ESNE: Efficient Surface Normal Estimation for LiDAR Point Clouds with Sequential Modeling and Variability Guidance

Kohei Matsuzaki and Keisuke Nonaka

KDDI Research, Inc., Japan
{ko-matsuzaki,ki-nonaka}@kddi.com

Abstract. While surface normal estimation from point clouds is a fundamental problem in computer vision, there has been limited exploration on LiDAR point clouds. Recent approaches adopt Transformer-based models designed for large-scale point clouds to achieve single-step estimation, still struggling to estimate surface normals with high computational efficiency and accuracy. In this paper, we propose an efficient surface normal estimation method for LiDAR point clouds, termed ESNE. It converts point clouds into a sequence data format and leverages a state space model to capture long-range dependencies. To capture local structure, the proposed method introduces a local attention module incorporating an efficient and explicit search for neighboring points on the sequence. Furthermore, the proposed method introduces a feature refinement module using structural variability as guidance to enhance the shape representations of local regions. Experimental results on large-scale LiDAR point cloud datasets demonstrate that the proposed method improves the mean accuracy by over 33.4% and achieves more than twice the inference speed compared to state-of-the-art methods.

Keywords: Surface Normal Estimation · LiDAR Point Clouds · Variability Guidance

1 Introduction

Surface normal estimation of point clouds is a fundamental problem in computer vision and graphics, aiming to determine normal vectors that represent the local orientation of object surfaces. Accurate surface normals play a crucial role in a wide range of applications, such as surface reconstruction [24, 26, 27, 48], registration [9, 53, 58], and autonomous driving [7, 50, 79]. However, the inherent nature of point clouds, including sparsity, irregularity, and geometric complexity, along with locally uneven distributions and missing points, remains a significant challenge that hinders accurate surface normal estimation.

Traditional surface normal estimation methods [1, 6, 22, 23, 28] estimate normals through geometric surface fitting techniques, such as principal component analysis and local polynomial fitting on neighboring points. However, these methods are highly sensitive to noise and the selection of neighboring points. With

advances in deep learning technology, learning-based methods have been introduced and have demonstrated promising performance in surface normal estimation [3, 11, 17, 32–35, 71, 81]. Most of these methods estimate the surface normal at the center of a local patch composed of neighboring points, leveraging detailed local geometric structures to achieve high accuracy. Nevertheless, they are primarily designed and evaluated on object-level or small-scale scene datasets, where point clouds are relatively densely and uniformly sampled. In contrast, LiDAR point clouds with a sparse, non-uniform distribution make local patches unreliable, degrading the performance of these methods.

Recently, a publicly available dataset and method designed to advance research on surface normal estimation for LiDAR point clouds have been proposed [47]. It is the first large-scale point cloud dataset with ground truth surface normal labels. The method is able to estimate surface normals of LiDAR point clouds in a single step without constructing local patches as a pre-processing step. However, the model of this method depends on many stacked convolutional and attention layers to achieve a wide receptive field, which can lead to high parameter counts and high inference costs. Additionally, it is limited in its ability to capture complex local structures since the model applies an attention mechanism within local point groups without performing an explicit k-nearest neighbor search. As a result, local structural variations that are important for accurate surface normal estimation may be missed.

To address these challenges, we propose an efficient and accurate surface normal estimation method for LiDAR point clouds, termed ESNE. The proposed method is a novel framework designed to effectively capture both global and local structures from point clouds with various point densities and long-range spatial distributions. The proposed method leverages the long-range modeling capabilities of a state space model [15] to capture global structural context from point clouds in a sequence data format. To model local surface details, the proposed method introduces an efficient local attention based on an explicit k-nearest neighbor search that exploits the spatial proximity of space-filling curves [55]. Furthermore, the proposed method introduces guidance based on structural variability to enhance local shape representations. This guidance is derived from the local attention weights obtained in the previous step, enabling efficient computation. Evaluation experiments on large-scale LiDAR point cloud datasets show that the proposed method achieves more accurate surface normal estimation while providing more than twice the inference speed of state-of-the-art methods.

The main contributions of this research can be summarized as follows:

- We propose an efficient and accurate surface normal estimation method for LiDAR point clouds, which effectively captures both global structural context and local surface details.
- We design a local attention module that incorporates an efficient k-nearest neighbor search operating on point clouds in sequence data format, enabling effective modeling of local structures.
- We introduce a feature refinement module that leverages structural variability based on attention weights as guidance to enhance shape representations of local regions.

– The experimental results demonstrate that the proposed method achieves superior surface normal estimation accuracy while providing more than twice the inference speed compared to state-of-the-art methods.

2 Related Work

2.1 Surface Normal Estimation

Early studies estimated surface normals by fitting a geometric surface to the local neighboring points [1, 22, 23, 28]. Among these approaches, Principal Component Analysis (PCA) [22] is the most widely used method, which estimates the surface normal as the direction of minimum variance within a local patch. Several variants based on more complex surface models have also been proposed, including moving least squares [30], jet fitting [6], and spherical fitting [16]. These methods rely on geometric priors for point cloud data and require careful parameter tuning depending on the data type. To improve the performance of surface fitting-based normal estimation, learning-based methods have been proposed [3,5,11,29,32,76,80,81]. DeepFit [3] predicts point-wise weights and applies weighted polynomial surface fitting to estimate local surfaces and their normals. Subsequent works, such as AdaFit [81], GraphFit [32], and Du *et al.* [11], also use learning-based approaches to improve the performance of local surface fitting.

Apart from the surface fitting, several studies have explored learning-based methods that directly regress surface normals from point clouds [4, 17, 33–37, 46, 47, 64, 71, 72]. PCPNet [17] predicts surface normals from features extracted from local patches with PointNet [56]. CMG-Net [71] addresses the ambiguity in scale selection by introducing a network architecture that aggregates multi-scale local features and fuses hierarchical geometric information. SHS-Net [34] extracts features independently from local patches and a globally subsampled point set, and predicts surface normals with an attention-weighted prediction module based on the aggregation of these features. MSECNet [72] introduces edge detection based on multi-scale feature fusion and edge conditioning to improve estimation accuracy in normal varying regions. However, many of these methods perform inference on a local patch basis, which makes them unsuitable for estimating surface normals from large and irregularly distributed LiDAR point clouds. In contrast, LiSu [47] enables surface normal estimation for LiDAR point clouds without constructing local patches as a pre-processing step by utilizing the Point Transformer V3 [69]. Nevertheless, it performs attention mechanisms at the group level without explicitly searching for neighboring points, which limits its ability to model local structures that provide crucial cues for surface normal estimation.

2.2 Deep Learning on 3D Point Clouds

While PointNet [56] is a pioneering deep learning framework that directly takes point clouds as input, it struggles to capture local structures. To enhance local feature extraction, subsequent methods have proposed effective network architectures based on point convolution [38, 41, 62, 68] and graph convolution [42,

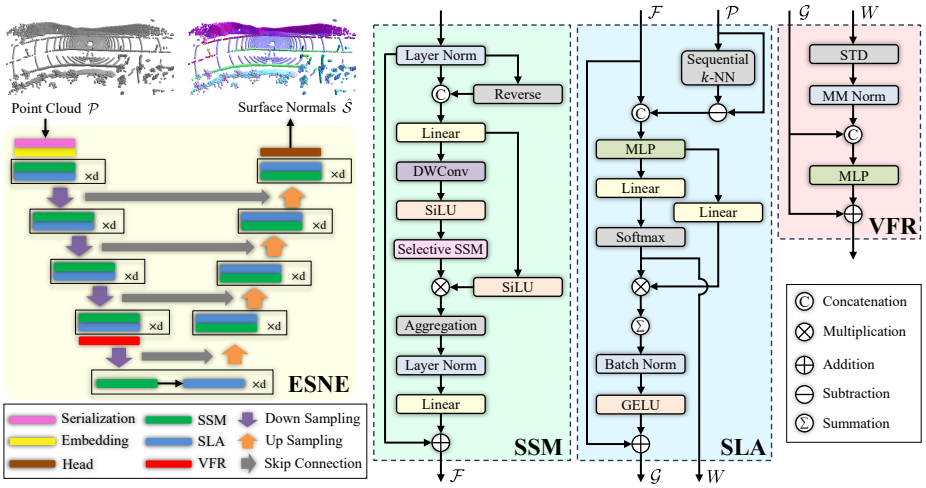

Fig. 1: Overview of the proposed method. The input is a point cloud $\mathcal{P} = \{\mathbf{p}_i \in \mathbb{R}^3\}_{i=1}^N$, and the output is the estimated surface normals $\hat{\mathcal{S}} = \{\hat{\mathbf{s}}_i \in \mathbb{R}^3\}_{i=1}^N$. The network architecture follows the U-Net framework, with each block comprising a State Space Model (SSM) module and a Sequential Local Attention (SLA) module. A Variability-guided Feature Refinement (VFR) module is introduced at a stage preceding the bottleneck.

