|z_1, z_2),\ z_1 = e_\varphi(\boldsymbol{P}_\mathcal{X}^1),\ z_2 = e_\psi(\boldsymbol{P}_\mathcal{X}^2), \quad (4)$$

where $\mathcal{E}$ is implemented by a neural network with parameter θ that is conditioned on two latent vectors $z_1, z_2 \in \mathbb{R}^c$, which are extracted from point clouds by encoders e_φ and e_ψ, respectively. $\boldsymbol{P}_\mathcal{X}^1$ and $\boldsymbol{P}_\mathcal{X}^2$ are subsample sets of the raw point cloud $\mathcal{P}$, *e.g.*, point patches around a given point $\mathcal{X}$.

Similar to existing unoriented normal estimation methods [5, 17, 32, 59], we use a local patch $\boldsymbol{p}_q$ to capture the local geometry for accurately describing the surface pattern around a query point q,

$$f_{\boldsymbol{p}}^{\mathbf{n}}(q) = \mathcal{E}_\theta^{\mathbf{n}}(q|z_q^{\mathbf{n}}),\ z_q^{\mathbf{n}} = e_\varphi(\boldsymbol{p}_q). \quad (5)$$

Since the interior/exterior of a surface cannot be determined reliably from a local patch, we take a global subsample set $\boldsymbol{P}_q$ from the point cloud $\mathcal{P}$ to provide additional information to estimate the sign at point q,

$$f_{\boldsymbol{P}}^{s}(q) = \mathrm{sgn}\big(g^s(q)\big) = \mathrm{sgn}\big(\mathcal{E}_\theta^s(q|z_q^s)\big),\ z_q^s = e_\psi(\boldsymbol{P}_q), \quad (6)$$

where $\mathrm{sgn}(\cdot)$ is signum function, $g^s(q)$ denotes logit of the probability that q has a positive sign. Thus, the signed hyper surface function at point q is formulated as

$$f_{\boldsymbol{S}}(q) = f_{\boldsymbol{p}}^{\mathbf{n}}(q) \cdot f_{\boldsymbol{P}}^{s}(q) = \mathcal{E}_\theta^{\mathbf{n},s}(q|z_q^{\mathbf{n}}, z_q^s). \quad (7)$$

Different from the surface reconstruction task that learns SDF by representing a surface as the zero-set of the SDF, we do not learn a distance field of points with respect to the underlying surface.

4.2. Oriented Normal Estimation

To simplify notations, we denote $\mathcal{E}_\theta^{\mathbf{n},s}(q|z_q^{\mathbf{n}}, z_q^s)$ as $\mathcal{S}_\theta(\mathcal{X}, \mathcal{Y})$, where $z_q^{\mathbf{n}} = \mathcal{X} \in \mathbb{R}^c$ and $z_q^s = \mathcal{Y} \in \mathbb{R}^c$ are high dimensional latent vectors. According to the explicit surface fitting, we formulate the signed hyper surface $\mathcal{S}_\theta : \mathbb{R}^{2c} \to \mathbb{R}^c$ as a feature-based polynomial function [32]

$$\mathcal{S}_{\theta,\mu}(\mathcal{X}, \mathcal{Y}) = \sum_{k=0}^{\mu} \sum_{j=0}^{k} \theta_{k-j,j}\, \mathbf{x}_{k-j} \mathbf{y}_j = \theta\,[\mathcal{X} : \mathcal{Y}], \quad (8)$$

where $[\,:\,]$ means the feature fusion through concatenation, μ denotes the number of fused items.

Similar to Eq. (2), the bivariate function $\mathcal{S}_{\theta,\mu}(\mathcal{X}, \mathcal{Y})$ aims to map a feature pair $(\mathcal{X}_i, \mathcal{Y}_i)$ to their ground-truth value $\mathcal{Z}_i = \hat{\mathcal{S}}(\mathcal{X}_i, \mathcal{Y}_i) \in \mathbb{R}^c$ in the feature space, *i.e.*,

$$\mathcal{S}_{\theta,\mu}^* = \underset{\theta,\mu}{\mathrm{argmin}} \sum_{i=1}^{N} \|\mathcal{Z}_i - \mathcal{S}_{\theta,\mu}(\mathcal{X}_i, \mathcal{Y}_i)\|^2. \quad (9)$$

To solve the oriented normal $\vec{\mathbf{n}}$ from signed hyper surfaces, we introduce a normal prediction module $\mathcal{H}(\cdot)$, thus

$$\mathcal{S}_{\theta,\mu}^* = \underset{\theta,\mu}{\mathrm{argmin}} \sum_{i=1}^{N} \|\mathcal{H}(\mathcal{Z}_i) - \mathcal{H}(\mathcal{S}_{\theta,\mu}(\mathcal{X}_i, \mathcal{Y}_i))\|^2. \quad (10)$$

Finally, the oriented normal is optimized by

$$\mathcal{S}_{\theta,\mu}^* = \underset{\theta,\mu}{\mathrm{argmin}} \sum_{i=1}^{N} \|\hat{\vec{\mathbf{n}}}_i - \vec{\mathbf{n}}_i\|^2. \quad (11)$$

