OXFORD CDE

Journal of Computational Design and Engineering, 2023, **10**, 2332–2344

DOI: 10.1093/jcde/qwad102
Advance access publication date: 13 November 2023
Research Article

Improved semantic segmentation network using normal vector guidance for LiDAR point clouds

Minsung Kim [1], Inyoung Oh[1], Dongho Yun[2] and Kwanghee Ko [1,*]

[1]The School of Mechanical Engineering, Gwangju Institute of Science and Technology, 123 Cheomdangwagiro, Gwangju 61005, Republic of Korea
[2]Automotive Mobility Materials and Components R&D Group, Korea Institute of Industrial Technology, Gwangju 61012, Republic of Korea
*Correspondence: khko@gist.ac.kr

Abstract

As Light Detection and Ranging (LiDAR) sensors become increasingly prevalent in the field of autonomous driving, the need for accurate semantic segmentation of three-dimensional points grows accordingly. To address this challenge, we propose a novel network model that enhances segmentation performance by utilizing normal vector information. Firstly, we present a method to improve the accuracy of normal estimation by using the intensity and reflection angles of the light emitted from the LiDAR sensor. Secondly, we introduce a novel local feature aggregation module that integrates normal vector information into the network to improve the performance of local feature extraction. The normal information is closely related to the local structure of the shape of an object, which helps the network to associate unique features with corresponding objects. We propose four different structures for local feature aggregation, evaluate them, and choose the one that shows the best performance. Experiments using the SemanticKITTI dataset demonstrate that the proposed architecture outperforms both the baseline models, RandLA-Net, and other existing methods, achieving mean intersection over union of 57.9%. Furthermore, it shows highly competitive performance compared with RandLA-Net for small and dynamic objects in a real road environment. For example, it yielded 95.2% for cars, 47.4% for bicycles, 41.0% for motorcycles, 57.4% for bicycles, and 53.2% for pedestrians.

Keywords: normal vector estimation, semantic segmentation, LiDAR sensor, point cloud, local feature extraction, intensity

List of symbols

p :	Specific point in LiDAR point cloud
k :	Index for each point of NLFA module input ($k = 1, \ldots, N$).
i :	Index for each point of k-NN algorithm for p ($i = 1, \ldots, K$).
p_i :	ith-nearest point of point p
p^k :	kth point of NLFA module input
$\boldsymbol{p}$:	$(x,\ y,\ z)$ position of point p
$\boldsymbol{n}$:	$(n_x,\ n_y,\ n_z)$ normal of point p
$\boldsymbol{p}_i$:	$(x,\ y,\ z)$ position of point p_i
$\boldsymbol{n}_i$:	$(n_x,\ n_y,\ n_z)$ normal of point p_i
I :	Normalized LiDAR intensity at point p
I_i :	Normalized LiDAR intensity at point p_i
w_i^m :	Function to calculate the weight for intensity at point p_i
M :	Covariance matrix for the neighborhood points p_i
M' :	Weighted covariance matrix for the neighborhood points p_i
$MLP(\cdot)$:	Multi-layer perceptron to extract feature
$\boldsymbol{r}_i^k$:	Encoded redundant point position of neighbor i at p^k
$\boldsymbol{l}_i^k$:	Encoded redundant point normal of neighbor i at p^k
$g(\cdot)$:	Shared MLP followed by the softmax function
W :	learnable weights of the shared MLP
f^k :	Extracted local feature of point p^k
f_i^k :	Extracted local feature of neighbor i at p^k
s_i^k :	Score mask for fittering feature of neighbor i at p^k
C :	Number of classes
TP_c :	Numbers of true positive predictions for each classes
FP_c :	Numbers of false positive predictions for each classes
TN_c :	Numbers of true negative predictions for each classes
FN_c :	Numbers of false negative predictions for each classes
L_{ce} :	Cross entropy loss of neural network
z_i :	ith values of neural network output
z_j :	jth values of neural network output
FOV :	Vertical field of view of the LiDAR sensor

1. Introduction

To advance the field of autonomous driving and facilitate its practical applications, a wide spectrum of research initiatives is currently underway. Addressing the complexity and safety challenges of urban autonomous driving, Noh and An (2022) introduced a reliable risk assessment framework tested in real-world urban conditions. Meanwhile, Eom and Lee (2022) found that a function-centered interface with visual and auditory feedback significantly improves driver mode awareness for vehicles at different automation levels. Research aimed at enhancing the performance and robustness of autonomous driving through the use of LiDAR sensors is also actively conducted. The LiDAR sensors excel in delivering high-precision 360-degree distance measurements, functioning effectively in low-light conditions, and generating three-dimensional (3D) maps, making them a crucial component for

Received: September 11, 2023. **Revised:** November 7, 2023. **Accepted:** November 7, 2023

autonomous driving. They generate 3D points that contain only geometric information about objects. Therefore, it is necessary to recognize which point belongs to which object. This step is called semantic segmentation, which assigns the label of an object to the corresponding points. Various attempts have been made to address this problem, and a deep learning-based approach has achieved remarkable success.

3D point semantic segmentation can be broadly categorized into two main approaches: projection- and point-based methods. Based on the projection method, the projection-based approach can be further divided into multi-view, spatial, volumetric, permutohedral, and hybrid methods. On the other hand, the point-based approach can be divided into pointwise Multi-Layer Perceptron (MLP), point convolution, Recurrent Neural Network (RNN)-based, and graph-based methods. This paper reviewed related papers based on two broad categories, projection- and 3D point-based, to make the presentation concise.

1.1. Projection-based methods

Convolutional Neural Network (CNN)-based deep learning models have proven successful for 2D image segmentation. Many researchers have attempted to apply these models to 3D point segmentation. In these approaches, 3D points are transformed into 2D images and used as input to the network. The transformation is given as equation (1) (Wu, Wan, *et al.*, 2018; Wu, Zhou, *et al.*, 2019). Each point $(x,\ y, z)$ is converted via a mapping to spherical coordinates $(u,\ v)$ and finally to image coordinates, as defined by

$$\begin{pmatrix} u \\ v \end{pmatrix} = \begin{pmatrix} \frac{1}{2}\left[1 - \arctan\left(y, x\right)\pi^{-1}\right] w \\ \left[1 - \left(\arcsin\left(zr^{-1}\right) + FOV_{\text{up}}\right)FOV^{-1}\right] h \end{pmatrix}, \tag{1}$$

where $(w,\ h)$ are the height and width of the image representation of the 3D point cloud, $FOV = FOV_{\text{up}} + FOV_{\text{down}}$ is the vertical field of view of the LiDAR sensor, and r is the depth of a point.

Wu *et al.* (Wu, Wan, *et al.*, 2018; Wu, Zhou, *et al.*, 2019) present a fully convolutional encoder-decoder neural network through generating a 360-degree image from a point cloud via spherical projection and predicted semantic labels "achieving a mean intersection over union (mIoU) score of 37.2% and 44.9% in a benchmark test". Milioto *et al.* (2019) proposed a high-performance architecture called RangeNet, based on Darknet53 (Redmon & Farhadi, 2018) backbone, with k-Nearest Neighbor (k-NN)-based noise elimination as post-processing. It achieved an mIoU score of 52.5% and showed high performance with a speed of 12 fps. However, projecting 3D points to 2D images and re-projecting the predicted 2D labels to the point cloud can cause information loss and errors, ultimately compromising segmentation performance.

Voxel-based approaches offer another solution to the problem of 3D point segmentation. Voxels are widely used to handle complex 3D shapes in rendering, segmentation, and reconstruction. Various approaches have been proposed to address the challenge of 3D point segmentation using voxels. In 3D U-Net (Çiçek *et al.*, 2016), a voxel-based convolutional segmentation network for general 3D point clouds was introduced. Zhou and Tuzel (2018) presented an effective method to segment sparse LiDAR point clouds. Hilbig *et al.* (2023) achieved performance improvement by using a geometric feature called a signed distance field for a 3D voxel network. Another method, i.e., Deep FusionNet proposed by Zhang *et al.* (2020) aims to minimize information loss during voxelization by combining voxel and point features. Alternatively, a novel grid in polar form was proposed to consider more points inside than the conventional voxel-based method by leveraging the characteristics of LiDAR sensors. In addition, a ring CNN architecture, called PolarNet, was developed to process such a grid efficiently. It achieved an mIoU of 54.3% in the benchmark test (Zhang *et al.*, 2020). However, it required a lot of memory and computation time and also suffered from voxel projection errors. Despite its effectiveness, the voxel-based method may encounter errors during voxelization. One way to reduce this problem is to use smaller grids. However, it can increase the computational cost because of the 3D convolution step in the neural network.

