Int. J. Adv. Eng. Pure Sci. 2021, ASYU 2020 Special Issue: 57-66
DOI: 10.7240/jeps.897306

RESEARCH ARTICLE / ARAŞTIRMA MAKALESİ

A Comparative Study of Point-Based Deep Learning Techniques for Semantic Classification in Search and Rescue Arenas

Arama ve Kurtarma Alanlarında Anlamsal Sınıflandırma için Nokta Tabanlı Derin Öğrenme Tekniklerinin Karşılaştırmalı Bir Çalışması

Kaya TURGUT [1], Burak KALECİ [1]

[1]*Eskisehir Osmangazi University, Electrical and Electronics Engineering Department, Eskisehir, Turkey*

Abstract
Post-disaster indoor environments, which occur after calamities such as floods, fires, and poisonous material spread, could include serious risks for search and rescue teams. For example, the building's structural integrity could be corrupted, and some harmful substances for humans and animals could exist. Exploiting robots could prevent search and rescue teams from these risks. Nevertheless, robots need to possess advanced techniques to produce high-level information from raw sensor data in these harsh environments. This study aims to explore the positive and negative aspects of point-based deep learning architectures for the semantic classification of ramps in search and rescue test arenas, which are proposed by the National Institute of Standards and Technology (NIST). Also, we take into account walls and terrain since they can provide crucial information for robots. In this study, we opted to utilize point cloud data that is robust against lousy illumination conditions, which is frequently encountered in post-disaster environments. We used the ESOGU RAMPS dataset that contains point cloud data captured from a simulated environment similar to NIST search and rescue arenas. We selected PointNet, PointNet++, Dynamic Graph Convolutional Neural Network (DGCNN), PointCNN, Point2Sequence, PointConv, and Shellnet point-based deep learning architectures to analyze their performance for semantic classification of ramps, walls, and terrain. The test results indicate that accuracy of semantic classification is over 90% for all architectures.
Keywords: point-based deep learning, semantic classification, NIST ramps, point cloud data.

Öz
Sel baskını, yangın ve zehirli madde yayılımı gibi felaketlerden sonra meydana gelen afet sonrası iç ortamlar, arama ve kurtarma ekipleri için ciddi riskler barındırabilir. Örneğin, binanın yapısal bütünlüğü bozulmuş ve insanlar ve hayvanlar için bazı zararlı maddeler mevcut olabilir. Arama ve kurtarma ekiplerinin bu risklerden korunmasını sağlayabilmek için robotlardan yararlanılabilir. Bununla birlikte, robotların bu zorlu ortamlarda ham algılayıcı verilerinden üst düzey bilgi üretmek için gelişmiş tekniklere sahip olması gerekir. Bu çalışma, Ulusal Standartlar ve Teknoloji Enstitüsü (NIST) tarafından önerilen arama kurtarma test alanlarında bulunan rampaların anlamsal sınıflandırması için nokta tabanlı derin öğrenme mimarilerinin olumlu ve olumsuz yönlerini araştırmayı amaçlamaktadır. Ayrıca robotlar için çok önemli bilgiler sağladıklarından dolayı duvarlar ve zeminde dikkate alınmıştır. Bu çalışmada, afet sonrası ortamlarda sıklıkla karşılaşılan kötü aydınlatma koşullarına karşı dayanıklı olan nokta bulutu verilerini kullanmayı tercih ettik. NIST arama ve kurtarma alanlarına benzer bir ortamdan alınan nokta bulutu verilerini içeren ESOGU RAMPS veri kümesini kullandık. Rampaların, duvarların ve zeminin anlamsal sınıflandırma performanslarını analiz etmek için PointNet, PointNet ++, Dinamik Grafik Evrişimli Sinir Ağı (DGCNN), PointCNN, Point2Sequence, PointConv ve Shellnet nokta tabanlı derin öğrenme mimarilerini seçtik. Test sonuçları, anlamsal sınıflandırma doğruluğunun tüm mimariler için %90'ın üzerinde olduğunu göstermektedir.
Anahtar Kelimeler: nokta tabanlı derin öğrenme, anlamsal sınıflandırma, NIST rampları, nokta bulutu verisi.

I. INTRODUCTION

In recent years, attempting to use robots for search and rescue missions in post-disaster environments, which occur after calamities such as floods, fires, and poisonous material spread, is among the hot topics in the robotic community. The main reasons for that are happening new collapses or leakage of harmful substances for humans and animals while search and rescue teams perform their tasks. Therefore, exploiting robots in these missions could prevent undesirable accidents and new casualties. However, post-disaster environments consist of some challenges even for robots such as uneven terrain and lousy illumination conditions due to these environments' dusty nature and power outages. To cope with these difficulties, robots need to possess advanced techniques to produce high-level information (semantic information of scenes or objects) from raw sensor data. Fortunately, robots have begun to utilize these advanced techniques owning to incoming perception technologies and developments in computer vision algorithms. Nevertheless, the advantages and disadvantages of these technologies and algorithms should be observed regularly for determining new research areas in the search and rescue domain. To achieve this, competitions related to search and rescue missions have been organized yearly by RoboCup society since 2001.