Attention-weighted Normal Prediction $\mathcal{H}(\cdot) : \mathbb{R}^c \to \mathbb{R}^4$. As shown in Fig. 4, we use an attention mechanism to recover the oriented normal $\vec{\mathbf{n}}_q$ of the query point q from c-dimensional fused surface embedding z_q,

$$(\dot{\mathbf{n}}_q, s) = \mathcal{O}\big(\mathcal{V}(o_q) \otimes \mathrm{MAX}\big\{\mathrm{softmax}_{\mathcal{N}_q}\big(\mathcal{Q}_j(o_q)_{j=1}^m\big)\big\}\big), \quad (12)$$

where $o_q = \tau \cdot z_q, \tau = \mathrm{sigmoid}(\mathcal{I}(z_q))$. $\mathcal{O}, \mathcal{V}, \mathcal{Q}$ and $\mathcal{I}$ are MLPs. $m = 64$ is the feature dimension size. First, a multi-head strategy is adopted to deliver m relative weights $\mathcal{Q}_j(o_q)$, which are normalized by softmax over neighbors $\mathcal{N}_q$ into positive interpolation weights. Then, the feature maxpooling $\mathrm{MAX}\{\cdot\}$ is performed to produce attention weights for each point. Meanwhile, the feature embedding o_q is refined through another branch $\mathcal{V}$ and modulated as the weighted sum through matrix multiplication. Finally, the normal and its sign (*i.e.*, orientation) $\vec{\mathbf{n}}_q = (\mathbf{n}_q \in \mathbb{R}^3, s \in \mathbb{R})$ is predicted as a 4D vector by $\mathcal{O}$, and $\mathbf{n}_q = \dot{\mathbf{n}}_q / \|\dot{\mathbf{n}}_q\|$.

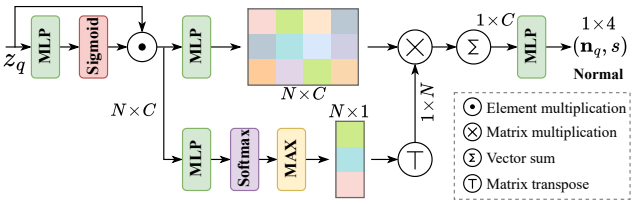

Figure 4. Attention-weighted normal prediction module $\mathcal{H}(\cdot)$.

4.3. Feature Encoding

Patch Encoding. Given a neighborhood point patch $\boldsymbol{p}_q$ of the query point, our local latent code extraction layer $\mathcal{F}$ is

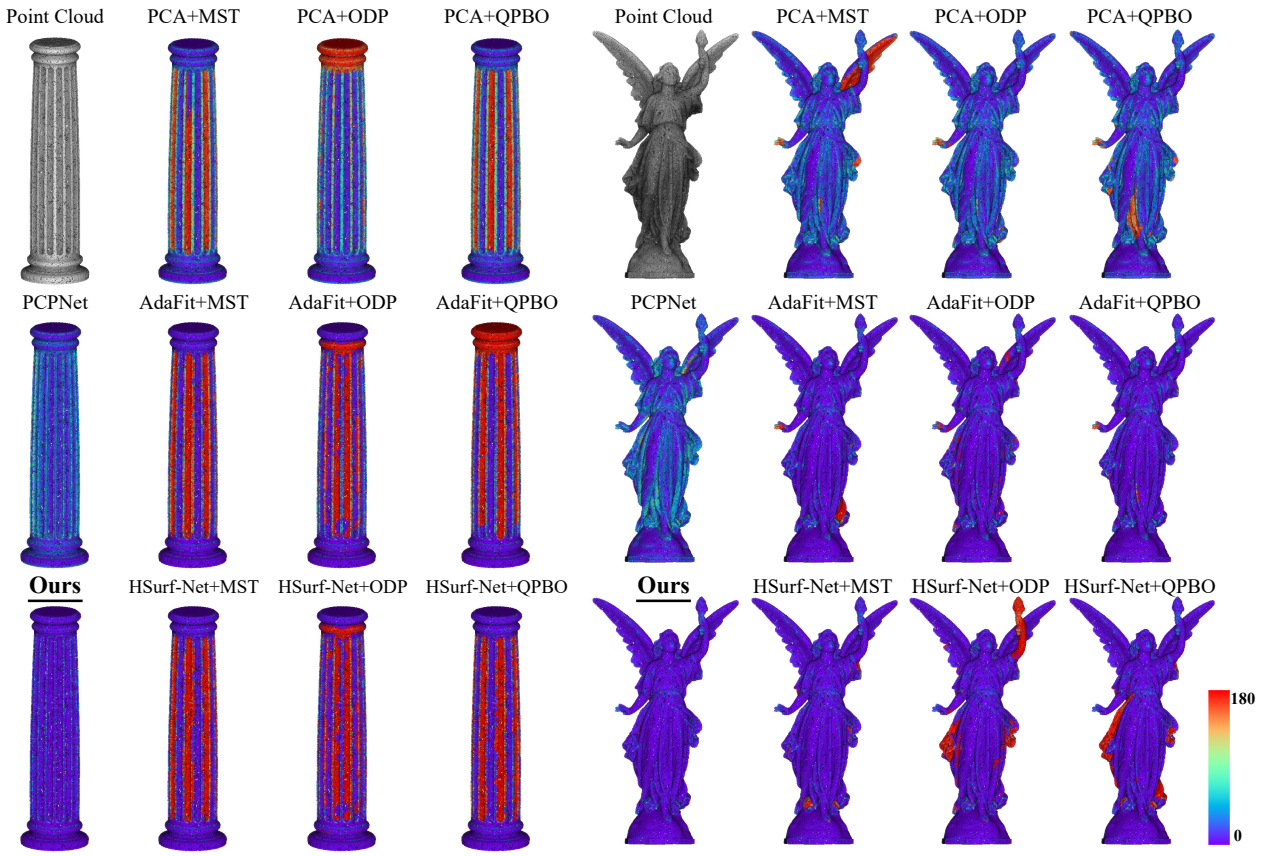

Figure 5. Visualization of the oriented normal error on datasets PCPNet (left) and FamousShape (right). The angle error is mapped to a heatmap ranging from 0° to 180°. The purple color indicates the same direction as the ground-truth, while the red color is the opposite.