1.2. Point-based methods

3D points can be directly utilized in segmentation. PointNet (Qi *et al.*, 2017a), the point-based neighborhood feature learning method, processes point clouds and extracts features through fully connected layers. PointNet++ (Qi *et al.*, 2017b) improved upon PointNet by employing hierarchical pooling and context representation. However, neither of these methods can handle large-scale point clouds obtained by LiDAR sensors due to the computation cost growing proportionally to the input size. A k-NN-based local feature extraction technique (Luo *et al.*, 2021) was proposed to enhance segmentation performance but still suffered from the computational cost issue. To solve this problem, Hu *et al.* (2020) proposed RandLA-Net, which uses a random sampling method and introduces a local feature aggregation module to process large point clouds and dramatically improve segmentation performance efficiently. In particular, it overcame the limitations of PointNet and achieved an mIoU of 53.9% in the benchmark test. SCF-Net (Fan *et al.*, 2021) presents a new local feature extraction method based on polar representation.

Benchmark tests have demonstrated that the point-based methods have the potential to be used for semantic segmentation in autonomous driving. However, they still need to improve the segmentation of various objects critical for driving safely, such as people and bicycles, which are small in size.

1.3. Contributions

Inspired by RandLA-Net, we proposed a neural network model that utilizes 3D points and surface normal vectors to enhance segmentation performance. Here, RandLA was selected as the base architecture due to its reputation for robust performance across different LiDAR sensor types. While 3D points represent the shape of an object, normal vectors provide information about the orientation of the object's local shape at each point. Using intensity information, we developed a novel normal estimation method from a point cloud. We introduced a normal local feature aggregation (NLFA) module that combines 3D points and normal vectors in the encoding process to extract local features that aid in segmentation.

The surface normal vectors are very useful for recognizing the features of objects within the LiDAR point cloud. It contains the orientation of the point, through which it is possible to obtain curvature information of the object. As a result, the object's appearance can be effectively grasped, and information on a small object with a significant change in curvature can be efficiently obtained. Moreover, we developed a principal component analysis (PCA)-based normal estimation method that utilizes the reflection intensity and physical characteristics of LiDAR sensors. Finally, we compared the proposed method with others using the SemanticKITTI datasets, and our results demonstrated better performance than other methods.

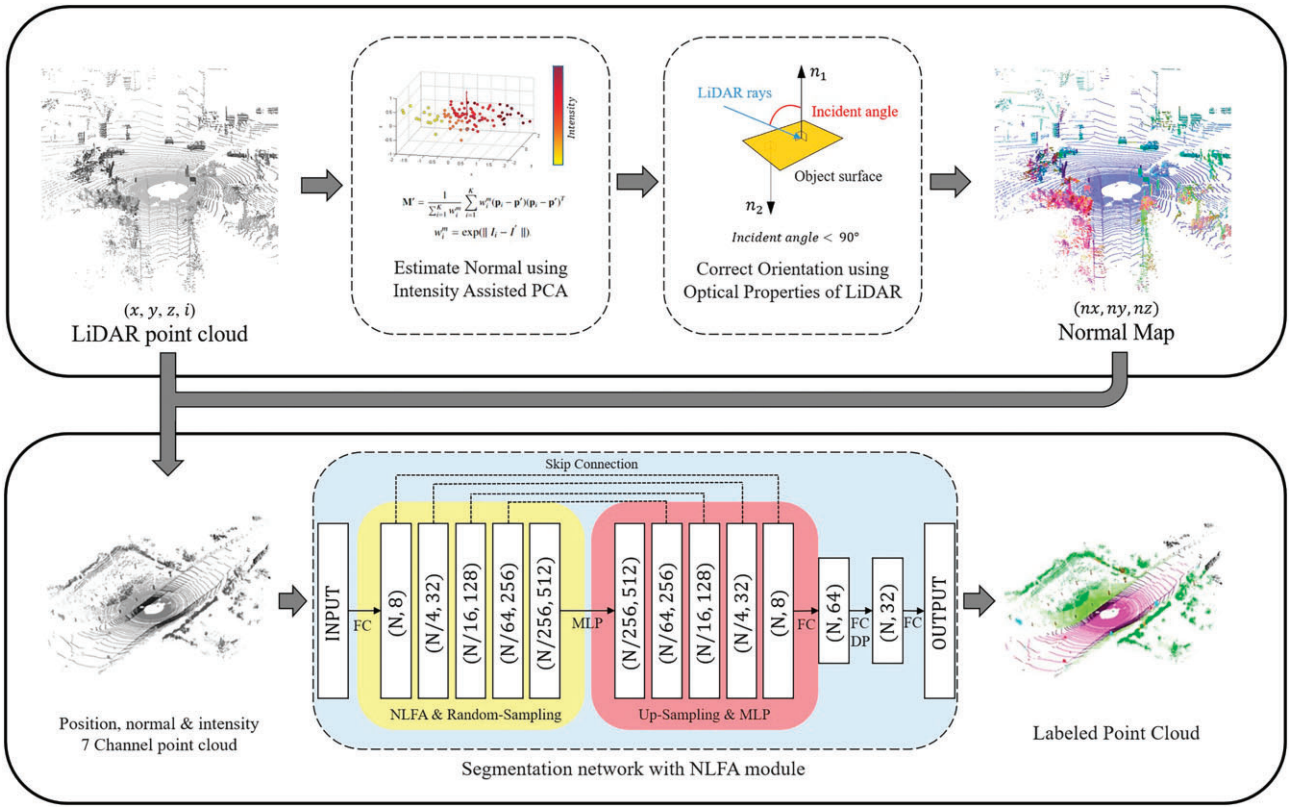


Figure 1: Framework of the proposed network.

The main contributions are 3-fold:

(i) We enhance the overall segmentation performance of 3D points by using a local feature aggregation module that incorporates both 3D points and normal vectors.
(ii) By utilizing normal features, we improve the recognition rate of small objects, such as humans, cars, and bicycles, which are important to autonomous driving.
(iii) We propose a novel normal estimation method that utilizes the physical properties of LiDAR sensors and intensity information to estimate surface normal vectors. Also, this method works robustly against irregular noise of LiDAR sensors.

2. Proposed Method

This section describes the proposed framework. The overall structure of the proposed method is shown in Fig. 1. A point cloud generated by a LiDAR is processed as input. The normal vectors at each point in the point cloud are estimated using the intensity-assisted method of normal vector estimation. A tuple is constructed using the position, normal vector, and intensity 7 channel $(x,\ y, z, n_x, n_y, n_z, i)$ value at each point. N tuples are then created and fed into the segmentation network. An encoder-decoder structure and a skip connection are used for the network. The front half of the network, enclosed in the dotted box, is an encoder. It consists of the first five layers next to the input layer, which include a novel NLFA module and a random sampling layer between the point sub-feature, and the decoder, composed of the next five layers following the encoder, has four up-samples with MLP. The header, located at the end of the network, has a simple Fully Connected (FC) layer with a dropout. An input of N points is given to the encoder. Each of the five layers of the encoder processes the input using a NLFA module and shrinks the input data size with a 4-fold decimation ratio by random sampling, while the dimension of extracted features at each point increases, assigning more features to the point. The decoder has a symmetric structure to the encoder, with two fully connected layers attached to its end, which produce N points labeled with n classes.

2.1. Intensity-assisted normal vector estimation

Normal vectors can be estimated from 3D points using traditional PCA (Hoppe *et al.*, 1992).

$$M = \sum_{i=1}^{K} (\boldsymbol{p}_i - \boldsymbol{p})\,(\boldsymbol{p}_i - \boldsymbol{p})^T. \tag{2}$$

Suppose that p is a point of interest and p_i $(i = 1, \ldots, K)$ are the points in the nearest neighborhood of p. Then, the normal vector at p corresponds to the eigenvector that has the highest eigenvalue of the covariance matrix M calculated for p_i. The covariance matrix is computed by equation (2).

However, the accuracy of this method for estimating normal vectors is sensitive to the value of K, the distribution pattern of points, and noise.