Corresponding Author: Burak KALECİ, Tel: +905557375966, e-posta: burakaleci@gmail.com
Submitted: 15.03.2021, **Revised:** 07.07.2021, **Accepted:** 07.07.2021

The RoboCup rescue competitions mainly aim to improve robots' abilities in autonomous navigation, mapping, and finding victims. Besides, these competitions monitor the performance of new technologies in software and hardware and encourage the researchers to introduce new challenges for robots. Kitano and Tadokoro [1] assessed the first RoboCup rescue competition considering the initial standards and evaluation metrics. They examined requirements that the robots need to have, and they projected future works according to these requirements. One of them was to evaluate robot performance in standard test environments. For that reason, NIST introduced reference test arenas for search and rescue missions in 2003 [2]. Figure 1 shows an example of these test arenas. In the first years of competitions, participant teams generally preferred to use visual and 2D laser range data to perform search and rescue missions. For example, the PoAReT team [3], the winner of the Virtual Robot Competition at RoboCup 2012, applied the simultaneous localization and mapping (SLAM) technique to 2D laser scans for building a map of reference test arenas. They also produced semantic information while determining rooms and corridors through the detection of doorway locations. This was the pioneering study in the search and rescue missions that considers semantic information to the best of our knowledge.

Sheh et al. [4] evaluated the RoboCup rescue competitions in the 16th year. They noticed that participant teams of competitions improved the abilities of robots while integrating new sensors into robot systems such as RGB-D cameras, LiDARs, and 3D laser scanners over the years. Besides, they observed that robots were able to perform search and rescue missions adequately in challenging environments. Robot Operating System (ROS) [5] have been begun to use in the RoboCup rescue competitions since 2017. This could be a milestone for competitions because ROS provides numerous packages that researchers can easily integrate their systems to achieve more complex tasks. The participant teams utilized GMapping [6] and Hector SLAM [7] packages to represent the environments with 2D maps. Then, OctoMap [8] and Real-Time Appearance-Based Mapping (RTAB-Map) [9] approaches were used to produce 3D information about the environment. OctoMap is an octree-based representation method, and it does not take into account semantic information. On the other hand, RTAB-Map exploited feature extractors to generate semantic information while recognizing frequently encountered objects in daily life. However, RTAB-Map is not able to extract semantic information for ramps at reference test arenas since ramps are specific pieces for search and rescue missions.

Figure 1. An example reference test arena [10]

Although participant teams of the RoboCup rescue competitions generally do not concentrate on producing semantic information for search and rescue missions, semantic classification of walls, ramps, and terrain via point cloud data has been addressed in previous studies. These approaches could be separated into two categories. In the first category, the studies handled the segmentation problem, which clusters points into a segment considering their features such as x, y, z coordinates, point normals, and colors. The reviews proposed by Nguyen and Le [11], Grilli et al. [12], and Xie et al. [13] mainly divide the segmentation approaches into five groups: Edge-based [14], region growing [15], model-based [16, 17], clustering-based [18], and graph-based [19]. These approaches have been frequently applied for segmentation problems since their implementations are available on Point Cloud Library (PCL) [20]. Besides, Eruyar et al. [21] examined the performance of segmentation approaches situated in PCL for structural planar surfaces. However, only region growing and RANSAC approaches were employed to classify walls, ramps, and terrain. Region growing begins with determining seed points and searches points within a predetermined radius or a number of neighbors around the seed points to identify points that have similar features with seed points. The main disadvantages of region growing approaches are high time complexity and sensitivity to rapid changes on point features. In contrast, RANSAC requires less computational load, and robust against noise, in other words, rapid changes. However, it does not utilize local information for segmentation, and this can cause clustering points that have similar mathematical models into a segment, although they belong to different planes. The approaches placed in the first category classified points considering plane equations to obtain semantic classes of points.

In the second category, machine and deep learning techniques were used to determine the semantic class of points. Machine learning techniques firstly determine suitable descriptors for the problem, and then they extract features. This step is probably the most challenging part because the selected descriptors directly affect the success of these techniques. Besides, 3D descriptors generally tend to overfit since

they are significantly high dimensional. On the other hand, deep learning architectures have become popular in recent years since they are able to provide useful features according to the problem. For example, Deng et al. [22] applied Convolutional Neural Network (CNN) approach to visual data and depth images acquired from a post-disaster environment similar to NIST reference test arenas to determine the semantic class of points.

The first attempts to exploit deep learning techniques with point cloud data are to use CNN because of success of CNNs with image data. However, CNN approaches cannot be employed to point cloud data directly since its permutation invariant and unstructured characteristic. For that reason, the deep learning techniques that have utilized point cloud data can be separated into two groups: direct (point-based) and indirect approaches [23]. Point-based approaches accept raw point cloud data, while indirect approaches convert unstructured point cloud data into a 2D or 3D structured form before receiving the data. It can be observed from the previous studies that indirect approaches have some drawbacks such as quantization artifact, loss of the geometric details, and computational cost of conversion. In consequence, the researchers have generally preferred to develop point-based approaches [23, 24]. Guo et al. [24] gave detailed information about 3D deep learning approaches and categorized them for classification, segmentation, object detection, and tracking problems. They addressed the point-based methods for semantic classification of point cloud data into four groups: point-wise Multi-Layer Perceptrons (MLP), point convolution, graph-based, and Recursive Neural Networks (RNN) based.

PointNet [25] was the first point-based deep learning architecture accepted as a milestone since it works directly on unordered and permutation-invariant point cloud data. PointNet utilizes the point-wise MLP followed by maximum pooling to summarize the global feature. However, PointNet does not extract the local relationship because it considers all points in the point cloud as individually. Many point-wise architectures like PointNet++ [26], ShellNet [27], and PointWeb [28] were proposed based on PointNet to encode local neighborhood information because of simplicity and performance.