59, 66, 73]. Subsequently, Transformer-based architectures that apply attention mechanisms for feature extraction have been explored [18,61,65,69,70,74,78]. To mitigate the quadratic computational complexity of global self-attention, most approaches adopt local attention mechanisms restricted to spatially neighboring points. Nevertheless, the computational complexity of local attention remains a challenge. Recent studies have adopted a state space model for point clouds to address the limitations of attention mechanisms, leveraging their ability to capture long-range dependencies with linear complexity [19, 31, 39, 40, 43, 63, 67, 75, 77]. However, a state space model has limited ability to capture local structures in point clouds. Although several methods address this issue by constructing local patches using the farthest point sampling [12, 49] and k-nearest neighbor search, the high computational complexity of these operations limits their scalability to large-scale point clouds. In contrast, the proposed method effectively captures both long-range dependencies and local structure through a hybrid architecture comprising a state space model and efficient local attention mechanisms.

3 Proposed Method

Fig. 1 shows an overview of the proposed method. The input to the proposed method is a point cloud $\mathcal{P} = \{\mathbf{p}_i \in \mathbb{R}^3\}_{i=1}^N$ containing only point coordinates, and the output is the estimated surface normals $\hat{\mathcal{S}} = \{\hat{\mathbf{s}}_i \in \mathbb{R}^3\}_{i=1}^N$. The network architecture of the proposed method follows the U-Net framework [57], which consists of four-stage encoders and decoders. The proposed method first generates a mapping that converts the input point cloud into a sequence data format

through serialization [69]. In this process, point coordinates are projected into a discrete space using a grid size, generating integer codes that represent their ordering along a particular space-filling curve [55]. Sorting these codes establishes a mapping to the order in the space-filling curve pattern. The embedding module projects the coordinates of the point cloud ordered according to this mapping into a high-dimensional feature representation. The processing blocks in the encoder and decoder at each stage consist of a State Space Model (SSM) module and a Sequential Local Attention (SLA) module. Each block is repeated according to a specific depth d. The proposed method introduces a Variability-guided Feature Refinement (VFR) module at the stage preceding the bottleneck to incorporate structural variability information. As downsampling and upsampling modules, the proposed method adopts grid pooling and grid unpooling [70] with skip connections. The head module predicts the surface normals from point-wise features obtained in the final decoder block. The Ground Truth (GT) of surface normals $\mathcal{S} = \{\mathbf{s}_i \in \mathbb{R}^3\}_{i=1}^{N}$ is used to compute the loss functions.

In the rest of this section, we describe the SSM, SLA, and VFR modules in Sec. 3.1, Sec. 3.2, and Sec. 3.3, respectively. We also describe the loss functions in Sec. 3.4 and the network architecture in Sec. 3.5.

3.1 State Space Model

SSM [15] represents continuous dynamical systems that capture the relationship between input sequences and outputs through latent hidden states. Given an input $x(t) \in \mathbb{R}$, the model maintains a hidden state $\mathbf{h}(t) \in \mathbb{R}^M$ and generates an output $y(t) \in \mathbb{R}$ with the linear ordinary differential equation:

$$\mathbf{h}'(t) = \mathbf{A}\mathbf{h}(t) + \mathbf{B}x(t), \quad y(t) = \mathbf{C}\mathbf{h}(t) + \mathbf{D}x(t), \tag{1}$$

where $\mathbf{A} \in \mathbb{R}^{M\times M}$, $\mathbf{B} \in \mathbb{R}^{M\times 1}$, and $\mathbf{C} \in \mathbb{R}^{1\times M}$ are learnable parameters, and $\mathbf{D} \in \mathbb{R}^1$ represents the residual connection, and M indicate the dimension of the hidden state. The continuous-time formulation is discretized using the zero-order hold method with a fixed time step Δ:

$$\bar{\mathbf{A}} = \exp(\Delta\mathbf{A}), \quad \bar{\mathbf{B}} = (\Delta\mathbf{A})^{-1}(\exp(\Delta\mathbf{A} - \mathbf{I})) \cdot \Delta\mathbf{B}. \tag{2}$$

From Eqs. (1) and (2), the discrete formulation is represented as follows:

$$\mathbf{h}_t = \bar{\mathbf{A}}\mathbf{h}_{t-1} + \bar{\mathbf{B}}x_t, \quad y_t = \mathbf{C}\mathbf{h}_t + \mathbf{D}x_t. \tag{3}$$

This discretization enables SSM to model long-range dependencies in linear time.

We describe the structure of the SSM module shown in Fig. 1. This module takes sequence data as input and first applies layer normalization [2]. To facilitate learning diverse contexts, a reverse sequence is generated from the forward sequence and subsequently concatenated with it. These sequences are processed sequentially through a linear layer, a Depth-Wise Convolution (DWConv) [8], a Sigmoid Linear Unit (SiLU) activation [13], and a selective SSM [15]. A multiplication gate mechanism is introduced through element-wise multiplication with the result of applying SiLU activation to the output of the linear layer. The hidden states from the forward and reverse sequences are aggregated by summing them after aligning their order. The aggregated hidden states are subjected to

layer normalization and a linear projection layer for output. Finally, residual connections to the output of the first layer normalization are applied.

3.2 Sequential Local Attention

While SSM excels at capturing long-range dependencies in one-dimensional sequence data, it is limited in modeling three-dimensional spatial structures. To address this limitation, the proposed method incorporates a local attention module that leverages k-nearest neighbor points to capture local structures. However, k-nearest neighbor search algorithms based on spatial distance have high computational complexity, making them problematic when scaling to point clouds with large numbers of points. To improve efficiency, the proposed method introduces a k-nearest neighbor search based on index differences of points in a sequence data format ordered through serialization. For clarity, we refer to this method as sequential k-NN, while the spatial distance-based method is spatial k-NN.

Let $\mathcal{I} = \{i \in \mathbb{Z} \mid 1 \leq i \leq N_l\}$ denote the index set of the point cloud in sequence data format, and $\mathcal{O} = \{i \in \mathbb{Z} \mid -o \leq i \leq o\}$ denote the set of index offsets. Here, N_l represents the number of points in the l-th stage. The index set of neighboring points of the i-th point is represented as

$$\mathcal{N}(i) = \{j \in \mathcal{I} \mid (j - i) \in \mathcal{O}\}, \tag{4}$$

where $|\mathcal{N}(i)| = 2o+1$, and we define $k = 2o + 1$. At the edges of the sequence data, replication padding is applied. Fig. 2 illustrates sequential k-NN for $k = 5$, which uses a Hilbert curve [21] as the serialization pattern. The query point itself is also included in the neighboring points. The sequential k-NN has the advantage of significantly higher processing efficiency than the spatial k-NN, even though it exhibits inferior spatial proximity.