formulated as

$$\dot{z}_i^{\mathbf{n}} = \mathcal{A}\left(\mathcal{B}\left(\mathrm{MAX}\{\mathcal{C}(w_j \cdot z_j^{\mathbf{n}})\}_{j=1}^{N_l}\right), z_i^{\mathbf{n}}\right), \tag{13}$$

where $i = 1, \cdots, N_{l+1}$, l is the neighborhood scale index and $N_{l+1} \leqslant N_l$. $z_i^{\mathbf{n}} = \mathcal{D}(p_i), p_i \in \boldsymbol{p}_q$ is the per-point feature in the patch. $\mathcal{A}, \mathcal{B}, \mathcal{C}$ and $\mathcal{D}$ are MLPs. $\mathrm{MAX}\{\cdot\}$ denotes the feature maxpooling over N_l-nearest neighbors of the query point q. w is a distance-based weight given by

$$w_j = \frac{\beta_j}{\sum_{i=1}^{N} \beta_i}, \ \beta_i = \mathrm{sigmoid}\big(\gamma_1 - \gamma_2 ||p_i - q||_2\big), \tag{14}$$

where γ_1 and γ_2 are learnable parameters with an initial value of 1.0. The weight w makes the layer focus on the point p_i which is closer to the query point q. As shown in Fig. 3, we stack two layers $\mathcal{F}$ to form a block, which is further stacked to build our patch feature encoder e_φ.

Shape Encoding. Since the global subsample set $\boldsymbol{P}_q = \{p_i\}_{i=1}^{N_{\boldsymbol{P}}}$ can be seen as a patch with points distributed globally on the shape surface, we adopt a similar network architecture with the patch feature encoder to get the global latent code z_q^s. To obtain $\boldsymbol{P}_q$, we use a probability-based sampling strategy [14], which brings more points closer to the query point q. It samples points according to a density gradient that decreases with increasing distance from the point q. Moreover, we find that adding more points from random sampling brings better results. Then, the gradient of a point is calculated by

$$\upsilon(p_i) = \begin{cases} \left[1 - 1.5 \dfrac{\|p_i - q\|_2}{\max_{p_j \in \mathcal{P}} \|p_j - q\|_2}\right]_{0.05}^{1} \\ 1 \quad \text{if } i \in \mathcal{R} \end{cases} \tag{15}$$

where $[\cdot]_{0.05}^{1}$ indicates value clamping. $\mathcal{R}$ is a random sample index set of $\mathcal{P}$ with $N_{\boldsymbol{P}}/1.5$ items. Finally, the sampling probability of a point $p_i \in \mathcal{P}$ is $\rho(p_i) = \upsilon(p_i) / \sum_{p_j \in \mathcal{P}} \upsilon(p_j)$.

Feature Fusion. In order to allow each point in the local patch to have global information and determine the normal orientation, we first use the maxpooling and repetition operation to make the output global latent code has the same dimension as the local latent code. Then, the two kinds of codes are fused by concatenation, *i.e.*, $[z_q^{\mathbf{n}} : z_q^s]$ in Eq. (8).

4.4. Loss Functions

For the query point q, we constrain its unoriented normal and normal sign (*i.e.*, orientation), respectively. To learn an accurate unoriented normal, we employ the ground-truth $\hat{\mathbf{n}}_q$ to calculate a normal vector sin loss [5]

$$\mathcal{L}_{sin} = \|\mathbf{n}_q \times \hat{\mathbf{n}}_q\|. \tag{16}$$

Table 1. Unoriented normal RMSE results on datasets PCPNet and FamousShape. $*$ means the source code is uncompleted or unavailable.

Category	PCPNet Dataset							FamousShape Dataset						
	Noise				Density		Average	Noise				Density		Average
	None	0.12%	0.6%	1.2%	Stripe	Gradient		None	0.12%	0.6%	1.2%	Stripe	Gradient	
Jet [10]	12.35	12.84	18.33	27.68	13.39	13.13	16.29	20.11	20.57	31.34	45.19	18.82	18.69	25.79
PCA [19]	12.29	12.87	18.38	27.52	13.66	12.81	16.25	19.90	20.60	31.33	45.00	19.84	18.54	25.87
PCPNet [17]	9.64	11.51	18.27	22.84	11.73	13.46	14.58	18.47	21.07	32.60	39.93	18.14	19.50	24.95
Zhou *et al.* * [57]	8.67	10.49	17.62	24.14	10.29	10.66	13.62	-	-	-	-	-	-	-
Nesti-Net [6]	7.06	10.24	17.77	22.31	8.64	8.95	12.49	11.60	16.80	31.61	39.22	12.33	11.77	20.55
Lenssen *et al.* [29]	6.72	9.95	17.18	21.96	7.73	7.51	11.84	11.62	16.97	30.62	39.43	11.21	10.76	20.10
DeepFit [5]	6.51	9.21	16.73	23.12	7.92	7.31	11.80	11.21	16.39	29.84	39.95	11.84	10.54	19.96
MTRNet* [9]	6.43	9.69	17.08	22.23	8.39	6.89	11.78	-	-	-	-	-	-	-
Refine-Net [56]	5.92	9.04	16.52	22.19	7.70	7.20	11.43	-	-	-	-	-	-	-
Zhang *et al.* * [54]	5.65	9.19	16.78	22.93	6.68	6.29	11.25	9.83	16.13	29.81	39.81	9.72	9.19	19.08
Zhou *et al.* * [58]	5.90	9.10	16.50	22.08	6.79	6.40	11.13	-	-	-	-	-	-	-
AdaFit [59]	5.19	9.05	16.45	21.94	6.01	5.90	10.76	9.09	15.78	29.78	38.74	8.52	8.57	18.41
GraphFit [31]	5.21	8.96	**16.12**	21.71	6.30	5.86	10.69	8.91	15.73	29.37	38.67	9.10	8.62	18.40
HSurf-Net [32]	4.17	8.78	16.25	21.61	4.98	4.86	10.11	7.59	15.64	29.43	**38.54**	**7.63**	7.40	17.70
Ours	**3.95**	**8.55**	16.13	**21.53**	**4.91**	**4.67**	**9.96**	**7.41**	**15.34**	**29.33**	38.56	7.74	**7.28**	**17.61**