To improve the robustness of normal vector estimation, Park *et al.* (2020) proposed the weighted method utilizing LiDAR intensity information. Weights w_i^m are calculated by exponential of difference of I intensity for each neighborhood point, as shown in equation (3), and a weighted covariance matrix M' is computed using equation (4):

$$w_i^m = \exp\left(\|I_i - I\|\right) \tag{3}$$

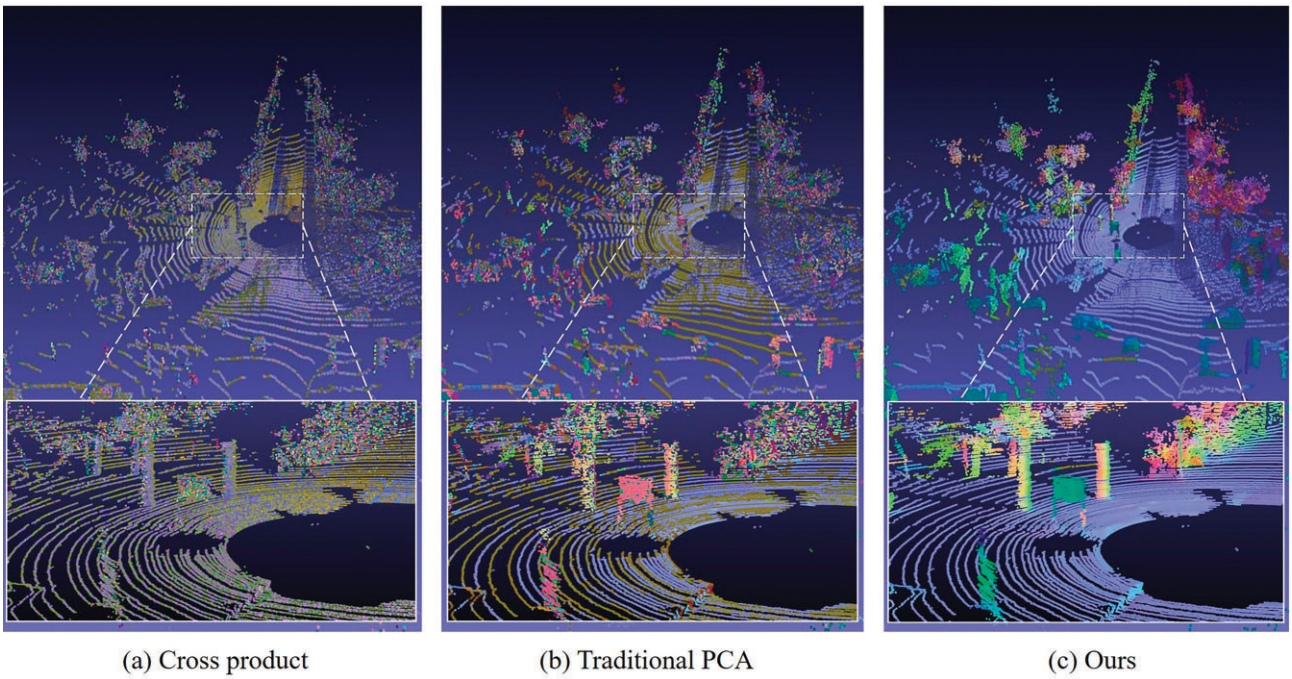


Figure 2: Normal estimation results in HSV color space on the KITTI Velodyne odometry dataset using (a) cross-product method, (b) PCA without intensity information, and (c) our proposed method considering the LiDAR features.

$$M' = \frac{1}{\sum_{i-1}^{K} w_i^m} \sum_{i=1}^{K} w_i^m \left(p_i - p\right)\left(p_i - p\right)^T. \tag{4}$$

Additionally, a median filter (Huber, 2004) is applied to remove the effects of noise from the estimated normal vector.

Next, normal vectors are selected using the following properties of LiDAR sensors. (i) A LiDAR sensor projects light, detects the reflected light, and generates a 3D point. Therefore, the angle between the direction of the normal vector and the reflection direction is less than 90 degrees. (ii) The LiDAR sensor is fixed on the z-axis to provide information about the reference vertical direction. Using these two characteristics, we can obtain highly accurate normal direction information. The proposed normal estimation method is compared with other methods, as shown in Fig. 2 below. In this figure, the direction of a normal vector is represented as a scalar value in the Hue, Saturation, Value (HSV) color space. Consequently, if normal vectors have identical directions in an area, the points in that region are painted with the same color.

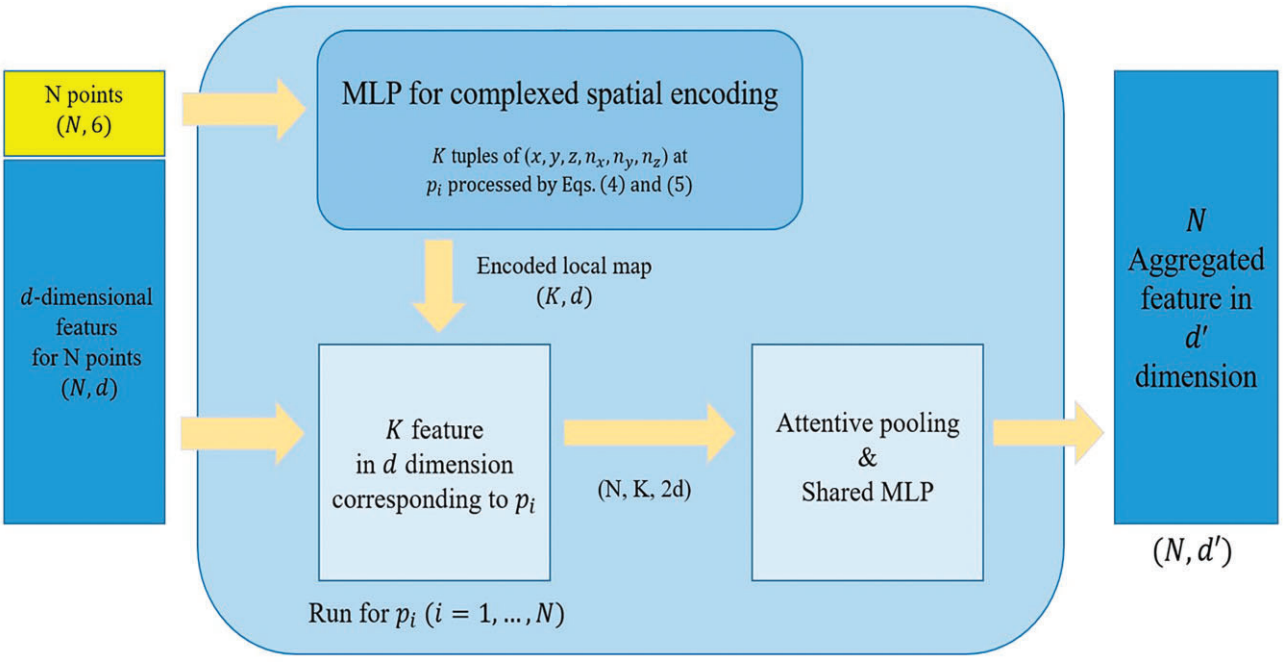


Figure 3: Structure of the NLFA module for local feature extraction using k-NN algorithm and six-channel point feature.