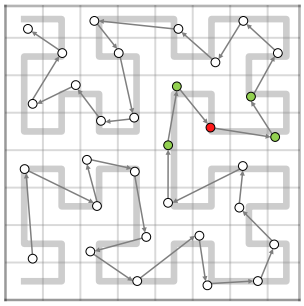

Fig. 2: Illustration of the sequential k-NN for $k = 5$. Red, green, and white points represent the query point, neighboring points, and others, respectively.

In the SLA module shown in Fig. 1, the input feature vectors and coordinates of the point cloud are denoted as $\mathcal{F} = \{\mathbf{f}_i \in \mathbb{R}^{D_l}\}_{i=1}^{N_l}$ and $\mathcal{P} = \{\mathbf{p}_i \in \mathbb{R}^3\}_{i=1}^{N_l}$, respectively. Here, D_l represents the channel dimension in the l-th stage. This module first searches the neighboring point index set $\mathcal{N}(i)$ for each point with sequential k-NN. Then, the expanded feature vector $\mathbf{z}_{ij}$ is generated by feeding the concatenation of the feature vector and relative coordinates with a point-wise Multi-Layer Perceptron (MLP) as follows:

$$\mathbf{z}_{ij} = \mathrm{MLP}(\mathbf{f}_i \parallel \mathbf{p}_i - \mathbf{p}_j), \quad \forall j \in \mathcal{N}(i), \tag{5}$$

where MLP consists of two linear layers with Rectified Linear Unit (ReLU) [14, 52] activations, and $\parallel$ denotes the concatenation operation. The feature vector $\mathbf{z}_{ij}$ is passed through a linear layer parameterized by θ and normalized using the softmax function. As a result, the weights w_{ij} that indicate the importance of the j-th neighboring points to the i-th point are estimated as

$$w_{ij} = \mathrm{Softmax}(\mathrm{Linear}_\theta(\mathbf{z}_{ij})). \tag{6}$$

The feature vectors $\mathbf{z}_{ij}$ are also passed through a linear layer parameterized by ϕ and aggregated as a weighted sum based on the weights w_{ij}, represented as

$$\mathbf{z}_i = \sum_{j \in \mathcal{N}(i)} w_{ij} \cdot \text{Linear}_\phi(\mathbf{z}_{ij}). \tag{7}$$

Then, the aggregated features are subjected to batch normalization [25] and Gaussian Error Linear Unit (GELU) [20] activation. Finally, residual connections with the input feature vectors are applied. Let $\mathcal{G} = \{\mathbf{g}_i \in \mathbb{R}^{D_l}\}_{i=1}^{N_l}$ and $W = (w_{ij}) \in \mathbb{R}^{N_l \times k}$ denote the set of feature vectors and weight matrix output by the module, respectively.

3.3 Variability-guided Feature Refinement

Although local attention mechanisms are effective at capturing local structures, they aggregate features through summation operations, leading to the loss of higher-order statistics in the weight distribution that capture structural heterogeneity. To model the structural characteristics of local regions, the proposed method introduces a variability-guided feature refinement. It learns to adaptively adjust feature vectors based on the degree of structural variability, serving as a refinement mechanism to enhance local shape representations.

In the VFR module shown in Fig. 1, the input feature vectors and weight matrix are denoted as $\mathcal{G} = \{\mathbf{g}_i \in \mathbb{R}^{D_l}\}_{i=1}^{N_l}$ and $W = (w_{ij}) \in \mathbb{R}^{N_l \times k}$, respectively. This module derives the standard deviation $\sigma_i \in \mathbb{R}$ for each point over the spatial dimension from the weight matrix W, which is given by

$$\mu_i = \frac{1}{k} \sum_{j=1}^{k} w_{ij}, \quad \sigma_i = \sqrt{\frac{1}{k} \sum_{j=1}^{k} (w_{ij} - \mu_i)^2}. \tag{8}$$

Since attention weights over homogeneous structures tend to be approximated by a uniform distribution, the standard deviation reflects deviations from structural homogeneity. Therefore, the σ_i can be interpreted as a scalar feature encoding the degree of structural variability of the local region. That is, a small σ_i indicates a homogeneous local structure, and a large σ_i indicates an inhomogeneous one. For numerical stability, min-max normalization is applied to the σ_i on a batch basis, and the normalized standard deviation is denoted as $\hat{\sigma}_i$. Fig. 3 illustrates the degree of variability, showing that it is higher in regions where the local structure is inhomogeneous. The module feeds the concatenation of $\mathbf{g}_i$ and $\hat{\sigma}_i$ into an MLP, applying residual connections to refine features guided by structural variability. Therefore, the refined feature vector $\mathbf{r}_i \in \mathbb{R}^{D_l}$ is given as

$$\mathbf{r}_i = \mathbf{g}_i + \text{MLP}(\mathbf{g}_i \parallel \hat{\sigma}_i), \tag{9}$$

where MLP consists of two linear layers with ReLU activations, and $\parallel$ denotes the concatenation operation. The output of this module is refined feature

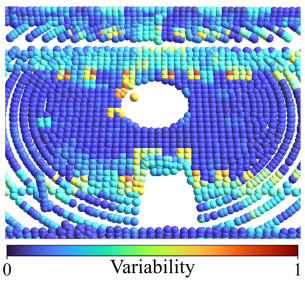

Fig. 3: Visualization of the normalized variability.

vectors. Large standard deviations derived from the weight matrix at shallow stages may arise from attention to noise and outliers rather than from inhomogeneous local structure. To address this concern, the proposed method introduces the VFR module at the stage preceding the bottleneck, where these effects are suppressed through downsampling.

3.4 Loss Functions

Similar to previous work [47], the model is trained using a combination of Weighted Mean Absolute Error (WMAE) loss, Spatial Graph Total Variation (SGTV) regularization, Temporal Graph Total Variation (TGTV) regularization, and Eikonal regularization.

The WMAE loss between the estimated surface normal and the corresponding GT surface normal is formulated as

$$\mathcal{L}_{\mathrm{WMAE}} = \frac{1}{N}\sum_{i=1}^{N} w_i \|\mathbf{s}_i - \hat{\mathbf{s}}_i\|_1, \tag{10}$$

where w_i is a weight that is inversely proportional to the frequency of $\mathbf{s}_i$.

The SGTV regularization term is introduced to spatially smooth the estimated surface normals. It is defined using the set of edges $\mathcal{E} = \{(i,j) \mid j \in \mathcal{N}(i)\}$ of the spatial k-NN graph constructed from the point cloud $\mathcal{P}$:

$$\mathcal{L}_{\mathrm{SGTV}} = \frac{1}{|\mathcal{E}|}\sum_{(i,j)\in\mathcal{E}} w_{ij} \|\hat{\mathbf{s}}_i - \hat{\mathbf{s}}_j\|_1, \tag{11}$$

where w_{ij} is a weight decaying with the distance between points $\mathbf{p}_i$ and $\mathbf{p}_j$.

The TGTV regularization term is introduced to enhance the temporal consistency between two consecutive frames. Let the point clouds at time t and $t+1$, transformed into a common coordinate system, be denoted as $\mathcal{P}^t = \{\mathbf{p}_i^t \in \mathbb{R}^3\}_{i=1}^{N^t}$ and $\mathcal{P}^{t+1} = \{\mathbf{p}_i^{t+1} \in \mathbb{R}^3\}_{i=1}^{N^{t+1}}$, respectively. Let the estimated surface normals corresponding to them be $\hat{\mathcal{S}}^t = \{\hat{\mathbf{s}}_i^t \in \mathbb{R}^3\}_{i=1}^{N^t}$ and $\hat{\mathcal{S}}^{t+1} = \{\hat{\mathbf{s}}_i^{t+1} \in \mathbb{R}^3\}_{i=1}^{N^{t+1}}$, respectively. The edge set in the spatial k-NN graph from $\mathcal{P}^t$ to $\mathcal{P}^{t+1}$ is constructed as $\mathcal{E}^{t\to t+1} = \{(i,j) \mid j \in \mathcal{N}^{t\to t+1}(i)\}$. The TGTV regularization term is defined as follows:

$$\mathcal{L}_{\mathrm{TGTV}} = \frac{1}{|\mathcal{E}^{t\to t+1}|}\sum_{(i,j)\in\mathcal{E}^{t\to t+1}} w_{ij} \|\hat{\mathbf{s}}_i^t - \hat{\mathbf{s}}_j^{t+1}\|_1, \tag{12}$$

where w_{ij} is a weight decaying with the distance between points $\mathbf{p}_i^t$ and $\mathbf{p}_j^{t+1}$.