For the normal orientation, we adopt the binary cross entropy H [14] to calculate a sign classification loss

$$\mathcal{L}_{sgn} = H\Big(\sigma\big(g^s(q)\big),\ [f_{\boldsymbol{S}}(q) > 0]\Big), \tag{17}$$

where σ is a logistic function that converts the sign logits to probabilities. $[f_{\boldsymbol{S}}(q) > 0]$ is 1 if the estimated normal faces the outward of surface $\boldsymbol{S}$ and 0 otherwise. Our method achieves a significant performance boost by dividing the oriented normal estimation into unoriented normal regression and its sign classification, instead of directly regressing the oriented normals of query points (see Sec. 5.3).

To make the model also pay attention to neighbor points $p_i \in \boldsymbol{p}_q$, we compute a weighted mean square error (MSE)

$$\mathcal{L}_{mse} = \frac{1}{N}\sum_{i=1}^{N} \tau_i \|\vec{\mathbf{n}}_i - \hat{\vec{\mathbf{n}}}_i\|^2, \tag{18}$$

where the neighborhood point normals $\vec{\mathbf{n}} = \delta(z_q)$ are predicted from the surface embedding z_q by an MLP layer $\delta : \mathbb{R}^c \to \mathbb{R}^3$. Moreover, we add a loss term according to coplanarity [54] to facilitate the learning of τ in Eq.(12),

$$\mathcal{L}_{\tau} = \frac{1}{N}\sum_{i=1}^{N}(\tau_i - \hat{\tau}_i)^2,\ \ \hat{\tau}_i = \exp\left(-\frac{(p_i \cdot \hat{\mathbf{n}}_q)^2}{\xi^2}\right), \tag{19}$$

where $\xi = \max(0.05^2,\ 0.3\sum_{i=1}^{N}(p_i \cdot \hat{\mathbf{n}}_q)^2/N)$. In summary, our final training loss for oriented normal estimation is

$$\mathcal{L} = \lambda_1\mathcal{L}_{sin} + \lambda_2\mathcal{L}_{sgn} + \lambda_3\mathcal{L}_{mse} + \lambda_4\mathcal{L}_{\tau}\ , \tag{20}$$

where $\lambda_1 = 0.1$, $\lambda_2 = 0.1$, $\lambda_3 = 0.5$ and $\lambda_4 = 1.0$ are factors.

5. Experiments

Implementation. We only train our network model on the PCPNet shape dataset [17], which provides the ground-truth normals with consistent orientation (outward of the surface). We follow the same train/test data split and data processing as in [5, 17, 32, 59]. For patch encoding, we randomly select a query point from the shape point cloud and search its 700 neighbors to form a patch. For shape encoding, we sample $N_{\boldsymbol{P}} = 1200$ points from the shape point cloud according to the sampling probability. The Adam optimizer is adopted with an initial learning rate of $9{\times}10^{-4}$ which is decayed to $1/5$ of the latest value at epochs $\{400, 600, 800\}$. The model is trained on an NVIDIA 2080 Ti GPU with a batch size of 145 and epochs of 800.

Dataset. Due to the lack of relevant datasets and the relatively simple test shapes of the PCPNet dataset [17], we further collect shapes with complex structures from other public datasets, such as the Famous dataset [14] and the Stanford 3D Scanning Repository [12]. We follow the same preprocessing steps as the PCPNet dataset to conduct data augmentation, *e.g.*, adding Gaussian noise with different levels (0.12%, 0.6% and 1.2%) and uneven sampling (stripe and gradient). The ground-truth oriented normals are extracted from mesh data and used for evaluation. We call this dataset *FamousShape* and it is available along with our code.

Metrics. We use the angle Root Mean Squared Error (RMSE) to evaluate the estimated normals and the Area Under the Curve (AUC) to show the error distribution [17, 32, 59]. Note that for the baseline methods, we flip their oriented normals if more than half of the normals face inward.

5.1. Unoriented Normal Comparison

We directly use our *oriented* normal estimation results to compare with baseline methods that are designed for estimating *unoriented* normals, such as the traditional methods PCA [19] and Jet [10], the learning-based surface fitting methods DeepFit [5] and AdaFit [59], and the learning-based regression methods Nesti-Net [6] and HSurf-Net [32]. As shown in Table 1, we report quantitative comparison results with the baselines in terms of normal angle RMSE on two datasets, PCPNet and FamousShape. On the PCPNet dataset, our method achieves the best perfor-

Table 2. Oriented normal RMSE results on datasets PCPNet and FamousShape. $*$ means the source code is uncompleted.