Applying the convolution kernel to the point cloud is not straightforward because data may have missing parts and does not have a regular pattern. Some studies applied continuous convolution [29, 30] or discrete convolution [31] to point cloud data. KPConv [29] learns the weights of kernel points defined in Euclidean space. Linear correlation is applied between kernel points and the points around the kernel points. In the PointConv [30], convolution kernels defined as nonlinear weighting function on 3D space. The kernel weights are learned with MLP layers. PointCNN [31] learns the transformation matrix to order canonically for applying the discrete convolutional operator.

In the graph-based methods, the points are considered as nodes, and the distance between nodes is treated as edges [32, 33]. Landrieu et al. [32] extended the superpixel term of images as superpoint for point clouds to partition into homogeneous parts. They introduced the superpoint graph to expose the contextual information between object parts. DGCNN [33] uses each point as a node and defines the graph for each local region. The features are extracted over edges, and the graph is updated in all layers.

RNN-based architecture aims to capture contextual information of local parts [34-36]. 3P-RNN [34] utilized a two-directional RNN structure to exploit the long-range relationship of uniformly-spaced blocks along x and y directions. Point2Sequence [35] used the RNN structure to extract the correlation of multi-scale local areas. An attention mechanism is used to highlight the critical feature of multi-scale local areas. RSNet [36] proposed the slice pooling layer, which slices the point cloud x, y, and z coordinate independently to project unordered point cloud onto a sequential form.

In this study, semantic classification of point cloud data as walls, ramps, and terrain is aimed. Producing semantic information about walls, ramps, and terrain could promote the mapping and navigation tasks of robots in a post-disaster environment. An example of that is the robot could augment the environment's representation with semantic information to generate maps that first-responders easily read. In this way, the robot can also improve its path plan while adding appropriate targets according to ramps. The robot can navigate without losing its balance when it passes over the ramps considering these target points. Besides, the robot can regulate its speed when being aware position and orientation of ramps.

In this study, we prefer to use point cloud data because visual and/or 2D laser range data could not be adequate for post-disaster environments. The success of studies that utilized visual data highly depends on the illumination condition of the environment, and the post-disaster environments have lousy illumination conditions due to these environments' dusty nature and power outages. Although 2D laser range data robust against these conditions, it only captures information about the plane at the height that the laser scan is placed. Besides, 2D laser range data cannot provide any information about the ramps below that height. On the other hand, point cloud data could describe 3D characteristics of walls, ramps, and terrain and cannot be influenced by the environment's illumination condition. After that point, we analyzed point-based and indirect deep learning approaches for semantic

classification of point cloud data, and we decided to use point-based approaches when disadvantages of indirect approaches are thought. In our previous works [37, 38], we showed that PointNet, PointNet++, DGCNN, and PointCNN point-based deep learning architectures are classified walls, ramps, and terrain with over 90% accuracy. This study aims to extend our previous works by considering Point2Sequence, PointConv, and Shellnet point-based deep learning architectures. In this way, these architectures' positive and negative aspects could be revealed for semantic classification of walls, ramps, and terrain in reference test arenas. To train and test these architectures, we used the ESOGU RAMPS dataset [39]. The architectures were evaluated with recall, precision, Intersection over Union (IoU), Mean Intersection over Union (MIoU), and accuracy metrics.

The rest of the paper is organized as follows: Section 2 briefly explains the point-based deep learning architectures. Section 3 describes the ESOGU RAMPS dataset. Section 4 and 5 presents experimental works and concludes the paper, respectively.

II. METHODS

The point cloud data is defined as a set of points that contain x, y, and z coordinates. Besides, other features such as point normal and color could be attached to each point. Although point cloud has a simple nature, a variety of architectures were proposed due to its unstructured, permutation, and rotation invariant characteristic. In this section, notable attributes of PointNet, PointNet++, DGCNN, PointCNN, Point2Sequence, PointConv, and Shellnet point-based deep learning architectures are concisely described.

2.1. PointNet

The first architecture that receives raw point cloud data, in other words, point-based deep learning architecture, is PointNet. It employs successive Multi-Layer Perceptron (MPL) layers to point's x, y, and z coordinates to learn weight matrices. These matrices are shared among points for each feature channel as similar to CNN structure. Besides, normal, curvature, and color features can be used in the feature extraction process. It is important to note that PointNet assesses each point individually and also independently from other points. Namely, neighbor points of a point do not take into account for feature extraction. This could be the most characteristic feature of the PointNet architecture. PointNet provides a maximum pooling method named as the symmetric function to obtain the global feature. The symmetric function receives the feature vector of all points and determines each feature channel's maximum values to summarize a single global feature vector. The classification scores for each category of points are acquired by successive MLP layer again after the global and local features are concatenated.