The Eikonal regularization term is introduced to impose a unit norm constraint on the estimated surface normals and is defined as

$$\mathcal{L}_{\mathrm{Eikonal}} = \frac{1}{N}\sum_{i=1}^{N} \left(\|\hat{\mathbf{s}}_i\|_2 - 1\right)^2. \tag{13}$$

The total loss $\mathcal{L}$ is formulated as a weighted sum of the loss and regularization terms:

$$\mathcal{L} = \mathcal{L}_{\mathrm{WMAE}} + \alpha\left(\mathcal{L}_{\mathrm{SGTV}} + \mathcal{L}_{\mathrm{TGTV}} + \mathcal{L}_{\mathrm{Eikonal}}\right), \tag{14}$$

where α is a weight for adjusting the balance of terms.

3.5 Network Architecture

We describe the details of the network architecture shown in Fig. 1. We set the encoder channel dimensions to [64, 64, 128, 256, 512] and the decoder channel dimensions to [64, 64, 128, 256]. The encoder depth is set to [1, 1, 2, 4, 1], and the decoder depth is set to [1, 1, 1, 1]. The grid size for serialization is set to 0.1. The grid size multiplier is set to 2 at all stages. The serialization patterns include a Hilbert curve [21] that sequentially scans along the x, y, and z axes, as well as variants that operate along the y, z, and x axes, and z, x, and y axes. Unlike the previous study [47], Z-order curves [51] are not included in the serialization patterns since they have low locality-preserving properties. The hidden state dimension of the SSM module is set to $M = 32$. For sequential k-NN, k is set to 9. The embedding module is implemented as an SLA module excluding residual connections. The output dimensions of the hidden layers of the MLP in Eqs. (5) and (9) are set to be the same as the dimensions of the final output layer. The prediction head is implemented as a point-wise MLP consisting of three linear layers with ReLU [14, 52] activations, except for the final layer. The channel dimensions of the prediction head are set to [64, 32, 16, 3].

4 Experiments

4.1 Experimental Setups

Datasets. As public LiDAR datasets with surface normal labels are limited, the main dataset used to evaluate the proposed method is the LiSu dataset [47]. This dataset consists of large-scale LiDAR point clouds generated using the CARLA [10] simulator from a variety of urban and rural environments, including downtown areas, small towns, and highways. The LiDAR sensor is configured to emit 64 laser beams at a frame rate of 10 Hz and capture approximately 100k points per frame. Gaussian noise with a standard deviation of 0.02 meters is introduced into the point clouds. Following [47], we downsampled the test set to every fifth frame to allow for a fair comparison within a reasonable timeframe.
Baselines. We adopt traditional surface normal estimation methods, PCA [22] and Jet [6], as baseline methods. The k-NN is set to $k = 32$ in these methods. We also adopt state-of-the-art supervised learning-based methods, including PCP-Net [17], GraphFit [32], Du *et al.* [11], NGLO [33], CMG-Net [71], SHS-Net [34], MSECNet [72], and LiSu [47] as baseline methods. We trained models of these supervised learning-based methods on the LiSu dataset according to the settings described in [47]. For all methods except LiSu, the patch size is set to 32.
Evaluation Metrics. As evaluation metrics, we adopt the Mean, Median, and Root Mean Square Error (RMSE) of the angle error in degrees. The Median reflects the accuracy achieved for most points, and the Mean and RMSE capture the overall error magnitude and the influence of large angular deviations. We also adopt 5.0°, 7.5°, 11.5°, 22.5°, and 30.0° as evaluation metrics, indicating the proportion of predictions with angular errors below each threshold. Furthermore, we present the average inference time (Time) to evaluate the efficiency.

Table 1: Quantitative performance comparison with state-of-the-art methods for surface normal estimation. Time is reported in seconds. The symbols ↓/↑ indicate that lower/higher values are better. Bold indicates the best result in each column.

Method	Mean↓	Med.↓	RMSE↓	5.0°↑	7.5°↑	11.25°↑	22.5°↑	30.0°↑	Time↓
PCA [22]	72.75	29.64	107.96	40.47	44.27	47.85	52.08	53.85	0.10
Jet [6]	38.41	21.69	53.85	10.33	18.17	30.19	52.04	59.52	12.34
PCPNet [17]	10.83	1.74	22.03	66.36	70.17	73.82	80.74	84.27	11.82
GraphFit [32]	9.45	1.18	21.96	70.77	72.83	74.71	77.08	79.35	15.76
Du *et al.* [11]	9.40	1.15	21.64	71.35	73.21	75.07	77.51	80.01	16.15
NGLO [33]	10.54	1.48	22.16	73.45	76.81	78.71	80.26	83.33	24.86
CMG-Net [71]	12.42	2.61	24.63	64.81	69.71	73.09	74.93	77.16	34.49
SHS-Net [34]	7.88	0.83	19.32	77.37	79.38	82.01	86.46	89.10	22.79
MSECNet [72]	6.65	1.55	15.01	72.19	79.64	85.55	92.18	94.12	7.53
LiSu [47]	6.26	0.40	17.38	81.01	84.31	87.12	91.52	93.03	0.08
Ours	**4.17**	**0.20**	**13.94**	**87.40**	**89.73**	**91.87**	**94.73**	**95.68**	**0.04**

Implementation Details. We implement the proposed method using the PyTorch framework [54]. We train the model for 50 epochs with a batch size of 4 in mixed precision. We use the AdamW optimizer [45] with a maximum learning rate of 0.0001 and weight decay of 0.005. The cosine annealing strategy [44] is used for scheduling the learning rate. During training, we randomly downsample the point cloud to 70k points. We use data augmentation, including rotation, scaling, jitter, flipping, and point shuffling. We apply random dropout with probability 0.3. In the k-NN graph for both SGTV and TGTV, k is set to 8 as in previous work [47]. In Eq. (14), α is set to 0.1. All experiments are conducted on a workstation equipped with two NVIDIA RTX 6000 Ada GPUs, an AMD Ryzen Threadripper PRO 7985WX CPU (3.20 GHz), and 128 GB of RAM.

4.2 Comparison with State-of-the-art Methods

Quantitative Evaluation. Table 1 shows a comparison of surface normal estimation performance. It can be seen that the proposed method achieves superior performance over the baseline methods across all metrics, improving mean accuracy by more than 33.4%, while being more than twice as fast in inference. Table 2 compares the number of model parameters (Params) with the floating-point operations (FLOPs) and Peak Memory Usage (PMU) during inference, for the proposed method and the second fastest method, LiSu [47]. The proposed method reduces the Params by 89.4% and also shows superior efficiency in terms of FLOPs and PMU, compared to LiSu. The high computational efficiency of the proposed method opens up the possibility of real-time processing for surface normal estimation.

Table 2: Comparison of efficiency.