Category	PCPNet Dataset							FamousShape Dataset						
	Noise				Density		Average	Noise				Density		Average
	None	0.12%	0.6%	1.2%	Stripe	Gradient		None	0.12%	0.6%	1.2%	Stripe	Gradient	
PCA [19]+MST [19]	19.05	30.20	31.76	39.64	27.11	23.38	28.52	35.88	41.67	**38.09**	60.16	31.69	35.40	40.48
PCA [19]+QPBO [45]	18.55	21.61	30.94	39.54	23.00	25.46	26.52	32.25	39.39	41.80	61.91	36.69	35.82	41.31
PCA [19]+ODP [38]	28.96	25.86	34.91	51.52	28.70	23.00	32.16	30.47	31.29	41.65	84.00	39.41	30.72	42.92
AdaFit [59]+MST [19]	27.67	43.69	48.83	54.39	36.18	40.46	41.87	43.12	39.33	62.28	60.27	45.57	42.00	48.76
AdaFit [59]+QPBO [45]	26.41	24.17	40.31	48.76	27.74	31.56	33.16	27.55	37.60	69.56	62.77	27.86	29.19	42.42
AdaFit [59]+ODP [38]	26.37	24.86	35.44	51.88	26.45	20.57	30.93	41.75	39.19	44.31	72.91	45.09	42.37	47.60
HSurf-Net [32]+MST [19]	29.82	44.49	50.47	55.47	40.54	43.15	43.99	54.02	42.67	68.37	65.91	52.52	53.96	56.24
HSurf-Net [32]+QPBO [45]	30.34	32.34	44.08	51.71	33.46	40.49	38.74	41.62	41.06	67.41	62.04	45.59	43.83	50.26
HSurf-Net [32]+ODP [38]	26.91	24.85	35.87	51.75	26.91	20.16	31.07	43.77	43.74	46.91	72.70	45.09	43.98	49.37
PCPNet [17]	33.34	34.22	40.54	44.46	37.95	35.44	37.66	40.51	41.09	46.67	54.36	40.54	44.26	44.57
DPGO* [50]	23.79	25.19	35.66	43.89	28.99	29.33	31.14	-	-	-	-	-	-	-
Ours	**10.28**	**13.23**	**25.40**	**35.51**	**16.40**	**17.92**	**19.79**	**21.63**	**25.96**	41.14	**52.67**	**26.39**	**28.97**	**32.79**

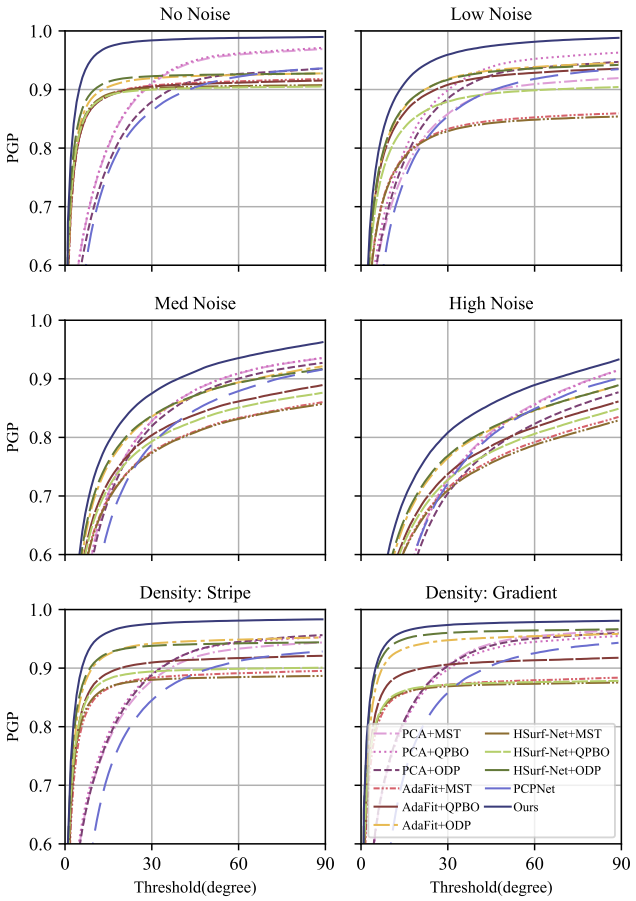

Figure 6. Oriented normal AUC on the PCPNet dataset. It shows the percentage of correctly estimated normals (PGP) for a given angle threshold. Our method has the best value for all thresholds.

mance under all noise levels and density variations. On our FamousShape dataset, our method achieves the best performance under most metrics and has the best average result.

5.2. Oriented Normal Comparison

We compare our approach for *oriented* normal estimation with various baseline methods, such as PCPNet [17] and DPGO [50]. In addition, we choose three *unoriented* normal estimation methods (PCA [19], AdaFit [59] and HSurf-Net [32]) and three normal orientation methods (MST [19], QPBO [45] and ODP [38]), and make different combinations of them to form two-stage pipelines for estimating oriented normals, such as PCA+MST. Among the baselines, PCA is a widely used traditional method, AdaFit is a representative surface fitting-based method, and HSurf-Net is a regression-based method and has the state-of-the-art performance for *unoriented* normal estimation. We use the original implementation of QPBO and ODP, and the implementation of MST in [47]. In Table 2, we show quantitative comparison results on datasets PCPNet and FamousShape. We can see that our method provides the most accurate normals under almost all noise levels and density variations for both datasets, and achieves huge performance gains in terms of average results compared to all baselines. From the experimental results, we find that the propagation-based normal orientation methods have significantly varied results when dealing with unoriented normal inputs from different estimation methods, such as PCA+MST and AdaFit+MST. A visual comparison of the normal errors is shown in Fig. 5. The error distributions of various methods are illustrated in Fig. 6.

