Sensors and Materials, Vol. 36, No. 9 (2024) 3997–4015
MYU Tokyo

S & M 3780

Seabed Segmentation of Airborne Bathymetric Light Detection and Ranging Point Cloud Using Window-based Attention and Orthogonal Regularized PointNet

Ahram Song[1] and Jaebin Lee[2*]

[1]Department of Location-based Information System, Kyungpook National University, Sangju, Gyeongsangbuk-Do 37224, South Korea
[2]Department of Architectural, Civil and Environmental Engineering, Mokpo National University, Muan, Jeonnam 58554, South Korea

(Received August 22, 2024; accepted September 17, 2024)

Keywords: airborne bathymetric LiDAR, seabed segmentation, PointNet, window-based attention, orthogonal regularization

Seabed segmentation from airborne bathymetric Light Detection and Ranging (LiDAR) point cloud data presents unique challenges, primarily due to variations in the *z*-axis resulting from differences in water depth and seabed topography. To address these complexities, we introduced an improved version of PointNet specifically designed for seabed segmentation using Airborne Bathymetric LiDAR (ABL) point cloud data. The proposed method integrates a window-based attention mechanism to capture spatial relationships in both horizontal and vertical dimensions while incorporating orthogonal regularization to preserve geometric integrity. The model's performance was assessed using various normalization methods and window sizes, demonstrating its effectiveness in accurately identifying seabed regions. Experimental results indicate that while the proposed network generally improves segmentation accuracy, its performance is sensitive to the choice of normalization and window parameters. This study represents a meaningful advancement in applying deep learning techniques to bathymetric LiDAR data, offering a robust framework for seabed segmentation.

1. Introduction

Modeling the topography of the coastal seabed has become critical with increasing human activities and the exploitation of coastal areas. Over the past few decades, airborne bathymetric Light Detection and Ranging (LiDAR), as an alternative to echo sounding, has demonstrated its efficiency and cost-effectiveness in producing accurate elevation data in coastal zones.(1) It is particularly economical and accurate for producing periodic elevation data over large coastal areas. Its applications include nautical charting, shoreline mapping, regional sediment management, and benthic habitat mapping.(2–4)

*Corresponding author: e-mail: lee2009@mokpo.ac.kr
https://doi.org/10.18494/SAM5331

ISSN 0914-4935

An airborne bathymetric LiDAR (ABL) system uses green lasers to penetrate water, with pulses traveling through air, hitting the water surface, and reaching the seabed. These pulses reflect off different surfaces and the system measures the return time to calculate the distance to the seabed. Using the speed of light in water, sensor geometry information, and the precise aircraft location from GPS, the system generates accurate seabed elevation data.[5] In traditional bathymetric LiDAR processing workflows, seafloor return segmentation typically relies on return waveforms, which are time series of return signal amplitudes. These waveforms often display peaks that correspond to the sea surface and seafloor, as well as volume backscatter, which represent returns from within the water column, and these peaks can be obtained through waveform decomposition.[6] By analyzing the relative positions and amplitudes of the peaks on each waveform, along with the characteristics of the decomposed waveforms, the returns corresponding to the peaks can be classified as seafloor, sea surface, and noise. For example, based on the American Society for Photogrammetry and Remote Sensing LAS file format[7] and Topo-Bathy Domain Profile,[8] Point Class 40 corresponds to bathymetric bottom (i.e., seafloor points). For this purpose, commercial bathymetric LiDAR systems accommodate the segmentation and geolocation of seafloor returns using proprietary software, such as Leica LiDAR Survey Studio, Riegl RiHYDRO, and Optech HydroFusion, but the algorithms are not publicly disclosed.[9] Scientists interested in modeling and studying the seafloor can generate bathymetric DEMs, meshes, contours, and other products using only the Point Class 40 points.

However, various environmental factors and noise can hinder the classification of local peaks in waveform data, making it difficult to identify the peak representing the seafloor. This problem is exacerbated in both deep and shallow areas: in deep areas, the waveform signal significantly attenuates, resulting in poor signal-to-noise ratios, whereas in shallow areas, a single peak may represent both the sea surface and seafloor returns.[10] Additionally, substances such as chlorophyll and suspended solids, which absorb certain wavelengths, can also distort the waveform shape and introduce errors in seafloor point classification.[11] This variation makes waveform classification difficult, resulting in point cloud data being provided in an unclassified state. Another issue is that return waveforms, intensities, trajectories, and other ancillary data are typically not provided to end users. Many previous ABL datasets are only available in point cloud form without waveform information. Recently, access to waveform data has become possible upon user request, but this is not common. Consequently, scientists and other end users may still lack the specialized knowledge in bathymetric LiDAR processing necessary to develop and implement their own seafloor segmentation techniques.

Several studies have been conducted to separate the seafloor in ABL using only the geometric information from point cloud data. Jung *et al.* proposed a novel clustering algorithm for seafloor segmentation using the geometric properties of point cloud data (i.e., x, y, and z coordinates) as the sole input.[12] Unlike conventional point cloud classification methods, their approach did not rely on any predefined statistical distribution of the points. Their study assumed that ABL pulses predominantly capture returns from both the sea surface and the seafloor. On the basis of this assumption, they identified the vertical separation between point clusters representing the seafloor and the water surface. This approach enabled the development of a computationally efficient, user-friendly seafloor extraction method that can be applied without the specialized

knowledge of bathymetric LiDAR processing. However, in areas where seafloor returns were limited or absent, there was a risk of misclassification. Kim *et al.* introduced a pseudo-waveform decomposition technique to classify the seafloor from ABL point clouds.[9] They generated pseudo-waveforms by analyzing the spatial distribution of point clouds and applied a self-developed waveform decomposition method to separate the seafloor. In their research, they specifically focused on classifying seafloor points in shallow areas (less than 2 m deep) and assessed the effectiveness of their method using datasets collected from two different ABL systems, the Riegl VQ-880-G and Seahawk.

Although previous studies have reported successful results, users still need to determine thresholds based on the topographical characteristics of the target area. Existing methods[13–15] that use machine learning algorithms have demonstrated good seafloor segmentation results, but they do not meet the needs outlined above, owing to the requirement for a large number of input variables (in some cases, including full waveforms).

In contrast, deep learning approaches have made great strides in various classification and segmentation tasks across different domains.[16,17] These methods have the advantage of learning complex patterns and features directly from the data, thus reducing the need for extensive preprocessing and manual parameter setting.[18] Recently, several studies have explored the application of deep learning techniques to bathymetric LiDAR data for seabed segmentation. For example, Liu *et al.* applied a convolutional neural network (CNN) to classify seabed types from LiDAR data and achieved a higher accuracy than traditional methods.[19] Zhang *et al.* utilized a deep-learning approach to segment the seabed, and the result demonstrated high precision and recall rates.[20] These studies indicate the usefulness of deep learning to enhance the accuracy and efficiency of seabed classification in bathymetric LiDAR data. Despite these advancements, the unique challenges posed by bathymetric LiDAR data require specialized approaches. Bathymetric LiDAR data typically consist of sparse, unstructured point clouds with varying densities and noise levels, making traditional 2D image-based CNNs less effective. Moreover, the critical importance of the *z*-axis in distinguishing between different types of return (e.g., water surface, water column, and seabed) necessitates a method that can effectively leverage 3D spatial information.