2.2. PointNet++

PointNet++ architecture derives from the PointNet. The main difference between PointNet and PointNet++ is that PointNet++ considers neighbors of points to extract the feature vector of each point. The architecture first selects center points to construct local regions. These center points are specified according to a predefined number of nearest neighbors or a predefined radius. After local regions are determined, the feature vector of each point that belongs to a local region is extracted by employing PointNet to point's x, y, and z coordinates. The symmetric function then takes the feature vector of all points in a local region and determines the region feature vector. This process is repeated for each local region. At that point, PointNet++ continues with the successive layer while extending the local regions hierarchically based on x, y, and z coordinates. In this way, local regions cover large portions, and the region feature vectors begin to approximate the scene's characteristics. After successive layers, similar to PointNet, the symmetric function summarizes local region feature vectors to obtain a global feature vector. In successive feature encoding layer, point number is subsampled. However, all original points feature is required to classify points semantically. A distance based interpolation technique is proposed to propagate the features of sampled points to original points. After interpolated feature are concatenated skip linked feature from feature encoding layer, PointNet is used to update these features. Eventually, the semantic class of each point is determined.

2.3. DGCNN

DGCNN architecture is categorized as graph-based, according to the review presented by Guo et al. [24], because it constructs local regions similar to PointNet++, and it builds a graph for each local region. The main difference between PointNet++ and DGCNN is that DGCNN does not expand local regions hierarchically. DGCNN first selects center points to construct local regions. Then, it searches a predefined number of nearest neighbors of these center points to determine the points that belong to local regions. To do that, DGCNN uses distances between a point and center points, which are calculated with point's x, y, and z coordinates in the first layer, while feature space distances are utilized in successive layers. At that point, the graphs for each local region are formed. The nodes of these graphs are the points in the local regions. However, it is crucial to notice that the edges only exist for the center point and other points. The edge weights are calculated considering spatial and feature space distances in the first and successive layers, respectively. This is the second difference between PointNet++ and DGCNN. After edge weights are determined, they are used in the feature extraction process by employing MLPs. The authors are named these steps as EdgeConv operator. In the DGCNN architecture, first EdgeConv

operator estimates features, and then PointNet accepts these features to classify points.

2.4. PointCNN

In contrast to PointNet, PointNet++, and DGCNN architectures that consider points individually while extracting features, PointCNN calculates features by applying convolution to a point and its neighbors. Unfortunately, convolution approaches, just like CNN, cannot be directly employed to point cloud data due to its permutation invariant nature. For that reason, the authors proposed a convolution operator, which is named as X-Conv. PointCNN first constructs local regions similar to previously mentioned architectures. Then, for each local region, a transformation matrix is learned through X-Conv operator, and according to the matrix, points in a local region are weighted and canonically ordered. Lastly, convolution is employed local regions to extract features. Similar to PointNet++ local regions are hierarchically expanded in the successive layers. PointCNN propagate summarized global into point feature with skip linked X-Conv.

2.5. Point2Sequence

Point2Sequence is an RNN-based architecture. Apart from the previously explained architectures, Point2Sequence does not have successive layers since it utilizes Long Short-Term Memory (LSTM) structure to extract features. Point2Sequence constructs local regions just like PointNet++. The main difference between Point2Sequence and PointNet++ is that Point2Sequence constructs more than one local region with different radius values around a center point, namely multi-scale concentric local regions. Then, features are determined separately for each different-scale local region. After features corresponding to multi-scale concentric local regions are obtained for a center point, RNN is employed to learn the correlation between these features. In this way, Point2Sequence intends to reveal contextual information about local regions. The relationship between features that belong to a local region is stored in LSTM, and the attention approach puts forward hidden states of LSTM. Consequently, the local region features are calculated through a context vector, which is built consolidation of different scale features. The global feature vector is propagated from shape level to point level by using the interpolation techniques in PointNet++.

2.6. PointConv

The continuous convolution-based PointConv applies the convolution operator to each point in the point cloud in a similar way as traditional 2D image convolution. Firstly, the local region is defined around each point, and the convolution kernel weight is learned by using weight-shared MLP over the relative position of points in the local neighborhood. Because it uses the weight-shared kernel across all points, permutation invariance is satisfied. The density function is used to re-weight convolution filter weight to handle non-uniform density in local regions. The learned convolution kernel is applied to the feature of the local points, and the encoded feature of each point is obtained. Similar to PointNet++, the feature encoding module consists of sampling, grouping, and PointConv layer to learn feature hierarchically. Also, the PointDeconv layer is introduced to increase the point number, which is decreased in the feature encoding layers by applying interpolation and PointConv convolution.

2.7. Shellnet

ShellNet architecture consists of the ShellConv convolution operator based on point-wise MLP and convolution approaches. In this architecture, firstly, the local region is constituted in a similar way as PointNet++. However, kNN neighborhoods of representative points are divided into multi-scale concentric shells. A fixed number of points is assigned to each shell. In each shell, MLP is used to encode relative point coordinates to higher-dimensional features. After the features came from the previous layer and encoded features are concatenated, maximum pooling is used to summarize the points feature in each shell. Thanks to maximum pooling, point order ambiguity is resolved in each shell. Also, by nature of the shell, it is ordered from inside to outside. Then, 1D convolution is applied to the shell features. Shellconv layers are added hierarchically to extract the abstract feature of the entire point cloud. Deconvolution layers with ShellConv are used to increase point size to the original point number for the point-wise classification.

III. THE ESOGU RAMPS DATASET

Gazebo [40] simulation environment and ROS [5] were utilized to construct the ESOGU RAMPS dataset. An Asus Xtion Pro RGB-D camera was emplaced on a Pioneer 3-AT mobile robot, and 681 scenes were captured. For training, 581 scenes were randomly selected, and the remaining 100 scenes were separated for testing. In each scene, there are four classes: wall, terrain, inclined, and flat ramps. Examples for the ESOGU RAMPS dataset are presented in Figure 2. The point cloud data is shown in the left column. Wall, terrain, inclined, and flat ramps classes are described with red, yellow, blue, and magenta. The dataset also provides RGB images, which are shown in the right column. The details about the dataset are given in [37, 38] and you can download the dataset from [39].