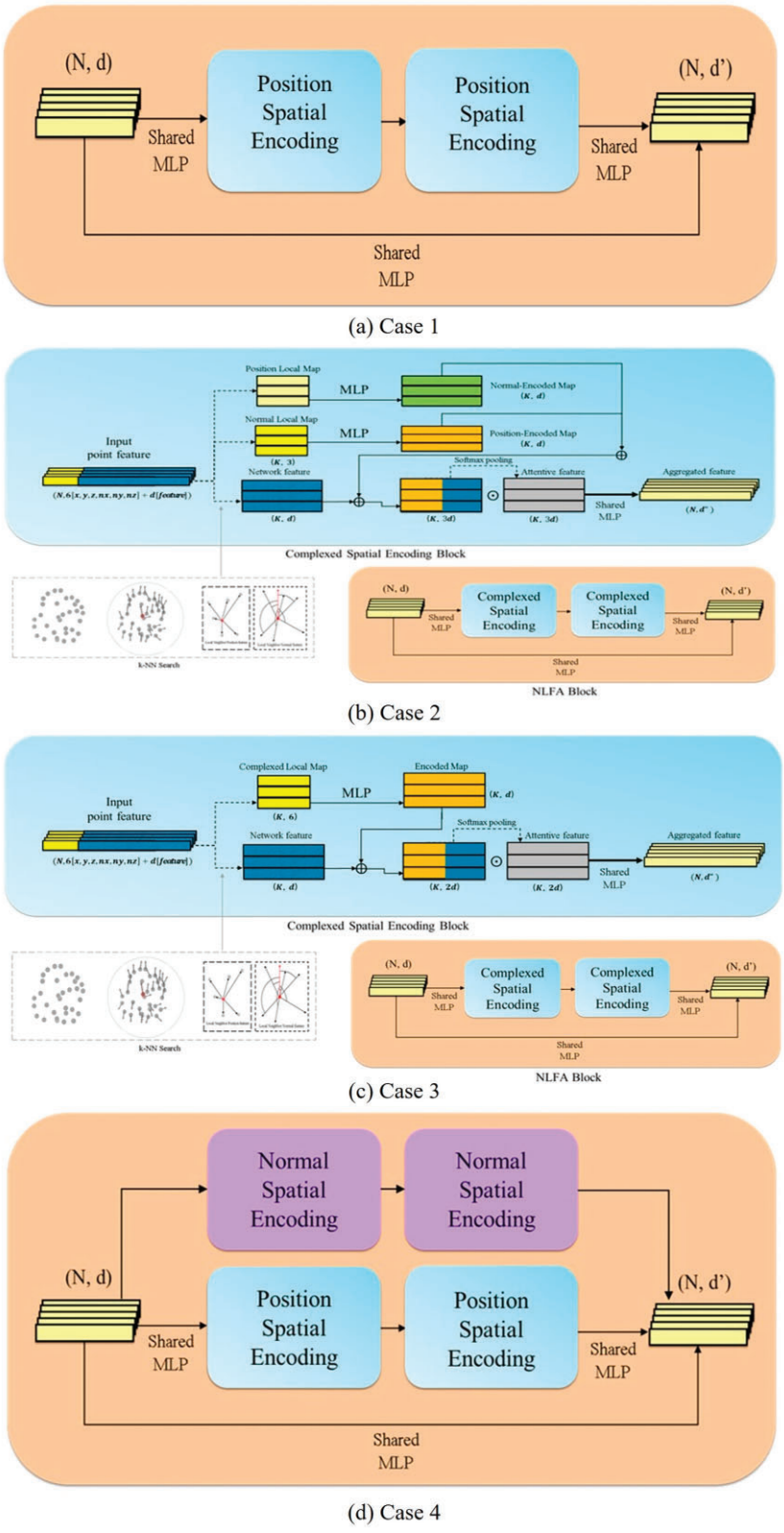


Figure 4: Detailed structure of each feature extraction module. (a) Traditional RandLA-Net with normal input, and (b)–(d) a complexed module including normal encoding.

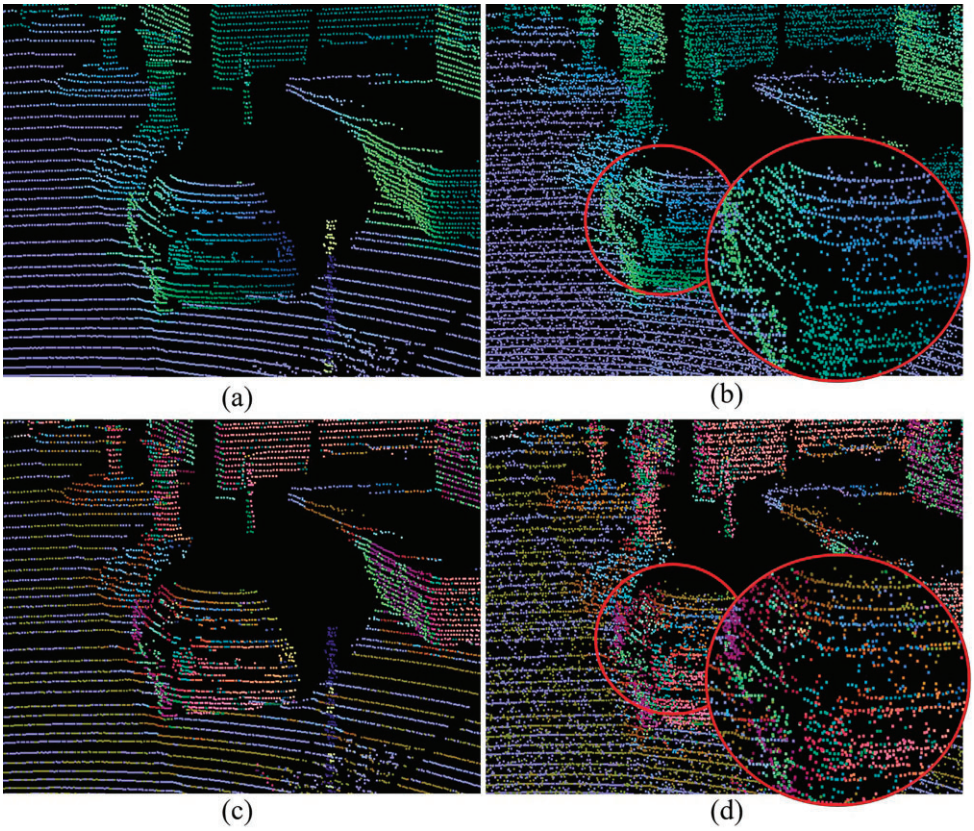


Figure 5: Normal estimation on LiDAR points with noise. (a) Our proposed normal estimation method, (b) our method with noise, (c) PCA, and (d) PCA with noise.

Figure 2a is the result of the cross-product of vectors formed by the positions of points, while Fig. 2b displays the normal vectors estimated by the PCA method. Figure 2c illustrates the normal vectors obtained using the proposed method, demonstrating that the points on each object are consistently painted with the same color. For example, consider the road in figure. The normal vectors at the points on the road should be oriented upward, resulting in the same or similar color being used to render the points. However, Fig. 2a and b show that some points are rendered with different colors, indicating inconsistent estimation of normal vectors. In contrast, the proposed method produces consistent normal vectors on the road, rendering the points in blue, as shown in figure.

2.2. NLFA module

Figure 3 illustrates the structure of the NLFA module. The NLFA module is performed twice between two layers of the encoder in Fig. 1, followed by random sampling. Therefore, d-dimensional feature vectors for N points are provided as input to the module. A complex spatial encoding scheme is introduced to incorporate normal information in the encoder, as shown in the figure.

Consider that the normal vector $\boldsymbol{n}_i = (\boldsymbol{n}_{xi}, \boldsymbol{n}_{yi}, \boldsymbol{n}_{zi})$ for $\boldsymbol{p}_i = (\boldsymbol{p}_{xi}, \boldsymbol{p}_{yi}, \boldsymbol{p}_{zi})$ has been computed using the proposed normal estimation method. The Euclidean distances from p to its neighbors are computed and sorted in an incremental order. Next, the K points from the sorted list's first point are selected to produce $\boldsymbol{p}_1^k, \ldots, \boldsymbol{p}_i^k$ and $\boldsymbol{n}_1^k, \ldots, \boldsymbol{n}_i^k$.

This selection can be efficiently performed using a k-NN algorithm. Here, $\boldsymbol{p}_j^k$ and $\boldsymbol{n}_j^k$ are the jth point and its normal vector associated with p^k. A local feature $(\boldsymbol{p}^k, \boldsymbol{p}_i^k, \boldsymbol{p}^k - \boldsymbol{p}_i^k, \|\boldsymbol{p}^k - \boldsymbol{p}_i^k\|)$ $(i = 1, \ldots, K)$ is constructed for relative position encoding. Here, $\boldsymbol{p}^k - \boldsymbol{p}_i^k$ is the relative position from $\boldsymbol{p}^k$ to $\boldsymbol{p}_i^k$, and $\|\boldsymbol{p}^k - \boldsymbol{p}_i^k\|$ is an impact factor. These terms are encoded using an MLP introduced in to yield the encoded redundant point position $\boldsymbol{r}_i^k$. Here, $\oplus$ is the concatenation operation:

$$\boldsymbol{r}_i^k = MLP\left(\boldsymbol{p}^k \oplus \boldsymbol{p}_i^k \oplus \left(\boldsymbol{p}^k - \boldsymbol{p}_i^k\right) \oplus \|\boldsymbol{p}^k - \boldsymbol{p}_i^k\|\right). \tag{5}$$