PointNet, a deep learning architecture for point cloud processing, has demonstrated remarkable performance in 3D classification and segmentation tasks.[21] The architecture, which is invariant to the order of input points, is crucial for handling unstructured point cloud data. This ensures that the performance of the model is not affected by the order in which points are presented. Furthermore, PointNet employs a multilayer perceptron network to learn spatial features from point clouds, capturing both local and global structures essential for distinguishing between different return types in bathymetric LiDAR data. Its robustness to noise and variations in point density, combined with its scalable architecture, makes it a strong candidate for efficiently processing extensive bathymetric LiDAR datasets. However, the standard PointNet architecture does not explicitly account for the unique characteristics of bathymetric LiDAR data, such as the importance of the *z*-axis for distinguishing between water surface and seabed points.

To address these challenges, in this study, we propose an enhanced version of PointNet, which incorporates window-based attention mechanisms and orthogonal regularization, tailored to the segmentation of bathymetric LiDAR point clouds. The modifications were designed to emphasize the *z*-axis, which is critical for accurate seabed classification, and to leverage attention mechanisms to accurately capture local and global point relationships. The key contributions of this study are as follows:

1. Enhanced PointNet Architecture: Incorporating window-based attention mechanisms that allow the network to focus on relevant local features within the point cloud while also capturing the global context. This is particularly beneficial for bathymetric LiDAR data, where local depth variations are critical for accurate classification.
2. *Z*-axis Emphasis: Implementing a normalization technique that assigns greater weight to the *z*-axis, enhancing the ability of the network to distinguish between different types of return (e.g., water surface and seabed).
3. Orthogonal Regularization: Integrating orthogonal regularization to maintain the orthogonality of transformation matrices within the network, thus improving the stability and performance of the model.

By integrating these advanced techniques into the PointNet architecture, we aim to provide a more effective and efficient approach for seabed segmentation using ABL data. The proposed method not only improves the classification accuracy but also simplifies the workflow, making it more accessible to researchers and practitioners in the field of coastal and marine studies.

2. Methods

2.1 Preprocessing

To prepare the bathymetric LiDAR point cloud data for input into the neural network, we employed a two-step preprocessing approach: normalization and *z*-axis emphasis. The primary goal is to compare the effects of standard normalization against *z*-axis-emphasized normalization with different weights. Normalization is a fundamental preprocessing step designed to center and scale the point cloud data to a standardized range. This step helps to stabilize training and improve the performance of the neural network. During normalization, a fixed number of points are randomly sampled from each point cloud to maintain consistency in the input size and reduce computational complexity, thus ensuring that the model handles a uniform amount of data regardless of the original point cloud size. For standard normalization, the sampled point cloud P is centered by subtracting the mean μ of the points [Eq. (1)].

$$P_{norm} = P - \mu \tag{1}$$

Here, $\mu = \frac{1}{N}\sum_{i=1}^{N} P_i$ and N is the number of sampled points. Each point in the point cloud is scaled by the maximum norm of the points, bringing the point cloud into a standard unit sphere and facilitating neural network learning [Eq. (2)].

$$P_{norm} = \frac{P_{sampled} - \text{mean}\left(P_{sampled}\right)}{\max\left(\left\|P_{sampled} - \text{mean}\left(P_{sampled}\right)\right\|_2\right)} \tag{2}$$

This standard normalization ensures that the point cloud is centered and lies within a unit sphere, thus facilitating learning.

Given the importance of the vertical dimension (z-axis) in distinguishing between the seabed and surface points, a z-axis emphasis was applied. This approach enhances the significance of the z-coordinate, aiding in the accurate classification of different point types. The z-axis-emphasized normalization involves the same initial steps of random sampling and centering. However, the z-coordinates of the points were multiplied by a weight factor w_z to emphasize the vertical dimension [Eq. (3)]. Equation (3) represents the formula used to apply this emphasis, where w_z is the weight applied to the z-coordinates to scale their effect relative to the x- and y-coordinates. The weight for the z-axis is typically set to 1 as it is generally not given special consideration. However, to progressively assess the importance of the z-values, we applied multipliers of 2 and 3. The points were then scaled by the maximum norm, similar to standard normalization. This process ensures that the z-axis is emphasized appropriately, enhancing the model's ability to differentiate between seabed and other points.

$$P_{norm}[:,2] = w_z \times P_{norm}[:,2] \tag{3}$$

To evaluate the effectiveness of the z-axis-emphasized normalization, we conducted a series of experiments comparing the standard normalization performance and z-axis-emphasized normalization with different weights. The experiments were designed to assess the impact of z-axis weighting on the classification accuracy of seabed and other points, including water column and surface points. Standard normalization was performed without any emphasis on the z-axis, whereas z-axis-emphasized normalization was performed for different weights (w_z = 2 and w_z = 3). Each of these preprocessing methods was applied to the same dataset, and the resulting point clouds were used to train the neural network. The performance of the model was evaluated on the basis of validation set accuracy and loss, as well as training set accuracy and loss. By comparing the results of these experiments, we aimed to determine the optimal preprocessing method for enhancing the classification accuracy of bathymetric LiDAR data using deep learning techniques.

2.2 Enhanced PointNet architecture

In this study, we present an enhanced version of PointNet that is tailored to the unique characteristics of bathymetric LiDAR data. The primary modifications include the introduction of a window-based attention mechanism and orthogonal regularization aimed at improving the network's ability to effectively process and segment the point clouds.

The Transform Block (Ortho. Reg.) aligns input features while preserving geometric properties (Fig. 1). Conv Blocks apply 1D convolutions to extract local spatial features. The

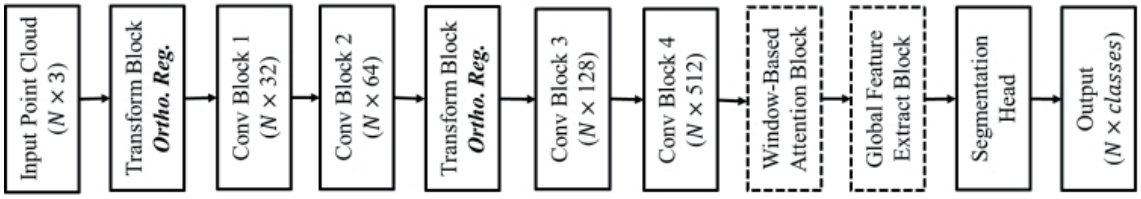


Fig. 1. Flowchart of enhanced PointNet architecture.

Window-based Attention Block highlights critical *z*-axis features for seabed differentiation. The Global Feature Extract Block captures global context, and the Segmentation Head assigns each point to its class.