IV. EXPERIMENTAL WORKS

4.1. Experimental Setup

PointNet, PointNet++, DGCNN, PointCNN, Point2Sequence, PointConv, and Shellnet point-based deep learning architectures were implemented with Tensorflow library [41] in Python programming

language for the semantic classification of scenes placed in the ESOGU RAMPS dataset. Before applying these architectures, we preprocessed the data, which is a requirement for point-based deep-learning architectures. First, the NaN values that correspond to unmeasured points were removed. Besides, using the whole scene to feed these architectures is making difficult to identify local features and causes losing data. Therefore, the point cloud data was separated into 1 m^2 blocks in xy plane to avoid these drawbacks. Lastly, the number of points that belong to a block must be a fixed number, and it was selected as 4096 in this study. The farthest point algorithm was employed for upsampling and downsampling to fix the number of points in blocks. Also, we discard the blocks that have less than 500 points. Although these architectures are able to process color and normal features of points, we only used point's x, y, and z coordinates. In the training process, we used default parameters for all architectures.Recall, precision, Intersection over Union (IoU), Mean Intersection over Union (MIoU), and accuracy (Acc) metrics were used to analyze the architectures' efficiency. True positive (TP) and false positive (FP) are defined as a correctly and incorrectly classified sample, which is owned to a positive class, respectively. On the other hand, false negative (FN) is an incorrectly classified sample to a positive class while the sample belongs to a negative class. The recall value of a class is the ratio of the true positive and total number of samples of that class (Equation (1)). The ratio of the true positive and total number of classified samples of a class is called the class's precision value (Equation (2)). The Intersection over Union of a class is the ratio of the true positive and the summation of the total number of samples and incorrectly classified samples (Equation (3)). Mean Intersection over Union (MIoU) is the mean of IoU of classes (Equation (4)). Lastly, accuracy is defined as the ratio of the total number of true positive for all classes and the total number of samples (Equation (5).