The normal vectors are used to form a local normal feature $(\boldsymbol{n}^k, \boldsymbol{n}_i^k, \boldsymbol{n}^k - \boldsymbol{n}_i^k, \boldsymbol{n}^k \cdot \boldsymbol{n}_i^k)$ $(i = 1, \ldots, K)$, where $\boldsymbol{n}^k - \boldsymbol{n}_i^k$ indicates the change of directions relative to $\boldsymbol{n}^k$ and $\boldsymbol{n}^k \cdot \boldsymbol{n}_i^k$ represents the similarity of the directions. These terms are encoded using the same MLP as position component to reduce the encoded redundant point normal $\boldsymbol{l}_i^k$, as shown in equation (6):

$$\boldsymbol{l}_i^k = MLP\left(\boldsymbol{n}^k \oplus \boldsymbol{n}_i^k \oplus \left(\boldsymbol{n}^k - \boldsymbol{n}_i^k\right) \oplus \boldsymbol{n}^k \cdot \boldsymbol{n}_i^k\right). \tag{6}$$

The computation of equations (4) and (5) is called complexed spatial encoding, which yields a local feature encoded in (k, d) shape. There are four different combinations of how to incorporate local position and normal features, which are further discussed in Section 3.3. Features in $(k, 2d)$ are obtained when combined with the network input. The encoding scheme is performed for each N point, producing N features in $(k, 2d)$ shape. Next, a softmax function is used to efficiently pool the attentive features from $(N, k, 2d)$ features using equations (7) and (8):

$$s_i^k = g\left(f_i^k, W\right) \tag{7}$$

$$f_i = \sum_{i=1}^{K}\left(f_i^k, s_i^k\right). \tag{8}$$

The score for each input feature is calculated as a mask s_i^k. g() consists of a shared MLP, followed by the softmax function. In addition, W represents the learnable weights of the shared MLP, and

Table 1: Experiment on the test set of SemanticKITTI.

Methods	Input	Size	mIoU	Road	Sidewalk	Parking	Other-ground	Building	Car	Truck	Bicycle	Motorcycle	Other-vehicle	Vegetation	Trunk	Terrain	Person	Bicyclist	Motorcyclist	Fence	Pole	Traffic-sign
SqueezeSeg	64 × 2048 pixels	0.9M	29.5	85.4	54.3	26.9	4.5	57.4	68.8	3.3	16.0	4.1	3.6	60.0	24.4	53.7	12.9	13.1	0.9	29.0	17.5	24.5
SqueezeSegV2		0.9M	39.7	88.6	67.6	45.8	17.7	73.7	81.8	13.4	18.5	17.9	14.0	71.8	35.8	60.2	20.1	25.1	3.9	41.1	20.2	36.5
Darknet21		25M	47.4	91.4	74.0	57.0	18.6	81.9	85.4	18.6	26.2	26.5	15.6	77.6	48.4	63.6	31.8	33.6	4.0	52.3	36.0	50.5
Darknet53		50M	49.9	91.8	74.6	64.8	25.5	84.1	86.4	25.5	24.6	32.7	22.6	78.3	50.1	64.0	36.2	33.6	4.7	55.0	38.9	52.2
RangeNet++		50M	52.2	91.8	75.2	65.0	25.7	87.4	91.4	25.7	25.7	34.4	23.0	80.5	55.1	64.6	38.3	38.8	4.8	58.6	47.9	55.9
PolarNet	500k vxs	14M	54.3	90.8	74.4	**61.7**	**28.5**	90.0	93.8	22.9	40.3	30.1	28.5	**84.0**	65.5	**67.8**	49.2	48.2	**7.2**	**61.3**	**51.8**	57.5
PoinNet	50k pts	3.5M	14.6	61.6	35.7	15.8	1.4	41.4	46.3	0.1	0.2	0.3	0.8	31.0	0.1	17.6	0.2	13.1	0.0	12.9	2.4	3.7
PoinNet++		6M	20.1	72.6	41.8	18.7	5.6	62.3	53.7	0.9	1,0	0,2	0.2	46.5	0.9	30.0	0.9	13.1	0.0	16.9	5.0	8.9
RandLA-Net		1.2M	53.9	90.7	73.7	60.3	20.4	86.9	94.2	40.1	26.0	25.8	38.9	81.4	**66.8**	49.2	49.2	48.2	**7.2**	56.3	49.2	47.7
Case 1	50k pts with normal	1.2M	56.0	90.4	73.5	56.8	22.1	86.5	94.8	**42.0**	41.4	36.8	**44.6**	79.9	60.4	66.5	50,6	54.4	4.6	55.4	46.2	57.5
Case 2		1.6M	56.9	91.1	75.2	60.6	23.0	87.3	94.9	40,1	42.7	38.7	42.5	81.3	59.7	67.5	50.1	56.8	4.9	57.7	48.0	58.9
Case 3		1.4M	**57.9**	**91.4**	**75.7**	60.5	20.5	88.0	**95.2**	39.9	**47.4**	**41.0**	41.1	81.2	61.1	67.4	**53.2**	**57.4**	6.8	59.0	50.0	**60.8**
Case 4		2.4M	56.8	90.9	74.5	59.5	10.8	88.9	95.0	38.6	47.0	37.0	41.8	81.5	60.8	67.3	51.3	56.5	6.4	60.2	49.1	60.7

f_i^k is the kth local features. The shared MLP produces N aggregated features in d' dimension (N, d').

2.3. Construction of NLFA structure

Four different structures of the network implementation were proposed, as shown below in Fig. 4.

Case 1 displays a RandLA-Net structure consisting of two position spatial encoding blocks using the position-based feature extraction method. At the module's start and end, the feature goes through shared MLP, and the input feature is also concatenated to the output feature by sMLP. That only uses normal information as network input.

Case 2 represents the improved structure that processes local feature encoding through different MLP layers containing normal feature extraction and merges the outputs to maintain the independence of the local normal and position features.

Case 3 considers the effect of the correlation between the position and normal features, where the features are first combined and processed through MLP encoding.

Case 4 processes each feature independently and combines them at the final stage of each layer. And the normal feature extraction block only performs normal encoding is named normal spatial encoding.

Cases 2, 3, and 4, with complexed modules, including normal feature encoding, perform better than Case 1 in most cases.

3. Experiments and Results

In this section, we will present experiments and analyze the results. We use the SemanticKITTI dataset (Behley *et al.*, 2019), which provides semantic annotations for all sequences.

3.1. Experimental settings

The SemanticKITTI dataset was used to train the proposed network model. The dataset was segmented into 19 classes and subdivided into three sequence groups for training (sequences from 0 to 10 except 8 with 19 130 scenes), validation (sequence 8 with 4071 scenes), and testing (sequences from 11 to 21 with 20 351 scenes). The initial learning rate was set to 0.01 with a decay rate of 0.95 for each epoch. The maximum number of epochs was set to 100, and the model with the best validation results was chosen. A value of 16 was used for k in the k-nearest search algorithm. The segmentation performance of the proposed method was evaluated using the mIoU (Everingham *et al.*, 2015) overall classes as defined in equation (9):

$$mIoU = \frac{1}{C}\sum_{c=1}^{C}\frac{TP_c}{TP_C + FP_C + FN_c} \tag{9}$$

where, C represents the number of classes, and TP_c, FP_C, and FN_c are the numbers of true positive, false positive, and false negative predictions for each class. A cross-entropy loss function with class-wise weights was used for training, as defined in equation (10):

$$L_{ce} = weight \cdot \left(-\ln\frac{\exp(z_j)}{\sum_{i=1}^{N}\exp(z_i)}\right) \tag{10}$$

where L_{ce} is the cross-entropy loss, and z_i and z_j are the ith and jth values of the output, respectively. The weights were determined to be inversely proportional to the inclusion ratio of the classes in the training dataset. The experiments used a workstation with NVIDIA RTX2080Ti (12GB) and TESLA T4 (16 GB) graphics cards.