The modified PointNet architecture began with a series of transformation and convolutional blocks designed to extract robust features from the raw point cloud data. Initially, an input transformation network was applied to the input points to align them into a canonical space. This transformation network incorporates orthogonal regularization, which ensures that the learned transformations preserve the geometric properties of the input data, thereby enhancing the stability and interpretability of the model.

Thereafter, several convolutional layers progressively increased the feature dimensions, capturing both local and global geometrical structures. Orthogonal regularization was again applied within the feature transformation block, further reinforcing the integrity of the learned features and improving the overall robustness of the network.

Figure 2 demonstrates the application of grid-based windows within a 3D point cloud, which is a core aspect of the window-based attention mechanism in the enhanced PointNet architecture. These windows, visualized as red cuboids, are strategically placed across the point cloud to capture local spatial features, particularly in the *z*-axis direction. By dividing the point cloud into smaller regions, the attention mechanism effectively highlights and processes critical features essential for accurate seabed differentiation. This approach enables the model to maintain a high level of spatial awareness, ensuring that both local variations and the global context are considered during segmentation. Grid-based windows serve as a visual representation of spatial regions where attention is focused, especially for capturing the significant variations in the *z*-axis, which are key to differentiating the seabed from other classes, including the water surface and column. This targeted attention allows the model to effectively learn the complex patterns inherent in 3D point cloud data, thereby enhancing overall segmentation performance.

Building on this concept, the incorporation of the window-based attention mechanism stands out as a key innovation in the modified architecture. Designed specifically to handle the distinct characteristics of bathymetric LiDAR data, this mechanism ensures that significant *z*-axis variations are effectively leveraged to distinguish between the seabed and other classes. The window-based attention mechanism divides the input point cloud into patches and applies multi-head attention within each patch to capture local context. Thereafter, it combines the outputs to form a comprehensive representation. This ensures that the model effectively utilizes both horizontal and vertical spatial information, as mathematically formulated in Eq. (4).

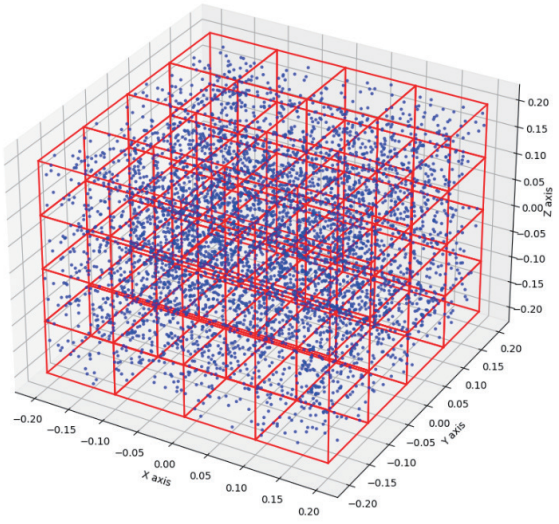


Fig. 2. (Color online) Visualization of 3D point cloud with focused grid-based windows.

$$Attention(Q,K,V) = softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{4}$$

Here, Q is the query matrix, K is the key matrix, V is the value matrix, and d_k is the dimension of the keys. The window-based attention mechanism was implemented with specific window sizes for the x, y, and z dimensions. To emphasize the z-axis, which is crucial for distinguishing between seabed and other returns, different window sizes were applied, with the z-axis window size being elongated compared with the x and y dimensions. Specifically, the window sizes used were (32, 32, 32), (32, 32, 64), and (32, 32, 128). This design allows the network to better capture the vertical variations in point cloud data.

Orthogonal regularization was also integrated into the network to maintain the orthogonality of the transformation matrices, which is crucial for preserving the geometric properties of the point clouds during feature extraction. This regularization was applied to the weights of the transformation networks, thus ensuring that the transformations remain stable and interpretable. The architecture concluded with a global feature extraction step using max pooling, followed by a segmentation head that outputs class probabilities for each point. The global features are tiled and concatenated with local features to ensure that both the global context and local details are considered in the final segmentation.

2.3 Training setup and configuration

To effectively train the enhanced PointNet model for seabed segmentation, we implemented a comprehensive training pipeline designed to handle the complexities of bathymetric LiDAR data. The training setup involves several key components, including data augmentation, dataset preparation, learning rate scheduling, and checkpointing.

To ensure that the model generalizes well to unseen data, data augmentation techniques were applied during training. This involved adding random noise to the point clouds to make the model robust to variations and noise inherent in real-world bathymetric LiDAR data. The point clouds and their corresponding labels were first loaded and cast to the appropriate data types. Each point cloud was sampled to a fixed number of points (N) to maintain consistency in input size. The augmentation function introduced random noise to the point clouds, defined as Eq. (5).

$$point\ cloud\ batch[i, j] = point\ cloud\ batch[i, j] + noise[i, j] \tag{5}$$

Here, *point cloud batch*[i, j] refers to the j-th coordinate (e.g., x, y, or z) of the i-th point in the point cloud batch. *noise*[i, j] is the random noise added to the j-th coordinate of the i-th point. Moreover, *noise*[i, j] was sampled from a uniform distribution $U(-0.005, 0.005)$. This operation was applied to each point i and each coordinate j in the point cloud, effectively perturbing the original coordinates by a small, random amount of noise. The goal is to make the model more robust to small variations in the data, which simulates potential noise and inaccuracies in real-world LiDAR measurement.

The dataset was prepared by splitting the data into training and validation sets with a validation split of 40%. The data was then shuffled, batched, and mapped with the augmentation function for training data or directly loaded for validation data. A learning rate scheduling strategy was employed to ensure efficient and stable training. The learning rate decreased exponentially over time to allow the model to converge smoothly. The learning rate started at 10^{-3} and decayed by 4% every 100000 steps. The enhanced PointNet model was compiled with the Adam optimizer and categorical cross-entropy loss. Early stopping was implemented to prevent overfitting, with a patience of 70 epochs, and the best model weights were restored on the basis of validation loss. Training was executed over 200 epochs. The performance of the model was evaluated on both the training and validation datasets, with the history of training metrics recorded for analysis.

2.4 Accuracy assessment

To evaluate the performance of the enhanced PointNet architecture, we employed both training and validation accuracy metrics. These metrics provide insights into how well the model has learned to classify and segment the point clouds, as well as its ability to generalize to unseen data.

The primary metric used for accuracy assessment was categorical accuracy, which measures the percentage of correctly classified points out of the total number of points. This metric is particularly suitable for multiclass classification problems, such as seabed segmentation and other classes in bathymetric LiDAR data.