KITTI Dataset. To verify the generalization ability of our method on LiDAR point clouds of outdoor scenes, we test directly on the KITTI dataset [15] with the model trained on the PCPNet dataset. We only report qualitative results on this dataset as it does not provide the ground-truth normals. In Fig. 7, we use the Poisson surface reconstruction algorithm [26] to generate surfaces using oriented normals estimated by different methods. As can be seen, compared to the baselines, our estimated normals facilitate the algorithm to reconstruct surfaces that can more accurately depict the spatial structure and distribution of real scenes.

5.3. Ablation Studies

We provide ablation results for oriented normal estimation in Table 3 (a)-(d), which are discussed as follows.

Table 3. Ablation studies for oriented normals on the PCPNet dataset. The last column is the average results under the unoriented metric.

	Ablation	Feat. Enco.	Module $\mathcal{H}$	Loss	Point Samp.	Noise None	Noise 0.12%	Noise 0.6%	Noise 1.2%	Density Stripe	Density Gradient	Oriented Average	Unoriented Average
(a)	w/o patch encoding		✓	✓	✓	35.19	42.23	55.59	61.38	38.92	41.49	45.80	18.83
	w/o shape encoding		✓	✓	✓	69.72	64.37	81.87	77.07	74.84	90.35	76.37	14.94
	w/o weight w		✓	✓	✓	11.15	14.32	26.49	36.03	17.99	26.03	22.00	10.48
(b)	w/o module $\mathcal{H}$	✓		✓	✓	12.08	14.53	25.87	35.88	18.45	31.84	23.11	10.24
(c)	w/o $\mathcal{L}_{sin}, \mathcal{L}_{sgn}$	✓	✓		✓	23.86	25.55	34.13	42.48	32.42	41.30	33.29	20.23
(d)	w/o density gradient	✓	✓	✓		12.10	18.25	28.05	38.15	19.79	28.09	24.07	10.00
	w/o random sample	✓	✓	✓		11.01	13.79	25.64	35.86	17.22	25.71	21.54	9.94
	$\zeta = 1/2$	✓	✓	✓		10.99	14.04	25.66	35.78	17.73	37.82	23.67	**9.92**
	$\zeta = 1/3$	✓	✓	✓		13.27	15.42	26.82	37.16	17.52	28.11	23.05	9.95
	$N_{\boldsymbol{P}} = 1100$	✓	✓	✓		10.67	14.21	25.54	35.97	16.80	26.98	21.69	9.99
	$N_{\boldsymbol{P}} = 1300$	✓	✓	✓		12.44	14.53	25.93	35.79	18.40	19.85	21.16	9.98
	Final	✓	✓	✓	✓	**10.28**	**13.23**	**25.40**	**35.51**	**16.40**	**17.92**	**19.79**	9.96

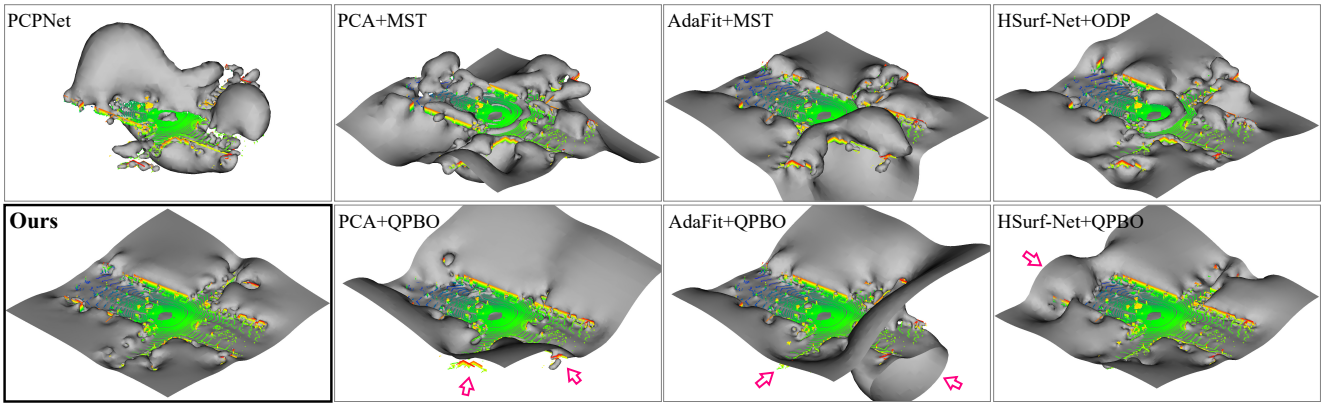

Figure 7. Street surfaces reconstructed using estimated normals on the KITTI dataset. The raw point cloud is colored with height values.

(a) Feature Encoding. (1) We realize the oriented normal estimation without using the patch encoding or the shape encoding. (2) The distance-based weight w is not used in both patch encoding and shape encoding.

(b) Module $\mathcal{H}$. The attention-weighted normal prediction module $\mathcal{H}$ is replaced with simple MLP layers.

(c) Loss $\mathcal{L}_{sin}, \mathcal{L}_{sgn}$. In our pipeline, we regress the unoriented normal and its sign of the query point q, and constrain them in loss functions $\mathcal{L}_{sin}$ and $\mathcal{L}_{sgn}$, respectively. Here, we alternatively directly predict the oriented normal $\vec{\mathbf{n}}_q$ from surface embedding and compute its MSE loss.