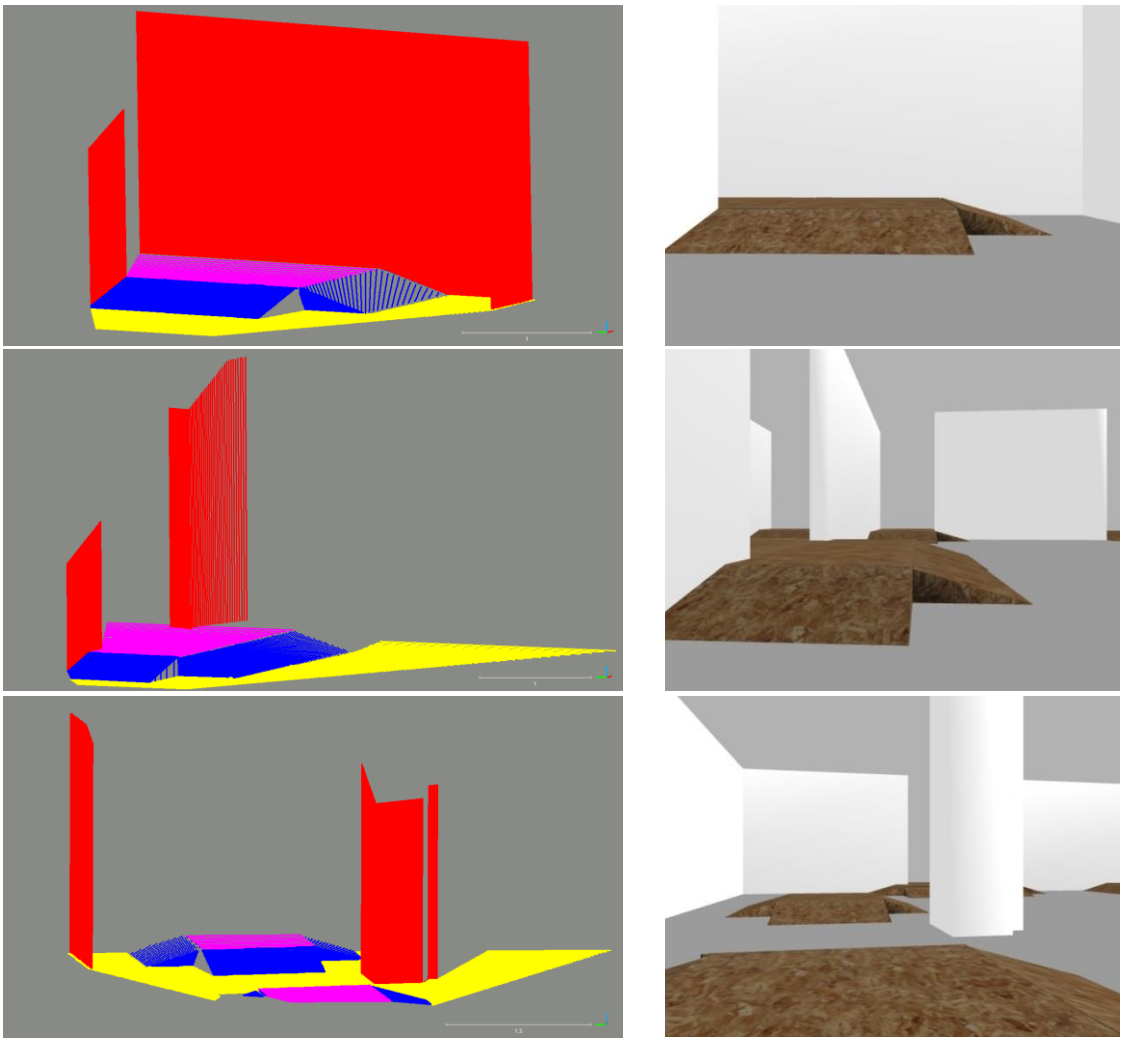


Figure 2. Examples for the ESOGU RAMPS dataset. The left column shows the point cloud data. The right column indicates the corresponding RGB images.

$$recall = \frac{TP}{TP + FN} \quad (1)$$

$$precision = \frac{TP}{TP + FP} \quad (2)$$

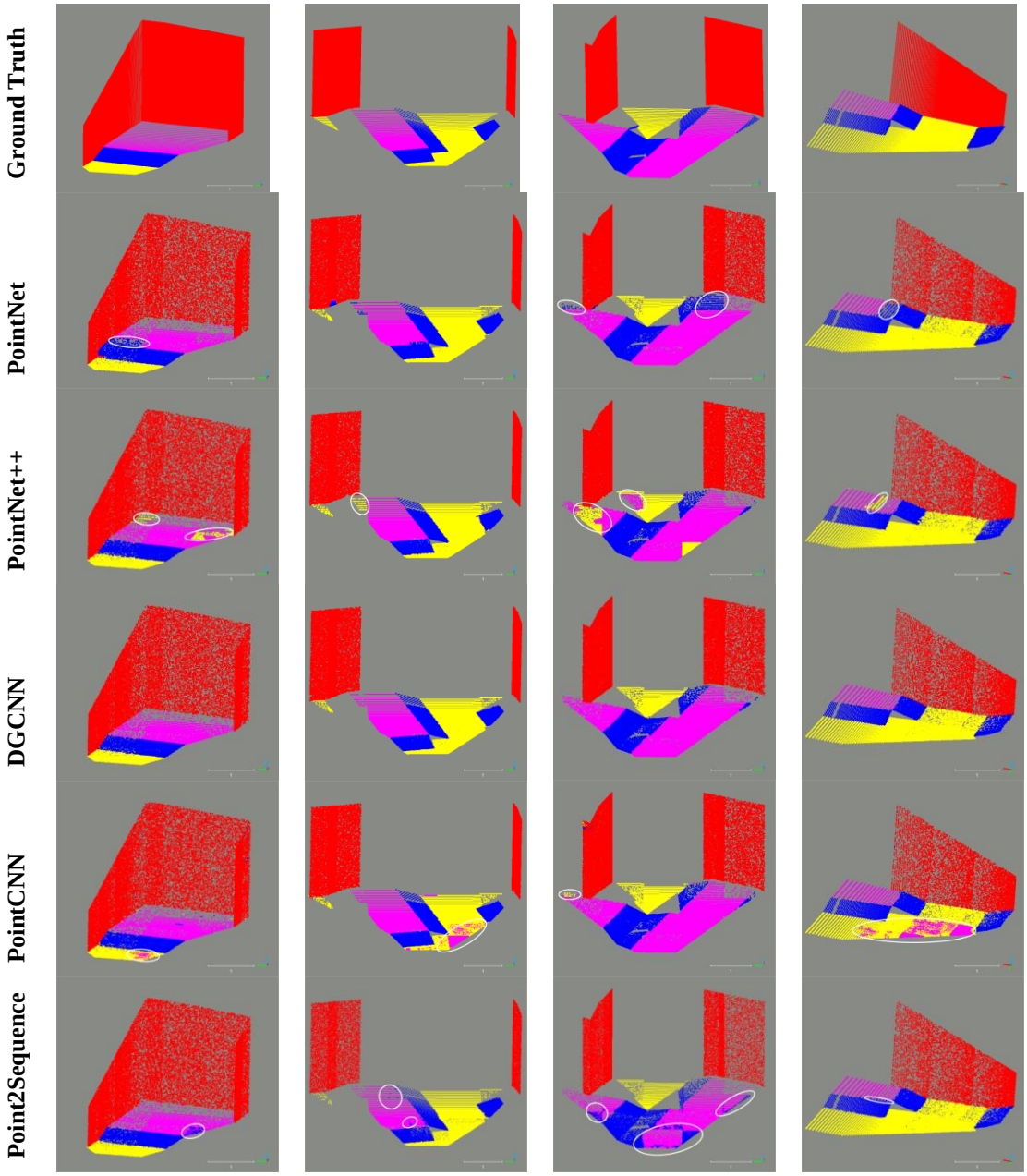
$$IoU = \frac{TP}{TP + FP + FN} \quad (3)$$

$$MIoU = \frac{\sum_{i=1}^{n} IoU_i}{n} \quad (4)$$

$$Acc = \frac{\sum_{i=1}^{n} TP_i}{\sum_{i=1}^{n} (TP_i + FP_i)} \quad (5)$$

4.2. Experimental Results

The metric results obtained on the test dataset are given in Table 1. Besides, four example scenes are selected to examine the positive and negative aspects of architectures, and they are shown in Figure 3. In the figure, rows describe the ground truth and architectures, and columns are depicted selected example scenes. Besides, white ellipses are used to emphasize the incorrectly classified regions of architectures. It has been observed that for all classes, the DGCNN architecture produces results over 99% in all metrics, and it is the most successful among architectures. In contrast to other architectures, DGCNN does not expand local regions. Besides, it utilizes x, y, and z coordinates in the feature extraction process in the first layer while it considers feature space in the successive layers. These facts are the main reasons for the success of DGCNN.