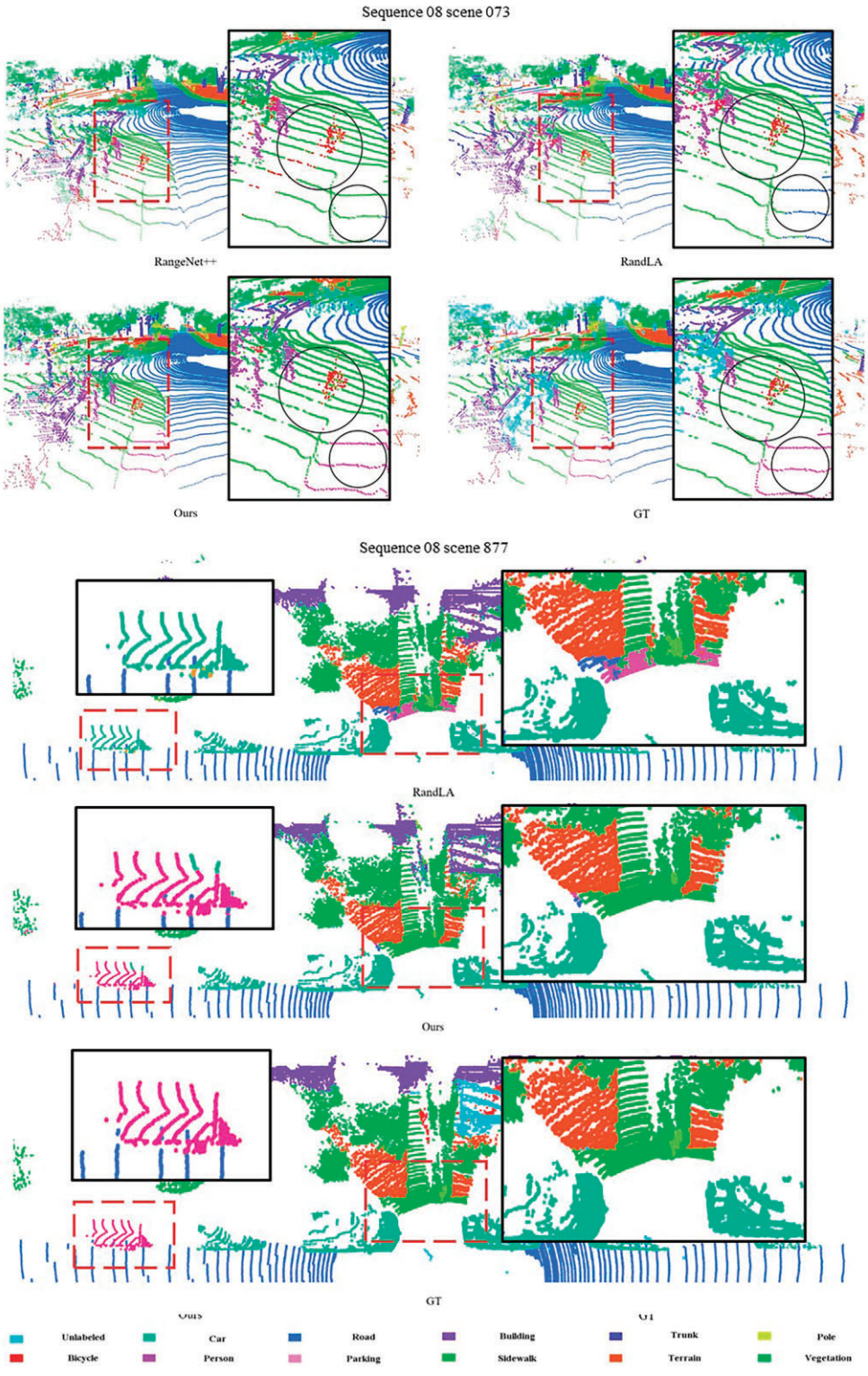


Figure 6: Comparison of the semantic segmentation results by RangeNet++, RandLA-Net, and the ground truth data on the validation set of SemanticKITTI. The proposed method produces more points with the same labels (color) as the ground truth than the others.

3.2. Robustness of normal estimation against noise

We have performed a few tests to demonstrate that the proposed normal estimation method is robust against noise. We created three point clouds by adding different noise levels sampled from normal distributions with standard deviations of 0.032, 0.065, and 0.12 m to a point cloud without noise. Normal vectors were estimated from each point cloud and compared. The proposed method yielded 4.18, 5.37, and 6.55 degrees for each noise level compared with the normal vectors without noise, whereas the traditional PCA showed 4.22, 6.22, and 8.76 degrees.

Table 2: Experiment on each normal estimation method.

Normal estimation method	mIoU (%)
Cross-product	59.1
PCA	60.2
Intensity-assisted PCA + orientation correction (ours)	61.5

Table 3: Experiment on normal feature encoder term.

Used normal term	mIoU (%)
$\boldsymbol{n}$	57.7
$\boldsymbol{n}, \boldsymbol{n}_i$	58.2
$\boldsymbol{n}, \boldsymbol{n}_i, \boldsymbol{n} - \boldsymbol{n}_i$	58.4
$\boldsymbol{n}, \boldsymbol{n}_i, \boldsymbol{n} - \boldsymbol{n}_i, \boldsymbol{n} \cdot \boldsymbol{n}_i$	61.5

Table 4: Experiment on K value of nearest neighbor search.

K for nearest neighbor search	mIoU (%)	Scan per second
$K = 4$	48.8	17.5
$K = 8$	57.3	6.8
$K = 16$	61.5	4.9
$K = 32$	57.1	2.8

Figure 5 shows the result of the estimated normal vectors using the traditional PCA and the proposed method. The normal vectors are encoded as color values and rendered at each point. Figure 5a and c show the normal vectors without noise estimated by the proposed and the PCA methods, respectively. Figure 5b and d show the normal vectors with a noise level of 0.065, estimated by the proposed and PCA methods, respectively. The figure indicates that the proposed method estimates consistent normal vectors compared with the PCA method, implying that it is not noise-sensitive.

3.3. SemanticKITTI benchmarks

The segmentation performance of the proposed method was evaluated using the SemanticKITTI dataset. Table 1 summarizes the performance of the proposed methods with four NLFA structures and recent approaches, including 2D view-based (projection-based), 3D voxel-based, and point-based approaches. The results show that incorporating normal information helps improve segmentation performance, verified by Case 1. The normal vector is a surface intrinsic property that indicates the orientation of the surface of an object. Therefore, the distribution of normal vectors at the points on the object can capture the structure of its geometric shape. The proposed semantic network model that utilizes both position and normal vector information can outperform similar models that use only position information. because more information about the object's shape is incorporated during training and prediction. Specifically, using normal vectors can enhance the performance of the local feature extraction module in the proposed network, resulting in improved segmentation performance.

The Case 3 model achieved a mIoU of 57.9%. Notably, the proposed method demonstrated high performance for relatively small objects: 95.2% for cars, 47.4% for bicycles, 41.0% for motorcycles, 57.4% for bicycles, and 53.2% for pedestrians. This suggests that the normal vector-based feature aggregation module has enhanced recognition performance for small targets by leveraging the orientation information of normal vectors. A small number of points define a small object due to its surface size. Therefore, a limited number of points cannot sufficiently represent it. The network can use more object-shape information by adding normal vectors, resulting in improved segmentation performance.

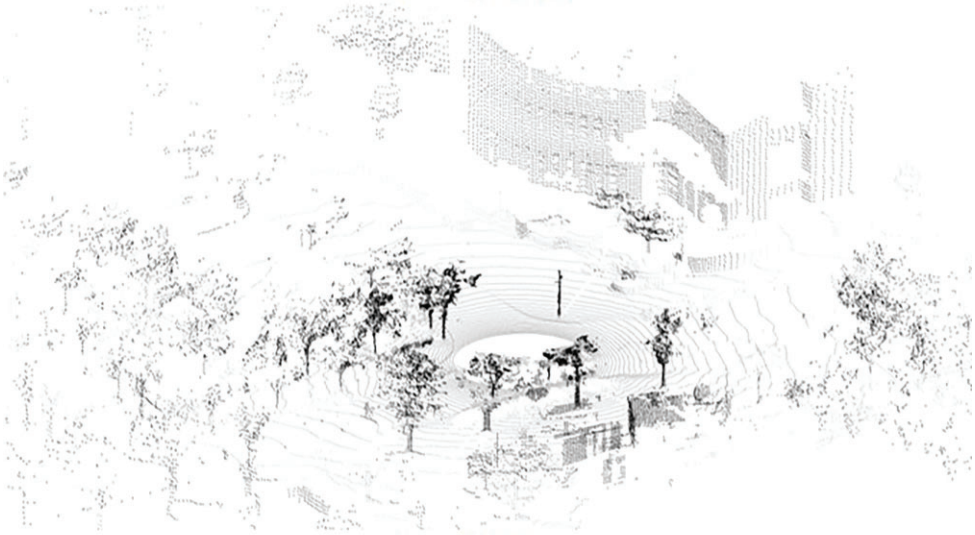
The proposed method took approximately 3 s to preprocess normal generation for each scene on a CPU and 202 ms for segmentation for about 50 000 input points with RTX 2080 GPU. SqueezeSeg 1 and 2 took 23 and 31 ms. RangeNet++ and PolarNet took 78 and 67 ms, while RandLA took 124 ms. The proposed method takes longer than the other methods due to processing normal information.