	LiSu [47]	Ours
Params↓	46.2 M	4.9 M
FLOPs↓	1670.1 G	413.1 G
PMU↓	1.01 GB	0.59 GB

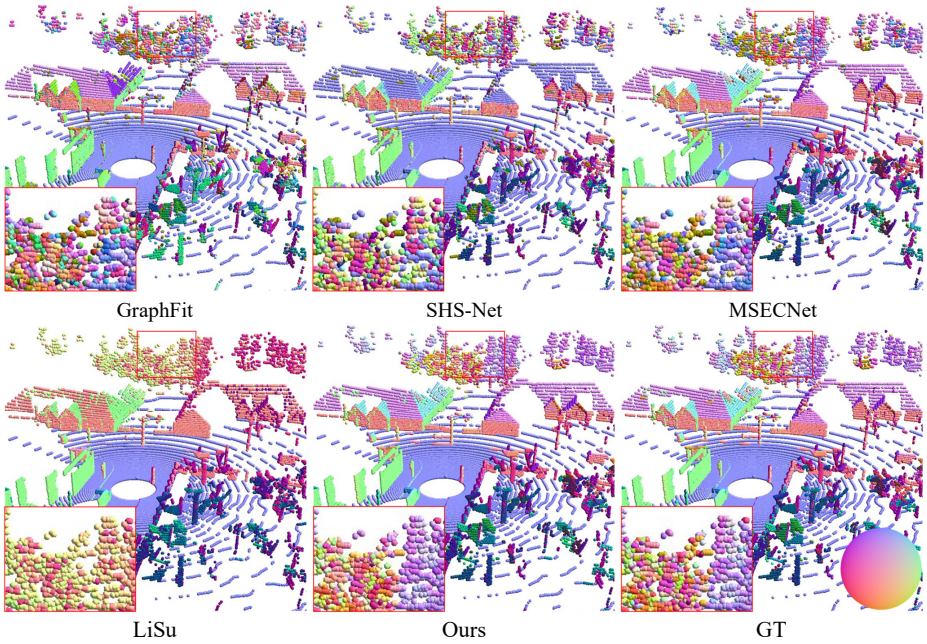

Fig. 4: Qualitative performance comparison with state-of-the-art methods on the LiSu dataset. The colors of the points represent the normal vectors mapped to RGB values. A legend sphere is shown in the bottom right. As highlighted by the red rectangle, the proposed method estimates surface normals close to the GT, even for irregular points.

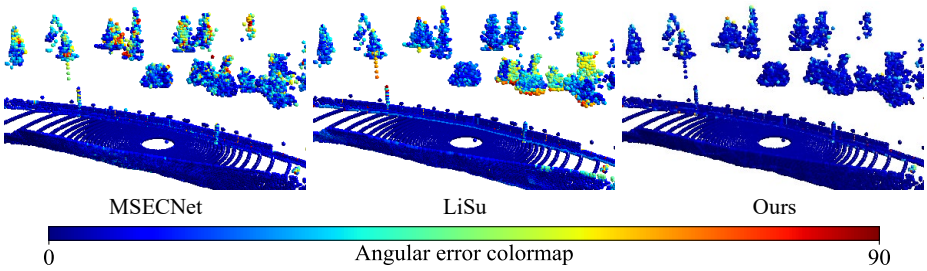

Fig. 5: Visualization of the angular errors of estimated normals. The colors of the points represent the angular errors, which are mapped to RGB values using the colormap.

Qualitative Evaluation. Fig. 4 shows the surface normal estimation results and the GT surface normals. The colors of the points represent the normal vectors mapped to RGB values. As highlighted by the red rectangle, baseline methods sometimes produce inaccurate estimation results of surface normals for irregular and sparse LiDAR point clouds. In contrast, it can be confirmed that the proposed method consistently achieves accurate surface normal estimation.

Visual Error Analysis. Fig. 5 visualizes the magnitude of the angular error between the estimated surface normals and the GT normals. The colors of the points represent the angular errors, which are mapped to RGB values using

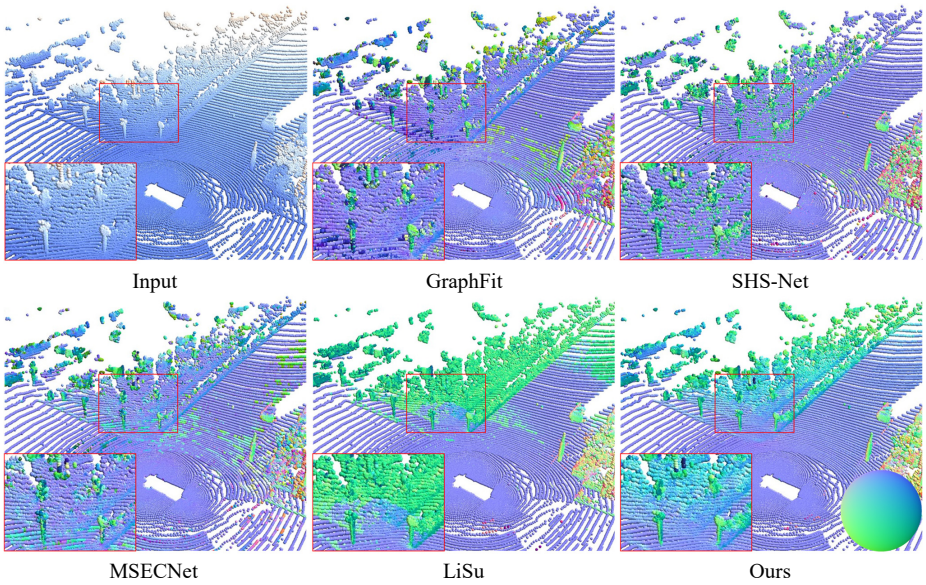

Fig. 6: Qualitative evaluation on real-world LiDAR point clouds. The colors of the points in the Input represent vertical heights, while in the other cases, they represent normal vectors mapped to RGB values. A legend sphere is shown in the bottom right. The red rectangle highlights that the proposed method estimates smoother normals.

the colormap. While baseline methods can estimate surface normals with high accuracy in planar areas such as the ground, they suffer from large errors for complex shapes such as street trees. The proposed method can estimate surface normals with high accuracy across the entire scene, including complex shapes.

Generalization to Real-world Data. We verify the generalization ability of the proposed method on LiDAR point clouds acquired in the real world. Fig. 6 shows the surface normal estimation results on the Waymo Open Dataset [60] using models trained on the LiSu dataset. This figure does not include GT since this dataset does not provide surface normal labels. Baseline methods are often observed to estimate surface normals that are inconsistent with spatial structure, as highlighted by the red rectangle. It can be seen that the proposed method estimates surface normals with greater consistency and smoothness than the baseline methods, demonstrating its generalization ability.

4.3 Ablation Study

Model Architecture. We evaluate the individual effectiveness of the main components of the proposed method: SSM, SLA, and VFR. Table 3 shows the results of the ablation study for these components. When ablating SLA, VFR is also ablated since our VFR requires the weight matrix obtained from SLA. Ablating SSM prevents the capture of global context, making accurate estimation of spatially consistent surface normals challenging. The abolition of SLA

Table 3: Ablation study. Time is reported in milliseconds. SSM: State Space Model. SLA: Sequential Local Attention. VFR: Variability-guided Feature Refinement.

SSM	SLA	VFR	Mean↓	Med.↓	RMSE↓	5.0°↑	7.5°↑	11.25°↑	22.5°↑	30.0°↑	Time↓
	✓	✓	4.29	0.24	14.03	86.51	89.06	91.40	94.61	95.65	38.2
✓			5.38	0.28	16.16	83.59	86.65	89.29	93.14	94.37	**35.0**
✓	✓		4.33	**0.20**	14.36	87.20	89.50	91.63	94.50	95.45	41.4
✓	✓	✓	**4.17**	**0.20**	**13.94**	**87.40**	**89.73**	**91.87**	**94.73**	**95.68**	41.5

Table 4: Comparison of different k-nearest neighbor searches in the proposed method. Time is reported in milliseconds.

Method	Mean↓	Med.↓	RMSE↓	5.0°↑	7.5°↑	11.25°↑	22.5°↑	30.0°↑	Time↓
Sequential k-NN	4.17	0.20	13.94	87.40	89.73	91.87	94.73	95.68	**41.5**
Spatial k-NN	**3.70**	**0.19**	**12.84**	**88.60**	**90.90**	**92.83**	**95.42**	**96.27**	495.2

Table 5: Impact of difficulty.