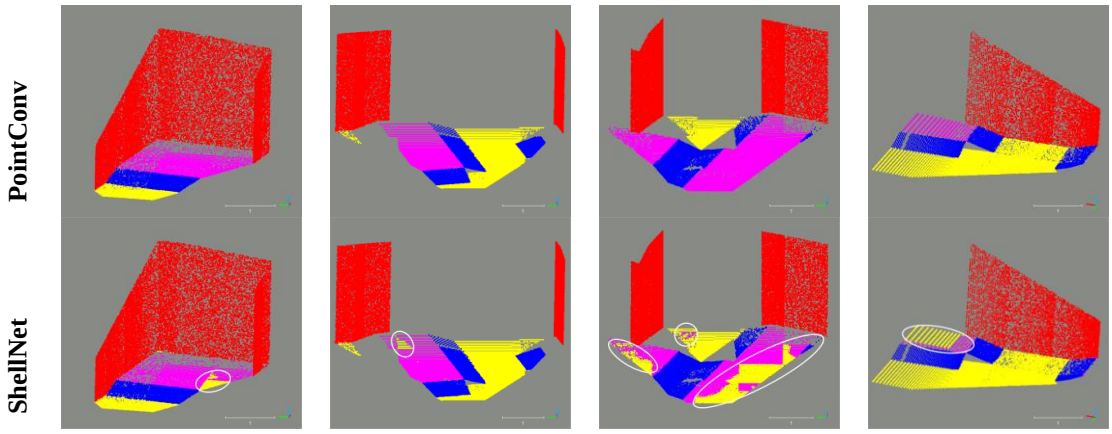


Figure 3. Visual Results

Table 1. Metric Results

	Inclined Ramp			Wall			Flat Ramp			Terrain				
	R	***P***	***IoU***	***R***	***P***	***IoU***	***R***	***P***	***IoU***	***R***	***P***	***IoU***	**MIoU**	**Acc**
PointNet	98.3	96.1	94.6	99.1	99.9	99.0	96.8	98.1	95.0	**99.9**	99.8	**99.8**	97.1	98.9
PointNet++	99.5	98.9	98.5	99.9	99.9	99.8	89.0	93.4	83.8	95.9	93.6	90.0	93.0	96.8
DGCNN	**99.8**	99.7	**99.6**	99.9	99.9	**99.9**	99.7	**99.8**	**99.6**	**99.9**	**99.9**	**99.8**	**99.8**	**99.9**
PointCNN	99.7	99.4	99.2	99.5	99.9	99.4	99.6	64.6	64.4	67.0	99.4	66.8	82.5	90.4
Point2Sequence	98.7	85.9	84.9	99.8	**100.0**	99.8	81.1	**99.8**	80.9	99.5	99.0	98.5	91.0	96.3
PointConv	97.9	**99.9**	97.8	**100.0**	99.7	99.7	**99.9**	98.3	98.2	**99.9**	99.7	99.7	98.8	99.5
ShellNet	99.7	**99.9**	**99.6**	**100.0**	**100.0**	**99.9**	71.2	93.5	67.9	97.1	84.7	82.7	87.5	94.2

PointConv also yielded successful results similar to DGCNN. PointConv learns the convolution kernel weight through weight-shared MLP over the relative position of points in the local neighborhood. Then, it re-weight the convolution kernel according to the density of points. In this way, PointConv understands the general characteristic of a scene while avoiding suppressed features that correspond to dense local regions.

PointNet assesses each point individually and also independently from other points. Therefore, it may produce erroneous results where the regions in which the points that belong to different classes are neighbors to each other. Examples for these erroneous regions are given in Figure 3. It is seen that wall and terrain classes are recognized successfully when the metric results are examined. However, PointNet may confuse flat and inclined ramp classes since they are generally placed together in scenes.

PointNet++ classifies wall and inclined ramp classes with over 98% in all metrics. However, in some cases, it may not distinguish between terrain and flat ramp classes, as shown in Figure 3. PointNet++ exploits local information when it extracts point features similar to DGCNN. However, it expands the local regions in successive layers. This may cause confusing terrain and flat ramp classes since PointNet++ generally considers exact feature of points while the DGCNN utilizes the difference feature (in other words edges) of points.

Point2Sequence, similar to PointNet, can identify wall and terrain classes. Point2Sequence constructs more than one local region with different radius values around a center point. Then, the features for these multi-scale concentric local regions are aggregated with the LSTM mechanism. Point2Sequence may yield erroneous results for the flat and inclined ramp classes. The reason for that, these ramps are generally situated together, and the

features for different radius can describe these different classes. As a result, the LSTM mechanism may not consolidate a feature vector to distinguish these classes. Also, because of attention approaches, descriptive features may be suppressed.