Figure 6 compares the segmentation results by RangeNet++, RandLA-Net, the proposed method, and the ground truth data using Sequence 8. Compared with the ground truth, the proposed method extracted more points with the same labels (color) and detected more objects than the other methods.

As observed in the experimental results from scene 073, the image-based method, RangeNet++, shows dimensional errors during projection. In contrast, our proposed approach yields highly accurate segmentation results, surpassing the

Figure 7: Autonomous vehicle equipped with Ouster-OS1 LiDAR.

(a) Camera image



(b) LiDAR scan

Figure 8: Driving test on Ouster-OS1 LiDAR sensor.

performance of RandLA, which needs help to differentiate between road and parking areas effectively. Furthermore, the evaluation on scene 877 demonstrates the robustness of our method in accurately distinguishing road structures and type of vehicles, as evidenced by the achieved segmentation results being in closest proximity to the ground truth. These compelling results affirm the superiority of our approach in addressing the segmentation challenges posed by complex urban scenes and underscore its potential for practical applications in various real-world scenarios.

3.4. Ablation study

In this section, we aim to investigate the impact of each term within the proposed framework through an ablation study. All the experiments described in this section were trained and tested on the Sequence 8 validation set of SemanticKITTI.

Table 2 shows that using accurate normal vectors helps to improve segmentation performance. The experiments show that the cross-product method explained in Section 2.1 achieved a mIoU of 59.1%, and traditional PCA achieved a mIoU of 60.2%. In contrast, the proposed method that produces refined normal vectors achieved a mIoU of 61.5%. In contrast to the non-uniform directionality observed in the two previous methods, our novel normal estimation approach optimized for LiDAR has yielded uniformly oriented normal information. The result shows this uniformity in orientation has significantly contributed to enhancing network performance. This, in turn, signifies that our proposed method enables the acquisition of highly reliable local features.

Table 3 summarizes how each term of $(\boldsymbol{n},\ \boldsymbol{n}_i, \boldsymbol{n} - \boldsymbol{n}_i, \boldsymbol{n} \cdot \boldsymbol{n}_i)$ influenced the overall segmentation performance. When $\boldsymbol{n}$ was used, the network achieved an mIoU of 57.7%. The mIoU value grew as more terms were included, resulting in a mIoU of 61.5% when the four terms were used. This indicates that each term in the tuple provided more information on the local geometric structure, helping the network extract each object's intrinsic features.

A new parameter introduced in the proposed method is k: the number of neighboring points included in the normal estimation step. The value of k directly affects the performance of normal estimation. A small k value means a small number of points, which may not represent the underlying geometry with sufficient accuracy, leading to poor estimation. On the other hand, as k

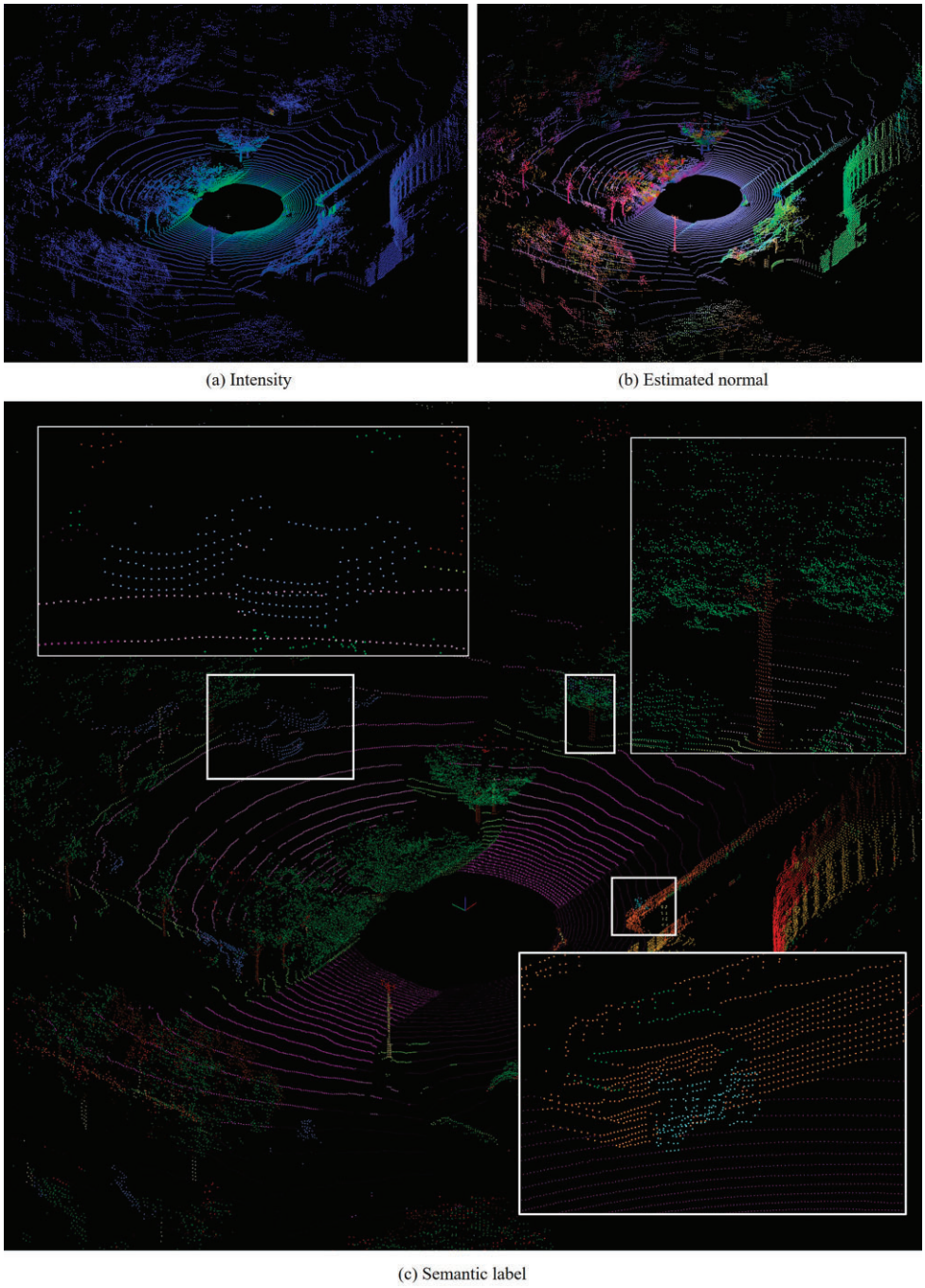


Figure 9: Visualization results of each process (a) show LiDAR intensity map, (b) estimated normal direction by our intensity guidance method, and (c) semantic segmentation result.

increases, more points are considered, which can represent the underlying geometric shape sufficiently. Therefore, more accurate normal vectors can be estimated. However, there is a limit to increasing k. A normal vector is a local feature that captures the geometric shape around a point. Too many points for large k would cover a large area, which may negatively affect normal estimation because they may include areas where the normal vectors deviate significantly from the true one. The optimal k-value selection depends on the point cloud's distribution pattern, which should be empirically determined through experimentation. Table 4 shows the computational times and mIoU values concerning k. The computational time increased proportionally to k. However, the segmentation performance improved as k increased up to 16 and then dropped when $k = 32$. These findings indicate an optimal k that influences the accuracy of normal estimation. In this work, we chose $k = 16$ through this experiment.

3.5. Road test

This section illustrates the results of applying the trained network for segmentation to LiDAR point cloud data obtained from real-world autonomous vehicle operations. The segmentation outcomes demonstrate the proposed framework's efficacy in an authentic driving environment. As shown in Fig. 7, experiments employed a 64-channel Ouster-OS1 LiDAR, attached to the vehicle's roof, to capture the surrounding environment in a 360-degree with 65 536 points per scan. The driving experiments took place

at GIST (Gwangju Institute of Science and Technology). Figure 8a is camera image captured by mobile device, and Fig. 8b is corresponding LiDAR scan by Ouster.