The loss function used for training the model was categorical cross-entropy. This loss function is defined in Eq. (6).

$$Loss = -\sum_{i=1}^{N}\sum_{c=1}^{C} y_{ic} \log\left(\hat{y}_{ic}\right) \tag{6}$$

Here, N is the number of samples, C is the number of classes, y_{ic} is the binary indicator (0 or 1) if class label c is the correct classification for sample i, and $\hat{y}_{ic}$ is the predicted probability that sample i belongs to class c. Categorical accuracy was calculated from Eq. (7).

$$Accuracy = \frac{1}{N}\sum_{i=1}^{N} 1\left(\text{argmax}_c\, \hat{y}_{ic} = \text{argmax}_c\, y_{ic}\right) \tag{7}$$

Here, 1 is the indicator function that returns 1 if the predicted class $\text{argmax}_c\, \hat{y}_{ic}$ matches the true class $\text{argmax}_c\, y_{ic}$ and 0 otherwise. During training, the performance of the model was continuously monitored using the validation dataset to ensure that it did not overfit the training data. Early stopping was decided on the basis of the validation loss, with patience set to 70 epochs. This means that if the validation loss did not improve for 70 consecutive epochs, training was halted to prevent overfitting. The final performance of the model was evaluated on the basis of the best validation accuracy achieved during training. The training and validation accuracy metrics were reported to provide a comprehensive view of the effectiveness of the model in both learning and generalization.

3. Experimental Results and Discussion

3.1 Test site and dataset

The proposed approach was tested on ABL datasets acquired offshore of Marco Island and Virginia Key, Florida, using the NOAA National Geodetic Survey (NGS) Remote Sensing Division's Riegl VQ-880-G system (Table 1). Marco Island is a barrier island in the Gulf of Mexico in southwest Florida, with offshore areas containing primarily sandy substrates and some seagrass beds.[22] It was selected to validate the ability of the proposed approach to classify ABL point cloud data in shallow water and a near-shore area that is generally known to be challenging for seabed point cloud classification from ABL data. Figure 3 and Table 2 present the test site and details of the ALB dataset relevant to the experiment, respectively. The reported accuracy of the ALB data was ±0.15 m (1σ) vertical and ±1.0 m (1σ) horizontal.[23] In the table, the Z range of the test data is much larger than the visually measured water depth range due to noise and outliers. Before the experiments, the outliers (e.g., returns from birds, atmospheric particles, and system noise) in the datasets were reduced using a statistical outlier removal filter in the point cloud processing software CloudCompare v2.11.3. The algorithm computes the average distance and standard deviation of each point to its neighbors and then removes the outlier points using the sigma rules.[24] In this study, we empirically set the number of neighbors to 10 and the standard deviation multiplier threshold to 3 (i.e., 3 sigma), considering the point density and noise level of the datasets. To construct reference data for accuracy assessment, the

Table 1
Specifications for Riegl VQ-880G system.

Parameter	Specification
Laser wavelength	532 nm (green), 1064 nm (NIR)
Laser pulse width	1.6 ns
Laser beam divergence	0.7–2.0 mrad (green), 0.2 mrad (NIR)
Field of view (FoV)	40°
Pulse repetition rate	550 kHz
Operating flight altitude	600 m
Minimum range (1.5 Secchi depth)	10 m
Main dimensions ($L \times W \times H$)	444 × 444 × 695 mm^3
Weight	65 kg

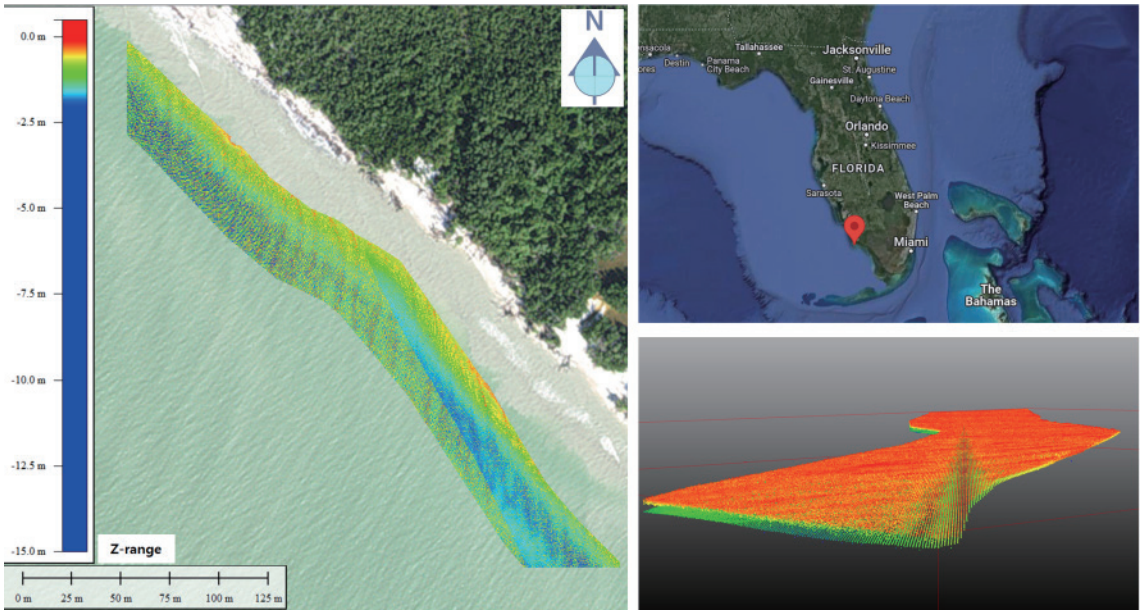


Fig. 3. (Color online) Test site: Marco Island in Florida, USA and VQ-880-G point cloud data.

Table 2
Test dataset.

System	Location	Year	Area (m^2)	No. of points	Point density (pts/m^2)	Z range (m)	Depth range (m)
Riegl VQ-880-G	Marco Island, Florida, USA	2016	14192	697044	49.12	−13.30–−0.09	0–1.81

acquired ABL data were labeled manually using the classification tool in Terrasolid TerraScan v3.4. For the training dataset, the point cloud data was divided into patches consisting of 10000 points each. During the actual training process, only 3000 points were randomly selected and used for training. This approach was intended to reduce computational complexity while maintaining a representative sample of the dataset for model training.

3.2 Preprocessing results

The preprocessing stage significantly affects the distribution of points in the point cloud, which in turn affects the performance of the segmentation model. To illustrate this, we compared the original point cloud data, the standard normalized data, and the *z*-axis-emphasized data with weights of 2 and 3. Figure 4(a) shows the original point cloud data without any preprocessing. Figure 4(b) shows the data obtained after standard normalization. Figures 4(c) and 4(d) present the data with *z*-axis-emphasized normalization using weights of 2 and 3, respectively.

In the original data, the points were distributed according to their raw measurements, which revealed a clear distinction between seabed and non-seabed points but with varying scales and densities that may complicate the learning process for the neural network. After standard normalization, the point cloud data was centered around the origin and scaled to a unit sphere. This step standardized the scale of the data, making the training process more stable and improving the model's ability to learn patterns across different scales. However, standard normalization treats all axes equally, which might not be optimal for distinguishing features that are more prominent along the *z*-axis.