Class	Low	Mid.	High
Sequential k-NN	1.25	2.03	9.23
Spatial k-NN	1.09	1.80	8.21
Deg. rate [%]	14.7	12.8	12.4

Table 6: Sensitivity analysis of k for the SLA module. Time is reported in milliseconds.

k	3	5	7	9	11	13	15
Mean ↓	5.04	4.66	4.32	4.17	4.12	4.08	4.03
Time ↓	37.6	38.9	40.4	41.5	43.9	46.1	51.0

makes it difficult to capture local structures, resulting in significant performance degradation. Performance decreases when VFR is not included, due to the lack of variability information. The proposed method including all components achieves the best estimation accuracy, confirming the effectiveness of these components.

Effect of k-NN Search. In the proposed method, the sequential k-NN search can be replaced with a spatial k-NN search. Table 4 compares the performance of surface normal estimation when using these k-NN searches. When spatial k-NN search is adopted, while the average inference time increases significantly, the estimation accuracy improves. This is because the spatial proximity of the k-nearest neighbors is improved by the spatially exact neighbor search. Therefore, the proposed method can further improve estimation accuracy by adopting the spatial k-NN search when inference time is not a primary constraint.

Impact on Challenging Cases. We also quantitatively evaluate the impact of sequential k-NN search on performance in geometrically challenging cases. For this purpose, we classified the points into Low, Middle, and High difficulty classes based on the curvature calculated with PCA. Table 5 compares the Mean metric of sequential k-NN search and spatial k-NN search. The degradation rate of sequential k-NN search is lowest in the High class, confirming its adverse impact on accuracy in geometrically challenging cases is not particularly significant.

Analysis of k for SLA Module. We analyze the sensitivity of k for the SLA module. Table 6 shows the Mean and Time metrics for different k values, indicating a trade-off between accuracy and inference speed. We chose $k = 9$ since the improvement in accuracy becomes less significant beyond this point.

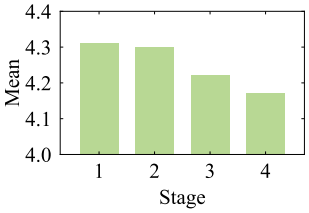

Fig. 7: Stage of VFR.

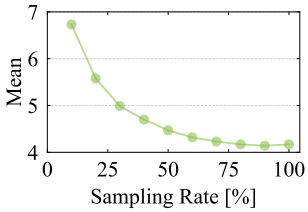

Fig. 8: Density variation.

Table 7: Robustness.

Setting	Mean↓
Default	4.17
Z-order [51]	4.23
Rotation	4.25
Noise	4.32

Stage of VFR Module. We evaluate the impact of introducing the VFR module at different stages. Fig. 7 shows the Mean metric when the VFR module is introduced in each of stages 1 to 4. The results show that the best performance is achieved in the fourth stage, supporting the effectiveness of our network design.
Variation of Point Density. We evaluate performance under varying densities. Fig. 8 shows the Mean metric when random sampling is applied to the test set at different rates. The proposed method achieves a Mean metric of 5.0 or less at sampling rates down to 30%, confirming its robustness to varying point densities.
Robustness Evaluation. Table 7 summarizes the robustness evaluation under different settings. The first row represents the default settings. The second row specifies a variant that uses the Z-order curve [51] instead of the Hilbert curve [21] as the serialization pattern. The third and fourth rows represent settings that apply random rotation around the vertical axis and Gaussian noise with a standard deviation of $\sigma = 0.01$ to the test set, respectively. The results demonstrate the robustness of the proposed method against these factors.

4.4 Evaluation on Surface Reconstruction

We further evaluate the effectiveness of the proposed method on surface reconstruction, an important downstream task of surface normal estimation. For surface reconstruction, we adopt NKSR [24], a neural surface reconstruction framework that takes point clouds associated with surface normals as input. We use the pre-trained NKSR model provided in the official repository. Following [24], we conduct experiments on the CARLA (Original) and CARLA (Novel) datasets, which consist of LiDAR point clouds captured in outdoor scenes. Quantitative evaluation metrics include Chamfer Distance based on the L1 norm (CD-L1), Normal Consistency (NC), and F-Score (FS).

Table 8 summarizes the surface reconstruction performance using surface normals estimated by the baseline methods and the proposed method. The final row of this table reports the results obtained when using the GT normals as input, provided as a reference. The proposed method achieves the best surface reconstruction performance that is closest to the results of GT normals, highlighting the high accuracy of surface normal estimation. Fig. 9 shows the reconstructed surfaces for qualitative evaluation. The colors of mesh faces represent their normal vectors mapped to RGB values. It can be seen that the proposed method reconstructs smooth surfaces with high fidelity, confirming its effectiveness.

Table 8: Surface reconstruction performance using estimated surface normals. CD-L1: Chamfer Distance based on the L1 norm. NC: Normal Consistency. FS: F-Score

	CARLA (Original)			CARLA (Novel)		
Method	CD-L1↓	NC↑	FS↑	CD-L1↓	NC↑	FS↑
MSECNet [72]	0.0374	0.939	0.927	0.0322	0.954	0.948
LiSu [47]	0.0362	0.948	0.932	0.0307	0.965	0.952
Ours	**0.0357**	**0.951**	**0.934**	**0.0306**	**0.966**	**0.953**
GT normals	0.0340	0.960	0.941	0.0290	0.974	0.961

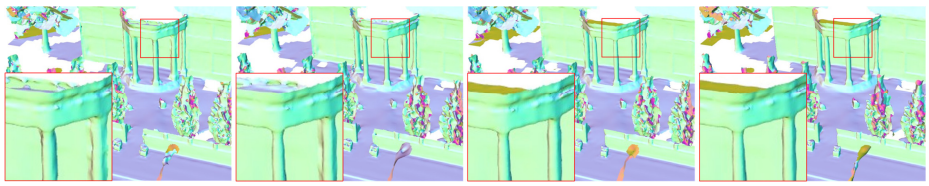

Fig. 9: Visualization of surface reconstruction results using estimated surface normals.

5 Conclusion

In this paper, we propose ESNE, a novel surface normal estimation method for LiDAR point clouds. The proposed method enables efficient and accurate surface normal estimation from point clouds converted to a sequence data format with a meticulously designed network architecture. Furthermore, the proposed method enhances local shape representations through feature refinement under the guidance of structural variability in local regions. Quantitative and qualitative evaluation experiments demonstrate that the proposed method achieves state-of-the-art performance in both efficiency and accuracy. The evaluation results on generalization to real-world data and surface reconstruction also support the robustness of the proposed method.

The proposed method has some limitations, which also suggest directions to be explored in future work. Although the sequential k-NN preserves a certain degree of spatial proximity, it does not guarantee an accurate identification of neighboring points. In addition, the proposed method still has room for acceleration through more effective placement and configuration of its constituent modules. We will address these limitations to improve performance and work on estimation for broader types of point clouds.

Acknowledgements

A part of this work is based on results obtained from the project, "Research and Development Project of the Enhanced Infrastructures for Post-5G Information and Communication Systems" (JPNP20017), subsidized by the New Energy and Industrial Technology Development Organization (NEDO).