Figure 9 presents visualizations of each step of the proposed framework. Figure 9a shows the normalized intensity of the point cloud acquired from the Ouster LiDAR, and Fig. 9b shows the estimation result following the application of our proposed technique to Fig. 9a. The results from both Fig. 9a and b were employed as inputs to the network, calculating semantic labels shown in Fig. 9c. The benchmark Case 3 network model is used for estimation. Despite the disparity between the sensor LiDAR model for training data (Velodyne, KITTI) and for acquiring driving data (Ouster), we get high-quality semantic labels. Notably, the system effectively differentiated objects of varying scales, such as cars and fences, trunks and pedestrians, and bicycles.

4. Conclusions

This paper presents a novel semantic segmentation network model that processes 3D LiDAR scans to enhance segmentation performance. It builds upon RandLA-Net by introducing an efficient method for embedding normal vectors in the network structure to improve local feature extraction performance. By combining a novel intensity-assisted normal estimation technique that enhances the accuracy of normal estimation, the proposed network outperforms existing methods by achieving a 4% higher mIoU score than the original RandLA-Net on the SemanticKITTI benchmark tests. In particular, it demonstrated superior performance to existing methods for small and dynamic objects, such as a 21% improvement for bicycles, 4% for pedestrians, and 15.2% for motorcycles compared with RandLA-Net.

This work primarily focuses on improving the performance of semantic segmentation of point clouds generated by LiDAR sensors using normal vectors. A new network structure that takes normal vectors as input was proposed and integrated into the RandLA network. The test results demonstrate that incorporating normal features has enhanced the segmentation performance of the RandLA network, which suggests its applicability to the latest methods, and an improvement in semantic segmentation performance can be anticipated.

The proposed method has two limitations. Firstly, normal estimation, an essential step in the proposed method, inevitably increases the overall computational time. In particular, the time for normal estimation grows proportionally to the size of input points, which prevents its use in real-time applications. Secondly, the proposed method cannot utilize LiDAR sensors that do not provide intensity. Although the proposed method works without intensity, the lack of intensity compromises the accuracy of normal vector estimation and subsequently negatively affects the segmentation performance of the network. Overcoming these limitations is recommended for future work.

Acknowledgments

This work was supported by Korea Institute of Industrial Technology as “Development of Core Technologies for a Smart Mobility (KITECH JA-23–0011)”. ChatGPT3.5 was used to check the grammar of a part of the manuscript.

Conflict of interest statement

None declared

References

Behley, J., Garbade, M., Milioto, A., Quenzel., J., Behnke, S., Stachniss, C., & Gall, J. (2019). SemanticKITTI: A Dataset for Semantic Scene Understanding of LiDAR Sequences. In *Proceedings of the IEEE/CVF international conference on computer vision*. (pp. 9297–9307). https://doi.org/10.48550/arXiv.1904.01416

Çiçek, Ö., Abdulkadir, A., Lienkamp, S. S., Brox, T., & Ronneberger, O. (2016). 3D U-Net: Learning dense volumetric segmentation from sparse annotation. In *Proceedings of the 19th International Conference on Medical Image Computing and Computer-Assisted Intervention–MICCAI 2016 Part II*(pp. 424–432). Springer International Publishing. https://doi.org/10.1007/978-3-319-46723-8_49.

Eom, H., & Lee, S. H. (2022). Mode confusion of human–machine interfaces for automated vehicles. *Journal of Computational Design and Engineering*, **9**(5), 1995–2009. https://doi.org/10.1093/jcde/qwac088.

Everingham, M., Eslami, S. A. M., Gool Van, L., Williams, C. K. I., Winn, J., & Zisserman, A. (2015). The PASCAL Visual Object Classes Challenge: A Retrospective. *International Journal of Computer Vision*, 98–136. https://doi.org/10.1007/s11263-014-0733-5

Fan, S., Dong, Q., Zhu, F., Lv, Y., Ye, P., & Wang, F. Y. (2021). SCF-Net: Learning spatial contextual features for large-scale point cloud segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*(pp. 14504–14513). IEEE. https://doi.org/10.1109/CVPR46437.2021.01427.

Hilbig, A., Vogt, L., Holtzhausen, S., & Paetzold, K. (2023). Enhancing three-dimensional convolutional neural network-based geometric feature recognition for adaptive additive manufacturing: A signed distance field data approach. *Journal of Computational Design and Engineering*, **10**(3), 992–1009. https://doi.org/10.1093/jcde/qwad027.

Hoppe, H., DeRose, T., Duchamp, T., McDonald, J., & Stuetzle, W. (1992). Surface reconstruction from unorganized points. In *Proceedings of the 19th Annual Conference on Computer Graphics and Interactive Techniques*(pp. 71–78). https://doi.org/10.1145/133994.134011.

Hu, Q., Yang, B., Xie, L., Rosa, S., Guo, Y., Wang, Z., & Markham, A. (2020). RandLA-Net: Efficient semantic segmentation of large-scale point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*(pp. 11108–11117). IEEE. https://doi.org/10.48550/arXiv.1911.11236.

Huber, P. J. (2004). *Robust statistics*, **(Vol. 523)**. John Wiley & Sons.

Luo, N., Wang, Y., Gao, Y., Tian, Y., Wang, Q., & Jing, C. (2021). kNN-based feature learning network for semantic segmentation of point cloud data. *Pattern Recognition Letters*, **152**, 365–371. https://doi.org/10.1016/j.patrec.2021.10.023.

Milioto, A., Vizzo, I., Behley, J., & Stachniss, C. (2019). RangeNet++: Fast and accurate LiDAR semantic segmentation. In *Proceedings of the 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*(pp. 4213–4220). IEEE. https://doi.org/10.1109/IROS40897.2019.8967762.

Noh, S., & An, K. (2022). Reliable, robust, and comprehensive risk assessment framework for urban autonomous driving. *Journal of Computational Design and Engineering*, **9**(5), 1680–1698. https://doi.org/10.1093/jcde/qwac079.

Park, Y. S., Jang, H., & Kim, A., (2020). I-LOAM: Intensity Enhanced LiDAR Odometry and Mapping. *17th International Conference on Ubiquitous Robots (UR)*(pp. 455–458). IEEE. https://doi.org/10.1109/UR49135.2020.9144987

Qi, C. R., Su, H., Mo, K., & Guibas, L. J. (2017a). PointNet: Deep learning on point sets for 3D classification and segmentation. In *Proceed-*

ings of the IEEE Conference on Computer Vision and Pattern Recognition(pp. 652–660). IEEE. https://doi.org/10.48550/arXiv.1612.00593.

Qi, C. R., Yi, L., Su, H., & Guibas, L. J. (2017b). PointNet++: Deep hierarchical feature learning on point sets in a metric space. In *Advances in neural information processing systems*(Vol. **30**, pp. 5099–5108). MIT. https://doi.org/10.48550/arXiv.1706.02413.

Redmon, J., & Farhadi, A. (2018). YOLOv3: An incremental improvement. preprint (arXiv:1804.02767). https://doi.org/10.48550/arXiv.1804.02767.

Wu, B., Wan, A., Yue, X., & Keutzer, K. (2018). SqueezeSeg: Convolutional neural nets with recurrent CRF for real-time road-object segmentation from 3D LiDAR point cloud. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*(pp. 1887–1893). IEEE. https://doi.org/10.1109/ICRA.2018.8462926.

Wu, B., Zhou, X., Zhao, S., Yue, X., & Keutzer, K. (2019). SqueezeSegV2: Improved model structure and unsupervised domain adaptation for road-object segmentation from a LiDAR point cloud. In *Proceedings of the 2019 International Conference on Robotics and Automation (ICRA)*(pp. 4376–4382). IEEE. https://doi.org/10.48550/arXiv.1809.08495.

Zhang, F., Fang, J., Wah, B., & Torr, P. (2020). Deep FusionNet for point cloud semantic segmentation. In *Proceedings of the 16th European Conference on Computer Vision–ECCV 2020, Part XXIV*(pp. 644–663). Springer International Publishing. https://doi.org/10.1007/978-3-030-58586-0_38.

Zhang, Y., Zhou, Z., David, P., Yue, X., Xi, Z., Gong, B., & Foroosh, H. (2020). PolarNet: An improved grid representation for online LiDAR point clouds semantic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*(pp. 9601–9610). IEEE. https://doi.org/10.48550/arXiv.2003.14032.

Zhou, Y., & Tuzel, O. (2018). VoxelNet: End-to-end learning for point cloud based 3D object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*(pp. 4490–4499). IEEE. https://doi.org/10.48550/arXiv.1711.06396.

Received: September 11, 2023. **Revised:** November 7, 2023. **Accepted:** November 7, 2023