By increasing the weight of the *z*-axis, the distinction between seabed and non-seabed points becomes more pronounced in the *z*-direction. This helps the model focus more on the vertical differences, which are crucial for accurate seabed segmentation. With a *z*-axis weight of 2, the

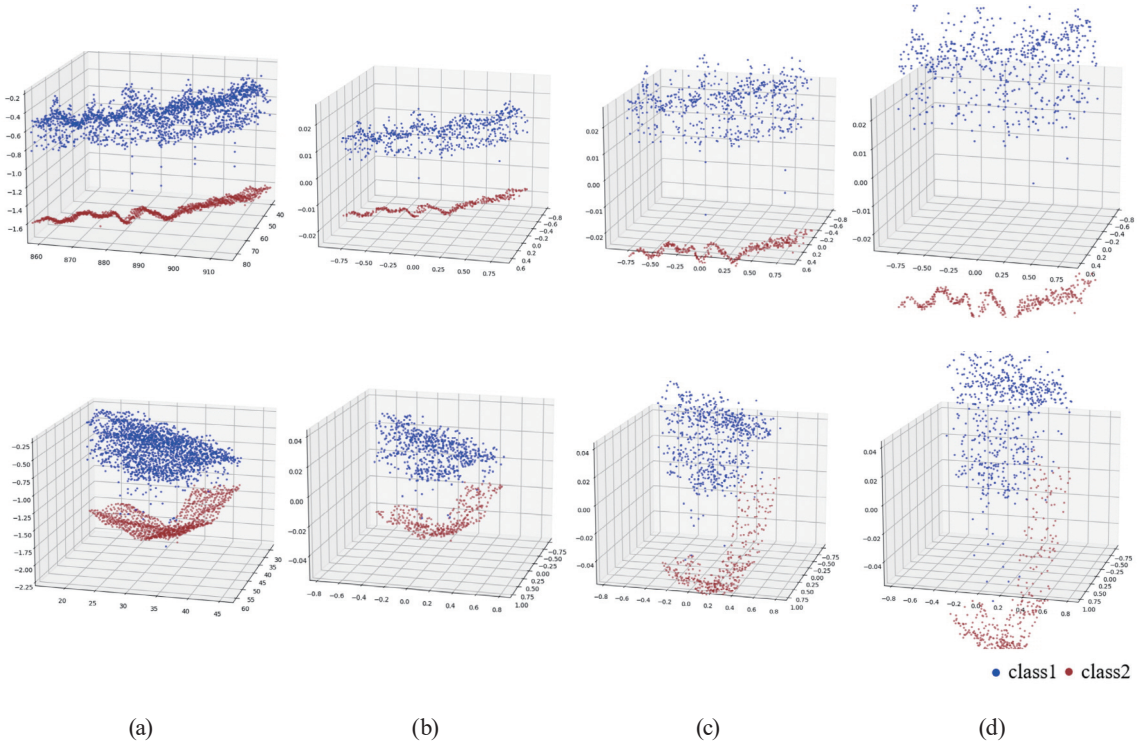


Fig. 4. (Color online) Example of a randomly selected 3D point cloud dataset: (a) original point cloud data without preprocessing, (b) after applying standard normalization, (c) after applying *z*-axis emphasis with a weight of 2, and (d) after applying *z*-axis emphasis with a weight of 3. Class 1 (blue points) represents the water surface and points excluding the seabed, including column points, whereas Class 2 represents the seabed.

points are more spread out along the *z*-axis, making it easier to distinguish between different classes. Further increasing the *z*-axis weight to 3 amplifies this effect. The points are even more distinctly separated along the *z*-axis. While this can improve the ability of the model to differentiate between seabed and non-seabed points, there is a risk of overemphasis, which may lead to the potential loss of horizontal spatial context.

3.3 Training results

The results of the experiments are summarized in Table 3, which presents the classification accuracy and loss metrics obtained from experiments conducted using both the original and window-based PointNet architectures. Each architecture was tested with three different normalization methods: standard normalization and *z*-axis-emphasized normalization with weights of 2 and 3. Additionally, the experiments with the enhanced PointNet architecture explored the effects of varying the window size along the *z*-axis (32, 64, and 128) to assess how these modifications affect the segmentation performance.

The results indicate that the enhanced PointNet with a window-based attention mechanism generally outperformed the original PointNet, particularly in most of the normalization settings and window sizes. However, there were instances where the original PointNet achieved a higher accuracy, suggesting that the effectiveness of the window-based approach may vary depending on specific data characteristics and settings. This variability implies that while the window-based mechanism provides a more sophisticated means of capturing local and global features, it may also introduce complexity that does not always lead to a higher performance across all scenarios. The results suggest that the window-based PointNet is especially effective in more complex settings where the point cloud data has significant spatial variability. In simpler cases, however, the original PointNet may still be competitive or even superior.

Table 3
Performance comparison of original and window-based PointNet models across different normalization methods. SN^* refers to standard normalization, whereas *Z-norm* $w = 2$ and $w = 3$ denote *z*-axis normalization with weights of 2 and 3, respectively.

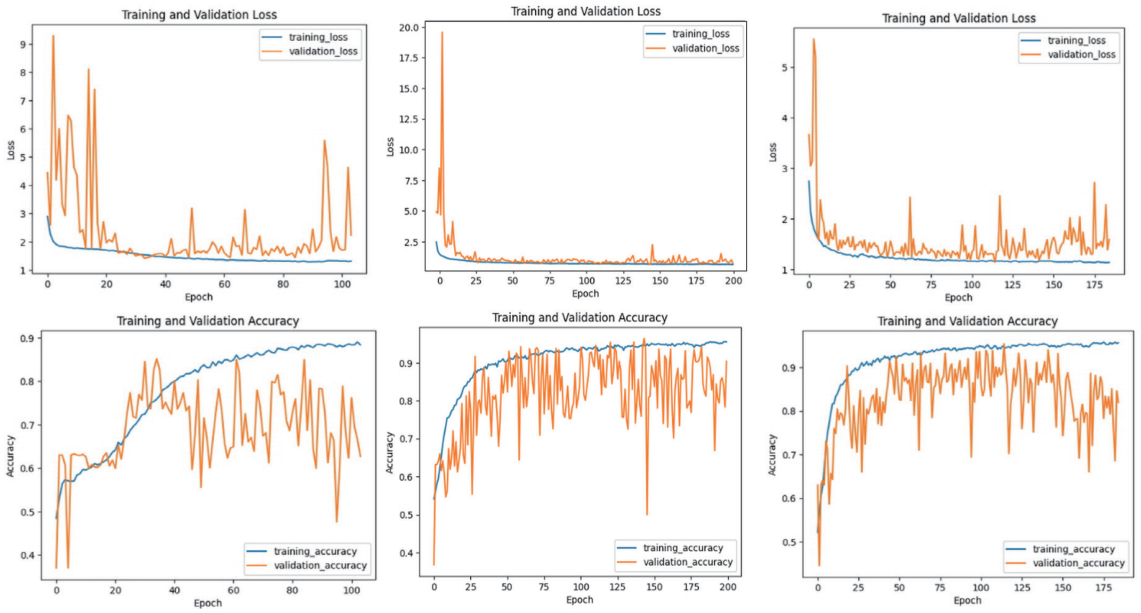
Network		Normalization method	Validation set		Training set	
			Loss	Accuracy	Loss	Accuracy
Original PointNet		SN^*	1.359	0.880	1.298	0.943
		Z-norm$_{w=2}$	1.348	0.910	1.230	0.947
		Z-norm$_{w=3}$	1.294	0.916	1.200	0.936
Enhanced PointNet	Window size = (32, 32, 32)	SN^*	1.328	0.853	1.310	0.884
		Z-norm$_{w=2}$	1.233	0.901	1.232	0.911
		Z-norm$_{w=3}$	1.284	0.917	1.198	0.934
	Window size = (32, 32, 64)	SN^*	1.405	0.825	1.367	0.831
		Z-norm$_{w=2}$	1.182	0.946	1.141	0.964
		Z-norm$_{w=3}$	1.197	0.948	1.143	0.954
	Window size = (32, 32, 128)	SN^*	1.351	0.854	1.257	0.905
		Z-norm$_{w=2}$	1.233	0.928	1.182	0.956
		Z-norm$_{w=3}$	1.220	0.928	1.197	0.959