References

1. Alexa, M., Behr, J., Cohen-Or, D., Fleishman, S., Levin, D., Silva, C.T.: Point set surfaces. In: Proceedings of the Visualization. pp. 21–29. IEEE (2001)
2. Ba, J.L., Kiros, J.R., Hinton, G.E.: Layer normalization. In: Advances in Neural Information Processing Systems 2016 Deep Learning Symposium (2016)
3. Ben-Shabat, Y., Gould, S.: DeepFit: 3D surface fitting via neural network weighted least squares. In: Proceedings of the European Conference on Computer Vision. pp. 20–34. Springer (2020)
4. Ben-Shabat, Y., Lindenbaum, M., Fischer, A.: 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. pp. 10112–10120 (2019)
5. Cao, J., Zhu, H., Bai, Y., Zhou, J., Pan, J., Su, Z.: Latent tangent space representation for normal estimation. IEEE Transactions on Industrial Electronics **69**(1), 921–929 (2021)
6. Cazals, F., Pouget, M.: Estimating differential quantities using polynomial fitting of osculating jets. Computer Aided Geometric Design **22**(2), 121–146 (2005)
7. Chen, X., Milioto, A., Palazzolo, E., Giguere, P., Behley, J., Stachniss, C.: Suma++: Efficient LiDAR-based semantic SLAM. In: Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems. pp. 4530–4537. IEEE (2019)
8. Chollet, F.: Xception: Deep learning with depthwise separable convolutions. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 1251–1258 (2017)
9. Deng, H., Birdal, T., Ilic, S.: PPFNet: Global context aware local features for robust 3D point matching. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 195–205 (2018)
10. Dosovitskiy, A., Ros, G., Codevilla, F., Lopez, A., Koltun, V.: CARLA: An open urban driving simulator. In: Proceedings of the Conference on Robot Learning. pp. 1–16. PMLR (2017)
11. Du, H., Yan, X., Wang, J., Xie, D., Pu, S.: Rethinking the approximation error in 3D surface fitting for point cloud normal estimation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 9486–9495 (2023)
12. Eldar, Y., Lindenbaum, M., Porat, M., Zeevi, Y.Y.: The farthest point strategy for progressive image sampling. IEEE Transactions on Image Processing **6**(9), 1305–1315 (1997)
13. Elfwing, S., Uchibe, E., Doya, K.: Sigmoid-weighted linear units for neural network function approximation in reinforcement learning. Neural Networks **107**, 3–11 (2018)
14. Fukushima, K.: Visual feature extraction by a multilayered network of analog threshold elements. IEEE Transactions on Systems Science and Cybernetics **5**(4), 322–333 (1969)
15. Gu, A., Dao, T.: Mamba: Linear-time sequence modeling with selective state spaces. In: Proceedings of the Conference on Language Modeling (2024)
16. Guennebaud, G., Gross, M.: Algebraic point set surfaces. In: ACM SIGGRAPH Papers, pp. 23–es (2007)
17. Guerrero, P., Kleiman, Y., Ovsjanikov, M., Mitra, N.J.: PCPNet learning local shape properties from raw point clouds. In: Proceedings of the Computer Graphics Forum. vol. 37, pp. 75–85. Wiley Online Library (2018)

18. Guo, M.H., Cai, J.X., Liu, Z.N., Mu, T.J., Martin, R.R., Hu, S.M.: PCT: Point cloud transformer. Computational Visual Media **7**, 187–199 (2021)
19. Han, X., Tang, Y., Wang, Z., Li, X.: Mamba3D: Enhancing local features for 3D point cloud analysis via state space model. In: Proceedings of the ACM International Conference on Multimedia. pp. 4995–5004 (2024)
20. Hendrycks, D., Gimpel, K.: Gaussian error linear units (GELUs). arXiv preprint arXiv:1606.08415 (2016)
21. Hilbert, D.: Über die stetige abbildung einer linie auf ein flächenstück. In: Dritter Band: Analysis· Grundlagen der Mathematik· Physik Verschiedenes: Nebst Einer Lebensgeschichte, pp. 1–2. Springer (1935)
22. Hoppe, H., DeRose, T., Duchamp, T., McDonald, J., Stuetzle, W.: Surface reconstruction from unorganized points. In: Proceedings of the Annual Conference on Computer Graphics and Interactive Techniques. pp. 71–78 (1992)
23. Huang, H., Li, D., Zhang, H., Ascher, U., Cohen-Or, D.: Consolidation of unorganized point clouds for surface reconstruction. ACM Transactions on Graphics **28**(5), 1–7 (2009)
24. Huang, J., Gojcic, Z., Atzmon, M., Litany, O., Fidler, S., Williams, F.: Neural kernel surface reconstruction. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 4369–4379 (2023)
25. Ioffe, S., Szegedy, C.: Batch normalization: Accelerating deep network training by reducing internal covariate shift. In: Proceedings of the International Conference on Machine Learning. pp. 448–456. pmlr (2015)
26. Kazhdan, M., Bolitho, M., Hoppe, H.: Poisson surface reconstruction. In: Proceedings of the Eurographics Symposium on Geometry Processing. vol. 7, pp. 61–70 (2006)
27. Kazhdan, M., Hoppe, H.: Screened poisson surface reconstruction. ACM Transactions on Graphics **32**(3), 1–13 (2013)
28. Lange, C., Polthier, K.: Anisotropic smoothing of point sets. Computer Aided Geometric Design **22**(7), 680–692 (2005)
29. Lenssen, J.E., Osendorfer, C., Masci, J.: Deep iterative surface normal estimation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 11247–11256 (2020)
30. Levin, D.: The approximation power of moving least-squares. Mathematics of Computation **67**(224), 1517–1531 (1998)
31. Li, D., Guan, J., Chen, Z., Liao, J., Du, J.: PointSSM: State space model for large-scale LiDAR point cloud semantic segmentation. International Journal of Applied Earth Observation and Geoinformation **144**, 104830 (2025)
32. Li, K., Zhao, M., Wu, H., Yan, D.M., Shen, Z., Wang, F.Y., Xiong, G.: GraphFit: Learning multi-scale graph-convolutional representation for point cloud normal estimation. In: Proceedings of the European Conference on Computer Vision. pp. 651–667. Springer (2022)
33. Li, Q., Feng, H., Shi, K., Fang, Y., Liu, Y.S., Han, Z.: Neural gradient learning and optimization for oriented point normal estimation. In: Proceedings of the SIGGRAPH Asia Conference Papers. pp. 1–9 (2023)
34. Li, Q., Feng, H., Shi, K., Gao, Y., Fang, Y., Liu, Y.S., Han, Z.: SHS-Net: Learning signed hyper surfaces for oriented normal estimation of point clouds. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 13591–13600 (2023)
35. Li, Q., Feng, H., Shi, K., Gao, Y., Fang, Y., Liu, Y.S., Han, Z.: Learning signed hyper surfaces for oriented point cloud normal estimation. IEEE Transactions on Pattern Analysis and Machine Intelligence **46**(12), 9957–9974 (2024)

36. Li, Q., Liu, Y.S., Cheng, J.S., Wang, C., Fang, Y., Han, Z.: HSurf-Net: Normal estimation for 3D point clouds by learning hyper surfaces. Advances in Neural Information Processing Systems **35**, 4218–4230 (2022)
37. Li, S., Zhou, J., Ma, B., Liu, Y.S., Han, Z.: NeAF: Learning neural angle fields for point normal estimation. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 37, pp. 1396–1404 (2023)
38. Li, Y., Bu, R., Sun, M., Wu, W., Di, X., Chen, B.: PointCNN: Convolution on x-transformed points. Advances in Neural Information Processing Systems **31**, 828–838 (2018)
39. Li, Z., Ai, Y., Lu, J., Wang, C., Deng, J., Chang, H., Liang, Y., Yang, W., Zhang, S., Zhang, T.: Pamba: Enhancing global interaction in point clouds via state space model. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 39, pp. 5092–5100 (2025)
40. Liang, D., Zhou, X., Xu, W., Zhu, X., Zou, Z., Ye, X., Tan, X., Bai, X.: Point-Mamba: A simple state space model for point cloud analysis. Advances in Neural Information Processing Systems **37**, 32653–32677 (2024)
41. Lin, Y., Yan, Z., Huang, H., Du, D., Liu, L., Cui, S., Han, X.: FPConv: Learning local flattening for point convolution. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 4293–4302 (2020)
42. Lin, Z.H., Huang, S.Y., Wang, Y.C.F.: Convolution in the cloud: Learning deformable kernels in 3D graph convolution networks for point cloud analysis. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 1800–1809 (2020)
43. Liu, J., Han, J., Liu, L., Aviles-Rivero, A.I., Jiang, C., Liu, Z., Wang, H.: Mamba4D: Efficient 4D point cloud video understanding with disentangled spatial-temporal state space models. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 17626–17636 (2025)
44. Loshchilov, I., Hutter, F.: SGDR: Stochastic gradient descent with warm restarts. In: Proceedings of the International Conference on Learning Representations (2017)
45. Loshchilov, I., Hutter, F.: Decoupled weight decay regularization. In: Proceedings of the International Conference on Learning Representations (2019)
46. Lu, D., Lu, X., Sun, Y., Wang, J.: Deep feature-preserving normal estimation for point cloud filtering. Computer-Aided Design **125**, 102860 (2020)
47. Malić, D., Fruhwirth-Reisinger, C., Schulter, S., Possegger, H.: LiSu: A dataset and method for LiDAR surface normal estimation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 17039–17049 (2025)
48. Matsuzaki, K., Nonaka, K.: Point cloud sampling preserving local geometry for surface reconstruction. In: Proceedings of the British Machine Vision Conference. pp. 1–14 (2023)
49. Moenning, C., Dodgson, N.A.: Fast marching farthest point sampling. Tech. rep., University of Cambridge, Computer Laboratory (2003)
50. Moosmann, F., Pink, O., Stiller, C.: Segmentation of 3D LiDAR data in non-flat urban environments using a local convexity criterion. In: Proceedings of the IEEE Intelligent Vehicles Symposium. pp. 215–220. IEEE (2009)
51. Morton, G.M.: A computer oriented geodetic data base and a new technique in file sequencing. International Business Machines Company (1966)
52. Nair, V., Hinton, G.E.: Rectified linear units improve restricted Boltzmann machines. In: Proceedings of the International Conference on Machine Learning. pp. 807–814 (2010)