(d) Point Sampling. In the shape encoding, we obtain a global point set $\boldsymbol{P}_q$ by a probability-based sampling strategy as in Eq. (15), which includes density gradient term and random sample term. The point set $\boldsymbol{P}_q$ includes $N_{\boldsymbol{P}} = 1200$ points, and the ratio of randomly sampled points is $\zeta = 1/1.5$. (1) We only adopt one of the two terms in Eq. (15) for point sampling, *e.g.*, 'w/o density gradient' means all points are randomly sampled. (2) The ratio ζ is changed to $1/2$ and $1/3$. (3) The number $N_{\boldsymbol{P}}$ is set to 1100 and 1300.

From Table 3, we can conclude that both patch encoding and shape encoding are vital for learning accurate oriented and unoriented normals in our pipeline. The adoption of the weight w and the attention-weighted normal prediction module $\mathcal{H}$ effectively improves the algorithm's performance. Compared to directly predicting the oriented normal $\vec{\mathbf{n}}_q$, solving it separately by learning signed hyper surface is significantly better. The combination of the density gradient term and the random sample term in sampling can produce better results than either one alone. The proportion and number of points in the sampling of shape encoding have a significant positive effect on oriented normals, but very little on unoriented normals.

6. Conclusion

In this work, we formulate the oriented normal estimation of point clouds as the learning of signed hyper surfaces. We first review the explicit surface fitting and the implicit surface learning, and derive the formulation of the signed hyper surfaces from their inspiration. Then, we propose to use an attention-weighted normal prediction module to recover the normal and its sign of the query point from the embedding of the signed hyper surfaces. Finally, we introduce how such surfaces can be learned from the patch encoding and shape encoding using the designed loss functions. We conduct extensive evaluation and ablation experiments to report the state-of-the-art performance and justify the effectiveness of our designs. We show that the oriented normal estimation is tightly coupled with the surface reconstruction, which is an area to be explored in future work.

References

[1] Marc Alexa, Johannes Behr, Daniel Cohen-Or, Shachar Fleishman, David Levin, and Claudio T Silva. Point set surfaces. In *Proceedings Visualization, 2001. VIS'01.*, pages 21–29. IEEE, 2001. 2

[2] 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, 3

[3] Nina Amenta and Marshall Bern. Surface reconstruction by voronoi filtering. *Discrete & Computational Geometry*, 22(4):481–504, 1999. 2

[4] Samir Aroudj, Patrick Seemann, Fabian Langguth, Stefan Guthe, and Michael Goesele. Visibility-consistent thin surface reconstruction using multi-scale kernels. *ACM Transactions on Graphics*, 36(6):1–13, 2017. 2

[5] 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, 4, 5, 6

[6] 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 Conference on Computer Vision and Pattern Recognition*, pages 10112–10120, 2019. 2, 6

[7] Jules Bloomenthal, Chandrajit Bajaj, Jim Blinn, Marie-Paule Cani, Brian Wyvill, Alyn Rockwood, and Geoff Wyvill. *Introduction to implicit surfaces*. Morgan Kaufmann, 1997. 3

[8] 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

[9] Junjie Cao, Hairui Zhu, Yunpeng Bai, Jun Zhou, Jinshan Pan, and Zhixun Su. Latent tangent space representation for normal estimation. *IEEE Transactions on Industrial Electronics*, 69(1):921–929, 2021. 2, 6

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

[11] Yi-Ling Chen, Bing-Yu Chen, Shang-Hong Lai, and Tomoyuki Nishita. Binary orientation trees for volume and surface reconstruction from unoriented point clouds. In *Computer Graphics Forum*, volume 29, pages 2011–2019. Wiley Online Library, 2010. 3

[12] Brian Curless and Marc Levoy. A volumetric method for building complex models from range images. In *Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques*, pages 303–312, 1996. 6

[13] Tamal K Dey and Samrat Goswami. Provable surface reconstruction from noisy samples. *Computational Geometry*, 35(1-2):124–141, 2006. 2

[14] Philipp Erler, Paul Guerrero, Stefan Ohrhallinger, Niloy J Mitra, and Michael Wimmer. Points2Surf: learning implicit surfaces from point clouds. In *European Conference on Computer Vision*, pages 108–124. Springer, 2020. 5, 6

[15] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 3354–3361. IEEE, 2012. 7

[16] Gaël Guennebaud and Markus Gross. Algebraic point set surfaces. In *ACM SIGGRAPH 2007 papers*. 2007. 2

[17] 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. 2, 3, 4, 6, 7

[18] Taisuke Hashimoto and Masaki Saito. Normal estimation for accurate 3D mesh reconstruction with point cloud model incorporating spatial structure. In *CVPR Workshops*, pages 54–63, 2019. 2, 3

[19] 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, 3, 6, 7

[20] Hui Huang, Dan Li, Hao Zhang, Uri Ascher, and Daniel Cohen-Or. Consolidation of unorganized point clouds for surface reconstruction. *ACM Transactions on Graphics*, 28(5):1–7, 2009. 2

[21] Zhiyang Huang, Nathan Carr, and Tao Ju. Variational implicit point set surfaces. *ACM Transactions on Graphics*, 38(4):1–13, 2019. 2, 3

[22] Johannes Jakob, Christoph Buchenau, and Michael Guthe. Parallel globally consistent normal orientation of raw unorganized point clouds. In *Computer Graphics Forum*, volume 38, pages 163–173. Wiley Online Library, 2019. 2, 3