3.3.1 Effect of normalization method

Comparing the results across different normalization methods—standard normalization and *z*-axis-emphasized normalization with weights of 2 and 3—revealed that emphasizing the *z*-axis improves classification accuracy. This is particularly relevant in bathymetric LiDAR data, where the *z*-axis contains critical information about the depth and surface of water bodies. The *z*-axis-emphasized normalization method with a weight of 2 provided the most balanced and accurate results, suggesting that it effectively enhances the distinguishability of points without excessively skewing the data distribution. When the weight was increased to 3, despite improvements over standard normalization, the gains were marginal, suggesting a point of diminishing returns.

The training graphs (Fig. 5) and the comparison of ground truth and prediction results (Fig. 6) demonstrate the impact of different normalization methods on the network's performance. The first method, standard normalization, treats all dimensions equally and demonstrates steady improvements in both training and validation accuracies. However, the accuracy plateaus relatively early, indicating that the model's ability to differentiate between classes is constrained by the equal treatment of the *z*-, *x*-, and *y*-axes. This suggests that the standard normalization might not fully exploit the vertical variations that are significant in bathymetric data.

When the *z*-axis was emphasized with a weight of 2, the model exhibited a marked improvement in accuracy and a more stable loss curve during training and validation. This enhancement suggests that the model benefits from the increased sensitivity to vertical differences, which are essential for accurately distinguishing between the different types of



(a) (b) (c)

Fig. 5. (Color online) Training and validation performance characteristics of enhanced PointNet under different normalization methods: (a) standard normalization, (b) *z*-axis normalized weight = 2, (c) *z*-axis normalized weight = 3 [window size = (32, 32, 64)].

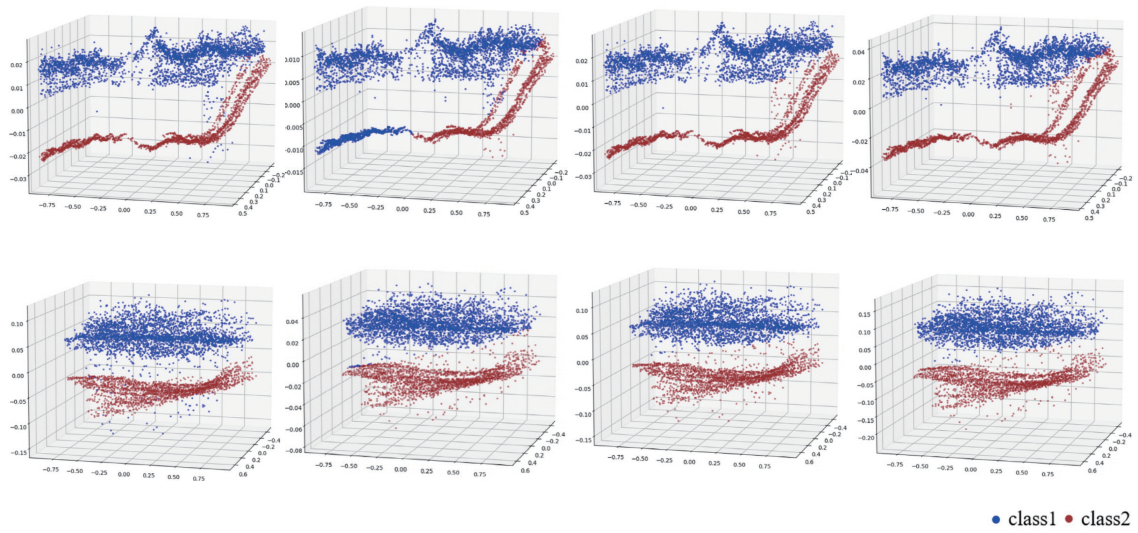


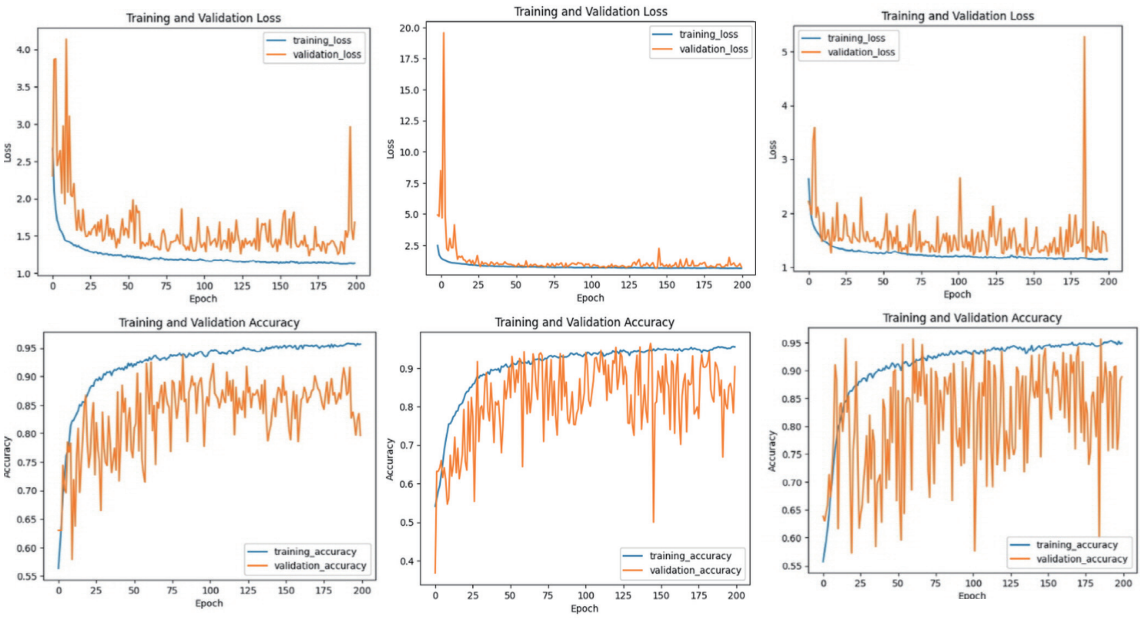
Fig. 6. (Color online) Comparison of ground truth and predicted point cloud classifications under different normalization methods: (a) ground truth, (b) standard normalization, (c) z-axis normalized weight = 2, and (d) z-axis normalized weight = 3. Class 1 (blue points) represents the water surface and points excluding the seabed, including column points, whereas Class 2 represents the seabed [window size = (32, 32, 64)].

point in the data. The focus on the z-axis allows the model to capture critical information that may be overlooked when using standard normalization. Further increasing the z-axis weight to 3 results in an even higher accuracy, indicating that the model is better at capturing vertical distinctions in the data. However, this increased sensitivity comes at a cost. The validation accuracy and loss curves exhibited greater fluctuations, suggesting that the model is more prone to overfitting. This behavior implies that while a higher z-axis weight can enhance the ability of the model to distinguish between classes, it may also make the model overly sensitive to noise in the z-dimension, leading to less generalizable results.