53. Park, J., Zhou, Q.Y., Koltun, V.: Colored point cloud registration revisited. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 143–152 (2017)
54. Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Köpf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., Chintala, S.: PyTorch: An imperative style, high-performance deep learning library. Advances in Neural Information Processing Systems **32**, 8024–8035 (2019)
55. Peano, G.: Sur une courbe, qui remplit toute une aire plane. In: Arbeiten zur Analysis und zur mathematischen Logik, pp. 71–75. Springer (1990)
56. Qi, C.R., Su, H., Mo, K., Guibas, L.J.: PointNet: Deep learning on point sets for 3D classification and segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 652–660 (2017)
57. Ronneberger, O., Fischer, P., Brox, T.: U-Net: Convolutional networks for biomedical image segmentation. In: Proceedings of the International Conference on Medical Image Computing and Computer-Assisted Intervention. pp. 234–241. Springer (2015)
58. Serafin, J., Grisetti, G.: NICP: Dense normal based point cloud registration. In: IEEE/RSJ International Conference on Intelligent Robots and Systems. pp. 742–749. IEEE (2015)
59. Simonovsky, M., Komodakis, N.: Dynamic edge-conditioned filters in convolutional neural networks on graphs. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 3693–3702 (2017)
60. Sun, P., Kretzschmar, H., Dotiwalla, X., Chouard, A., Patnaik, V., Tsui, P., Guo, J., Zhou, Y., Chai, Y., Caine, B., et al.: Scalability in perception for autonomous driving: Waymo open dataset. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 2446–2454 (2020)
61. Sun, P., Tan, M., Wang, W., Liu, C., Xia, F., Leng, Z., Anguelov, D.: SWFormer: Sparse window transformer for 3D object detection in point clouds. In: Proceedings of the European Conference on Computer Vision. pp. 426–442. Springer (2022)
62. Thomas, H., Qi, C.R., Deschaud, J.E., Marcotegui, B., Goulette, F., Guibas, L.J.: KPConv: Flexible and deformable convolution for point clouds. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 6411–6420 (2019)
63. Wang, C., Zha, Y., Yang, W., Zhang, T.: StruMamba3D: Exploring structural mamba for self-supervised point cloud representation learning. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 28546–28555 (2025)
64. Wang, H., Zhao, M., Quan, W., Chen, Z., Yan, D.M., Wonka, P.: E^3-Net: Efficient E(3)-equivariant normal estimation network. IEEE Transactions on Visualization and Computer Graphics **32**(2), 2230–2242 (2026)
65. Wang, P.S.: OctFormer: Octree-based transformers for 3D point clouds. ACM Transactions on Graphics **42**(4), 1–11 (2023)
66. Wang, Y., Sun, Y., Liu, Z., Sarma, S.E., Bronstein, M.M., Solomon, J.M.: Dynamic graph CNN for learning on point clouds. ACM Transactions on Graphics **38**(5), 1–12 (2019)
67. Wu, P., Chai, B., Zheng, M., Li, W., Hu, Z., Chen, J., Zhang, Z., Li, H., Sun, X.: Efficient spiking point mamba for point cloud analysis. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 26393–26403 (2025)

68. Wu, W., Qi, Z., Fuxin, L.: PointConv: Deep convolutional networks on 3D point clouds. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 9621–9630 (2019)
69. Wu, X., Jiang, L., Wang, P.S., Liu, Z., Liu, X., Qiao, Y., Ouyang, W., He, T., Zhao, H.: Point transformer v3: Simpler faster stronger. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 4840–4851 (2024)
70. Wu, X., Lao, Y., Jiang, L., Liu, X., Zhao, H.: Point transformer v2: Grouped vector attention and partition-based pooling. Advances in Neural Information Processing Systems **35**, 33330–33342 (2022)
71. Wu, Y., Zhao, M., Li, K., Quan, W., Yu, T., Yang, J., Jia, X., Yan, D.M.: CMG-Net: Robust normal estimation for point clouds via Chamfer normal distance and multi-scale geometry. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 38, pp. 6171–6179 (2024)
72. Xiu, H., Liu, X., Wang, W., Kim, K.S., Matsuoka, M.: MSECNet: Accurate and robust normal estimation for 3D point clouds by multi-scale edge conditioning. In: Proceedings of the ACM International Conference on Multimedia. pp. 2535–2543 (2023)
73. Xu, Q., Sun, X., Wu, C.Y., Wang, P., Neumann, U.: Grid-GCN for fast and scalable point cloud learning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 5661–5670 (2020)
74. Yang, Y.Q., Guo, Y.X., Xiong, J.Y., Liu, Y., Pan, H., Wang, P.S., Tong, X., Guo, B.: Swin3D: A pretrained transformer backbone for 3D indoor scene understanding. Computational Visual Media **11**(1), 83–101 (2025)
75. Zhang, G., Fan, L., He, C., Lei, Z., Zhang, Z., Zhang, L.: Voxel mamba: Group-free state space models for point cloud based 3D object detection. Advances in Neural Information Processing Systems **37**, 81489–81509 (2024)
76. Zhang, J., Cao, J.J., Zhu, H.R., Yan, D.M., Liu, X.P.: Geometry guided deep surface normal estimation. Computer-Aided Design **142**, 103119 (2022)
77. Zhang, T., Yuan, H., Qi, L., Zhang, J., Zhou, Q., Ji, S., Yan, S., Li, X.: Point cloud mamba: Point cloud learning via state space model. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 39, pp. 10121–10130 (2025)
78. Zhao, H., Jiang, L., Jia, J., Torr, P., Koltun, V.: Point transformer. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 16259–16268 (2021)
79. Zheng, X., Zhu, J.: Efficient LiDAR odometry for autonomous driving. IEEE Robotics and Automation Letters **6**(4), 8458–8465 (2021)
80. Zhou, J., Jin, W., Wang, M., Liu, X., Li, Z., Liu, Z.: Improvement of normal estimation for point clouds via simplifying surface fitting. Computer-Aided Design **161**, 103533 (2023)
81. Zhu, R., Liu, Y., Dong, Z., Wang, Y., Jiang, T., Wang, W., Yang, B.: AdaFit: Rethinking learning-based normal estimation on point clouds. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 6118–6127 (2021)