[23] Sagi Katz, Ayellet Tal, and Ronen Basri. Direct visibility of point sets. In *ACM SIGGRAPH*, pages 24–es. 2007. 3

[24] Michael Kazhdan. Reconstruction of solid models from oriented point sets. In *Proceedings of the third Eurographics Symposium on Geometry Processing*, 2005. 1

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

[26] Michael Kazhdan and Hugues Hoppe. Screened poisson surface reconstruction. *ACM Transactions on Graphics*, 32(3):1–13, 2013. 1, 7

[27] Sören König and Stefan Gumhold. Consistent propagation of normal orientations in point clouds. In *VMV*, pages 83–92, 2009. 2

[28] Carsten Lange and Konrad Polthier. Anisotropic smoothing of point sets. *Computer Aided Geometric Design*, 22(7):680–692, 2005. 2

[29] 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. 1, 2, 6

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

[31] 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. *European Conference on Computer Vision*, 2022. 1, 2, 6

[32] 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. *Advances in Neural Information Processing Systems (NeurIPS)*, 2022. 1, 2, 4, 6, 7

[33] Shujuan Li, Junsheng Zhou, Baorui Ma, Yu-Shen Liu, and Zhizhong Han. NeAF: Learning neural angle fields for point normal estimation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2023. 2

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

[35] Viní cius Mello, Luiz Velho, and Gabriel Taubin. Estimating the in/out function of a surface represented by points. In *Proceedings of the Eighth ACM Symposium on Solid Modeling and Applications*, pages 108–114, 2003. 2, 3

[36] 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

[37] Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy Networks: Learning 3D reconstruction in function space. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4460–4470, 2019. 3

[38] Gal Metzer, Rana Hanocka, Denis Zorin, Raja Giryes, Daniele Panozzo, and Daniel Cohen-Or. Orienting point clouds with dipole propagation. *ACM Transactions on Graphics*, 40(4):1–14, 2021. 1, 2, 3, 7

[39] Niloy J Mitra and An Nguyen. Estimating surface normals in noisy point cloud data. In *Proceedings of the Nineteenth Annual Symposium on Computational Geometry*, pages 322–328, 2003. 2

[40] Patrick Mullen, Fernando De Goes, Mathieu Desbrun, David Cohen-Steiner, and Pierre Alliez. Signing the unsigned: Robust surface reconstruction from raw pointsets. In *Computer Graphics Forum*, volume 29, pages 1733–1741. Wiley Online Library, 2010. 2, 3

[41] A Cengiz Öztireli, Gael Guennebaud, and Markus Gross. Feature preserving point set surfaces based on non-linear kernel regression. In *Computer Graphics Forum*, volume 28, pages 493–501. Wiley Online Library, 2009. 2

[42] Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. DeepSDF: Learning continuous signed distance functions for shape representation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 165–174, 2019. 3

[43] Mark Pauly, Markus Gross, and Leif P Kobbelt. Efficient simplification of point-sampled surfaces. In *IEEE Visualization, 2002. VIS 2002.*, pages 163–170. IEEE, 2002. 2

[44] Riccardo Roveri, A Cengiz Öztireli, Ioana Pandele, and Markus Gross. PointProNets: Consolidation of point clouds with convolutional neural networks. In *Computer Graphics Forum*, volume 37, pages 87–99. Wiley Online Library, 2018. 2

[45] Nico Schertler, Bogdan Savchynskyy, and Stefan Gumhold. Towards globally optimal normal orientations for large point clouds. In *Computer Graphics Forum*, volume 36, pages 197–208. Wiley Online Library, 2017. 1, 2, 3, 7

[46] Lee M Seversky, Matt S Berger, and Lijun Yin. Harmonic point cloud orientation. *Computers & Graphics*, 35(3):492–499, 2011. 2, 3

[47] CloudCompare (version 2.12) [GPL software]. User documentation. *http://www.cloudcompare.org*, 2022. 7

[48] Christian Walder, Olivier Chapelle, and Bernhard Schölkopf. Implicit surface modelling as an eigenvalue problem. In *Proceedings of the 22nd International Conference on Machine Learning*, pages 936–939, 2005. 2, 3

[49] Jun Wang, Zhouwang Yang, and Falai Chen. A variational model for normal computation of point clouds. *The Visual Computer*, 28(2):163–174, 2012. 2, 3

[50] Shiyao Wang, Xiuping Liu, Jian Liu, Shuhua Li, and Junjie Cao. Deep patch-based global normal orientation. *Computer-Aided Design*, page 103281, 2022. 2, 3, 7

[51] Dong Xiao, Zuoqiang Shi, Siyu Li, Bailin Deng, and Bin Wang. Point normal orientation and surface reconstruction by incorporating isovalue constraints to poisson equation. *arXiv preprint arXiv:2209.15619*, 2022. 3

[52] Hui Xie, Kevin T McDonnell, and Hong Qin. Surface reconstruction of noisy and defective data sets. In *IEEE Visualization*, pages 259–266. IEEE, 2004. 3

[53] Minfeng Xu, Shiqing Xin, and Changhe Tu. Towards globally optimal normal orientations for thin surfaces. *Computers & Graphics*, 75:36–43, 2018. 2, 3

[54] 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, 6

[55] 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. 2

[56] 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. 2, 6

[57] 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, 6

[58] Jun Zhou, Wei Jin, Mingjie Wang, Xiuping Liu, Zhiyang Li, and Zhaobin Liu. Improvement of normal estimation for point clouds via simplifying surface fitting. *arXiv preprint arXiv:2104.10369*, 2021. 2, 6

[59] 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, 4, 6, 7