3.3.2 Effect of window size

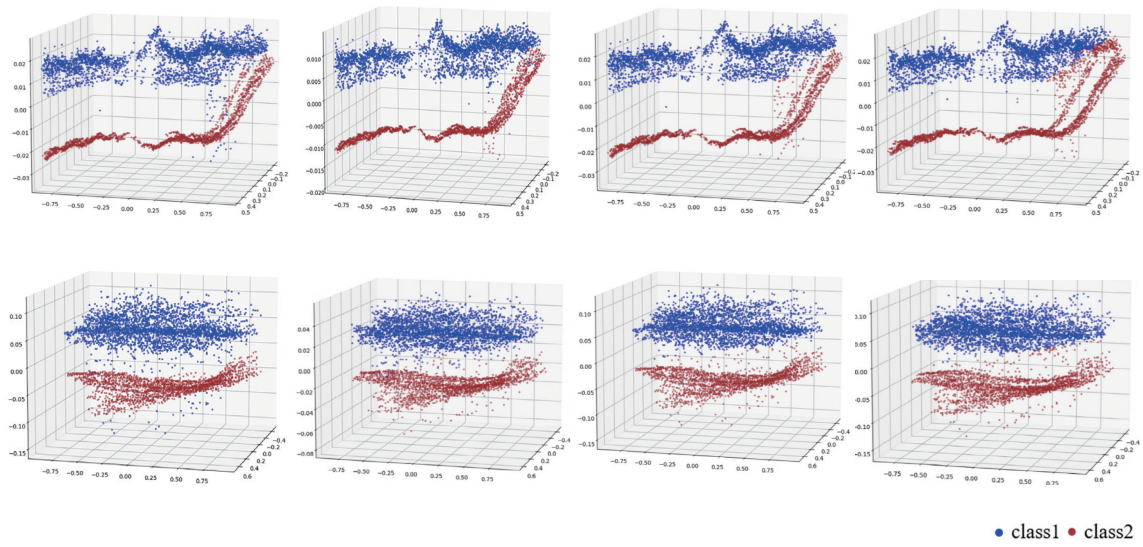
Figures 7 and 8 present the training and validation performance characteristics of the window-based PointNet architecture under different window sizes (32, 32, 32; 32, 32, 64; and 32, 32, 128). These figures demonstrate how the variation in window size affects both the model's learning and its ability to accurately classify seabed, water column, and surface points.

Figure 7 shows a consistently high training accuracy across all window sizes, indicating that the model effectively learns the patterns in the training data. However, the validation accuracy displays a high variability, particularly in the 32 × 32 × 64 configuration [Fig. 7(b)]. This fluctuation suggests that the model may experience challenges in generalization when using this specific window size, despite the strong performance on the training set. The increased fluctuation in validation accuracy could be indicative of overfitting, where the model adapts too closely to the training data at the expense of its performance on unseen data.



(a) (b) (c)

Fig. 7. (Color online) Training and validation performance characteristics of enhanced PointNet under different window sizes: (a) 32, 32, 32, (b) 32, 32, 64, and (c) 32, 32, 128 ($w_z = 2$).



(a) (b) (c) (d)

Fig. 8. (Color online) Comparison of ground truth and predicted point cloud classifications under different window sizes: (a) ground truth, (b) 32, 32, 32, (c) 32, 32, 64, and (d) 32, 32, 128. Class 1 (blue points) represents the water surface and points excluding the seabed, including column points, whereas Class 2 represents the seabed ($w_z = 2$).

Figure 8 illustrates the impact of different window sizes on the classification accuracy of the point cloud data and shows a comparison of the model's predictions against the ground truth. The comparison reveals that with the smallest window size of 32 × 32 × 32, the model tends to

produce compact and clustered classifications. This clustering can lead to less precise boundaries between the seabed and other classes, potentially due to the limited contextual information within smaller windows. When the window size increases to 32 × 32 × 64 [Fig. 8(c)], the model demonstrates better separation between seabed and sea surface points, indicating improved classification accuracy due to the additional vertical context. However, with the largest window size of 32 × 32 × 128 [Fig. 8(d)], an unexpected problem arises. The model starts to struggle with overlapping classifications between the seabed and sea surface points. This overlap suggests that the larger window size, while providing more context, may also introduce noise or irrelevant information, particularly in regions where the seabed and sea surface are close together. Thus, the ability of the model to distinguish between these two classes decreases, leading to misclassifications in areas where these points are close together.

3.4 Discussion

The attention mechanism allowed the network to focus on localized regions within the point cloud, capturing more detailed spatial relationships and improving the ability of the model to differentiate between the seabed and surface points. However, the results also highlight that while the enhanced architecture generally outperformed the original PointNet, there were instances where the original model, particularly under certain normalization methods, yielded a higher performance. This indicates that while the enhancements provide substantial benefits, they are not universally superior across all scenarios, and the simplicity of the original PointNet may be advantageous in certain contexts.

Standard normalization provided a baseline for comparison, and while it achieved reasonable performance, the introduction of the *z*-axis-emphasized normalization had distinct advantages in scenarios where the vertical separation between classes was critical. From the results, emphasizing the *z*-axis with a weight of 2 or 3 generally improved the ability of the model to separate seabed points from surface points. However, this enhancement came at a cost; increasing the *z*-axis weight excessively overemphasized vertical information, which in turn caused some misclassifications, particularly in areas where the seabed and surface were close together. Thus, while *z*-axis normalization is effective, it requires careful tuning to balance the effects of vertical and horizontal spatial information.

The choice of window size had a significant impact on the model's performance. Smaller window sizes (32 × 32 × 32) resulted in more compact classifications with less spatial context, which could limit the ability of the model to correctly classify points in complex regions. On the other hand, larger window sizes (32 × 32 × 128) provided more context but introduced the risk of misclassification in areas where different classes were closely aligned, such as the seabed and sea surface. The intermediate window size (32 × 32 × 64) appeared to strike a balance by providing sufficient context without introducing significant noise or overlap in classification.

Overall, the enhanced PointNet architecture with window-based attention demonstrates substantial improvements in the classification of bathymetric LiDAR data. However, the results also underscore the importance of selecting appropriate hyperparameters, particularly the window size and normalization method. The optimal configuration may vary depending on the

specific characteristics of the dataset and the spatial relationships within the point cloud. Future research could explore adaptive methods for determining these parameters, potentially leading to more robust and generalized models for bathymetric LiDAR data segmentation. Additionally, investigating other deep learning architectures and attention mechanisms could further enhance performance, particularly in challenging environments with closely spaced or overlapping features.

4. Conclusions

In this study, we introduced an enhanced version of PointNet tailored to the challenges presented by bathymetric LiDAR data, particularly in classifying seabed and water surface points. The key modifications included the integration of a window-based attention mechanism and the application of orthogonal regularization, which collectively aimed to improve the ability of the model to capture and utilize the complex spatial information inherent in three-dimensional point clouds.

The experimental results demonstrated that the enhanced PointNet architecture generally outperformed the original PointNet across various scenarios, particularly when an appropriate window size and a normalization method were selected. The introduction of the z-axis-emphasized normalization proved particularly effective in scenarios where vertical separation between classes was critical, although it required careful tuning to avoid overemphasis on vertical features.

Furthermore, the choice of window size significantly affected model performance. While smaller window sizes provided compact classifications with limited context, larger window sizes introduced more context at the risk of misclassifying closely aligned classes. An intermediate window size ($32 \times 32 \times 64$) offered a balance, yielding improved performance by providing adequate spatial context without excessive overlap.

Despite the improvements offered by the enhanced PointNet, we also highlighted that the original PointNet is still advantageous for certain scenarios, particularly where simplicity and less complex spatial relationships are involved. This finding suggests that the best approach may depend on the specific characteristics of the dataset and the application context. The enhanced PointNet architecture shows promise for improving the classification of bathymetric LiDAR data, but it also underscores the importance of careful selection of model parameters and preprocessing techniques. One limitation of this study is the reliance on specific window sizes and normalization methods, which may not generalize across all datasets. Future work can focus on developing adaptive mechanisms for parameter selection and exploring alternative deep learning architectures to further refine and optimize the classification of complex three-dimensional point cloud data.

Acknowledgments

This research was supported by the Korea Institute of Marine Science & Technology (KIMST) funded by the Ministry of Oceans and Fisheries (RS-2023-00254717).

References

1 G. C. Guenther: The Dem Users Manual, Maune, D.F., Ed. (ASPRS Publications, Bethesda, MD, USA, 2007) 2nd ed., Chap. 8.
2 J. C. Brock and S.J. Purkis: J. Coastal Res. **10053** (2009) 1. https://doi.org/10.2112/SI53-001.1.
3 G. Chust, M. Grande, I. Galparsoro, A. Uriarte, and Á. Borja: Estuar. Coast. Shelf Sci. **89** (2010) 200. https://doi.org/10.1016/j.ecss.2010.07.002
4 T. Kumpumäki, P. Ruusuvuori, V. Kangasniemi, and T. Lipping: Remote Sens. **7** (2015) 13390. https://doi.org/10.3390/rs71013390
5 X. Li, C. Liu, Z. Wang, X. Xie, D. Li, and L. Xu: Meas. Sci. Technol. **32** (2020) 032002. https://doi.org/10.1088/1361-6501/abc867
6 Y. Chen, Y. Le, L. Wu, S. Li, and L. Wang: Sensors **22** (2022) 7681. https://doi.org/10.3390/s22197681
7 L. Graham: Photogrammetric Eng. Remote Sens. **71** (2005) 777.
8 R. Quintero: LiDAR Mag. **3** (2013) 68.
9 H. Kim, J. Jung, J. Lee, and G. Wie: Can. J. Remote Sens. **49** (2023) 2172957. https://doi.org/10.1080/07038992.2023.2172957
10 K. Saylam, R. J. Hupp, R. A. Averett, F. W. Gutelius, and W. B. Gelhar: Int. J. Remote Sens. **39** (2018) 2518. https://doi.org/10.1080/01431161.2018.1430916
11 T. Kutser, I. Miller, and D. L.B. Jupp: Estuar. Coast. Shelf Sci. **70** (2006) 449. https://doi.org/10.1016/j.ecss.2006.06.026
12 J. Jung, J. Lee, and C.E. Parrish: Remote Sens. **13** (2021) 3665. https://doi.org/10.3390/rs13183665
13 K. Lowell and B. Calder: Remote Sens. **13** (2021) 1604. https://doi.org/10.3390/rs13091604
14 T. Kogut and A. Slowik: IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens. **14** (2021) 1959. https://doi.org/10.1109/JSTARS.2021.3050799
15 F. Eren, S. Pe'eri, Y. Rzhanov, and L. Ward: Remote Sens. Environ. **206** (2018) 260. https://doi.org/10.1016/j.rse.2017.12.035
16 Y. Zhang, Y. Zheng, S. Qiao, and Q. Tian: Pattern Recognit. **103** (2020) 107298. https://doi.org/10.1016/j.patcog.2020.107298
17 G. Litjens, T. Kooi, B. E. Bejnordi, A. A. A. Setio, F. Ciompi, M. Ghafoorian, and J. A. W. M. van der Laak: Medical Image Analysis **42** (2017) 60. https://doi.org/10.1016/j.media.2017.07.005
18 A. Krizhevsky, I. Sutskever, and G.E. Hinton: Commun. ACM **60** (2017) 84. https://doi.org/10.1145/3065386
19 X. Liu, Y. Wang, and Z. Li: Remote Sens. **10** (2018) 1234. https://doi.org/10.3390/rs10081234
20 H. Zhang, Q. Wu, and H. Zhao: IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens. **12** (2019) 4567. https://doi.org/10.1007/978-3-031-16213-8_6
21 C. R. Qi, H. Su, K. Mo, and L. J. Guibas: Proc. IEEE Conf. Computer Vision and Pattern Recognition (IEEE, 2017) 652–660
22 M. P. Weinstein and K.L. Heck: Compos., Structure Community Ecology Mar. Biol. **50** (1979) 97.
23 NOAA National Geodetic Survey: https://catalog.data.gov/dataset/2016-noaa-ngs-topobathy-lidar-marco-island-fl1 (accessed August 2024).
24 Cloud Compare: https://www.cloudcompare.org/doc/wiki/index.php?title=SOR_filter (accessed August 2024).

About the Authors

Ahram Song received her B.S. degree from Inha University, South Korea, in 2012 and her Ph.D. degree from Seoul National University, South Korea, in 2019. Since 2021, she has been a professor at KyungPook National University. Her research interests are in deep learning and change detection. (ars@knu.ac.kr)

Jaebin Lee received his B.S. degree from Yonsei University, South Korea, in 2000 and his Ph.D. degree from Seoul National University, South Korea, in 2008. Since 2009, he has been a professor at Mokpo National University. His research interests are in the full-waveform processing and classification of airborne bathymetric LiDAR data. (lee2009@mnu.ac.kr)