ShellNet behaves similar to PointNet++, which means that it can classify wall and inclined ramp, but it may confuse between terrain and flat ramp classes. The main difference between ShellNet and PointNet++ is that ShellNet applies maximum pooling and then 1D convolution to the shells' points, while PointNet++ directly employed maximum to all points in the local regions. In this way, ShellNet suppresses the dominant features, making it challenging to distinguish terrain and flat ramp classes.

In contrast to other architectures, except PointConv, that consider points individually while extracting features, PointCNN calculates features by applying discrete convolution to a point and its neighbors. On the other hand, PointConv employs continuous convolution, and it does not order the points canonically. In this way, it preserves the general

characteristic of the scene. However, PointCNN confuses terrain and flat ramp classes since it only considers the transformation matrix learned through the X-Conv operator.

V. CONCLUSION AND FUTURE WORKS

This study aimed to classify walls, ramps, and terrain in NIST reference test arenas. To achieve that, PointNet, PointNet++, DGCNN, PointCNN, Point2Sequence, PointConv, and Shellnet point-based deep learning architectures were implemented. The tests were conducted using ESOGU RAMPS dataset. The test results showed that DGCNN and PointConv architectures are capable of classifying all classes. Besides, all architectures successfully identified wall class. However, it was observed that each architecture produce erroneous results depending on its own characteristic. Thus, by revealing the positive and negative aspects of these architectures, it was aimed to create a reference point for researchers who will use them for the classification of planar surfaces in NIST test reference areas.

For future works, it is planned to classify walls, ramps, and terrain with data captured from an environment similar to NIST test reference areas in Eskisehir Osmangazi University Electrical and Electronics Engineering Artificial Intelligence and Robotics laboratory. At this point, the idea that is using the transfer learning method of the models trained with the ESOGU RAMPS dataset obtained from the Gazebo simulation environment and a small number of training data from the real environment comes to the fore.

REFERENCES

[1] Kitano, H. and Tadokoro, S. (2001). RoboCup Rescue: A Grand Challenge for Multiagent and Intelligent Systems. *AI Magazine.*, 22(1), 39-52.

[2] Jacoff, A., Messina, E., Weiss, B. A., Tadokoro, S., & Nakagawa, Y. (2003, October). Test arenas and performance metrics for urban search and rescue robots. *In Proceedings 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2003)* (pp. 3396-3403).

[3] Amigoni, F., Visser, A., & Tsushima, M. (2012, June). Robocup 2012 rescue simulation league winners. *In Robot Soccer World Cup* (pp. 20-35). Springer, Berlin, Heidelberg.

[4] Sheh, R., Schwertfeger, S., & Visser, A. (2016). 16 years of robocup rescue. *KI-Künstliche Intelligenz.*, 30(3), 267-277.

[5] Robot Operating System (ROS), Open source robotics foundations (OSRF), https://www.ros.org/, (March,2021).

[6] Grisetti, G., Stachniss, C., & Burgard, W. (2007). Improved techniques for grid mapping with rao-blackwellized particle filters. *IEEE transactions on Robotics.*, 23(1), 34-46.

[7] Kohlbrecher, S., Von Stryk, O., Meyer, J., & Klingauf, U. (2011, November). A flexible and scalable SLAM system with full 3D motion estimation. In 2011 *IEEE international symposium on safety, security, and rescue robotics* (pp. 155-160).

[8] Hornung, A., Wurm, K. M., Bennewitz, M., Stachniss, C., & Burgard, W. (2013). OctoMap: An efficient probabilistic 3D mapping framework based on octrees. *Autonomous robots*., 34(3), 189-206.

[9] Labbé, M., & Michaud, F. (2011, September). Memory management for real-time appearance-based loop closure detection. *In 2011 IEEE/RSJ international conference on intelligent robots and systems* (pp. 1271-1276).

[10] Example reference test arena, Rescue Robot League https://en.wikipedia.org/wiki/Rescue_Robot_League, (March,2021).

[11] Nguyen, A., & Le, B. (2013, November). 3D point cloud segmentation: A survey. *In 2013 6th IEEE conference on robotics, automation and mechatronics (RAM)* (pp. 225-230).

[12] Grilli, E., Menna, F., & Remondino, F. (2017). A review of point clouds segmentation and classification algorithms. *The International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences.*, XLII-2/W3, 339-344.

[13] Xie, Y., Tian, J., & Zhu, X. X. (2020). Linking points with labels in 3D: A review of point cloud semantic segmentation. *IEEE Geoscience and Remote Sensing Magazine*., 8(4), 38-59.

[14] Kaleci, B., & Turgut, K. (2019, October). Plane Segmentation of Point Cloud Data Using Split and Merge Based Method. *In 2019 3rd International Symposium on Multidisciplinary Studies and Innovative Technologies (ISMSIT)* (pp. 1-7).

[15] Rabbani, T., Van Den Heuvel, F., & Vosselmann, G. (2006). Segmentation of point clouds using smoothness constraint. *International archives of photogrammetry, remote sensing and spatial information sciences*., 36(5), 248-253.

[16] Fischler, M. A., & Bolles, R. C. (1981). Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*., 24(6), 381-395.

[17] Ballard, D. H. (1981). Generalizing the Hough transform to detect arbitrary shapes. *Pattern recognition.*, 13(2), 111-122.

[18] Melzer, T. (2007). Non-parametric segmentation of ALS point clouds using mean shift. *Journal of Applied Geodesy Jag.*, 1(3), 159-170.

[19] Golovinskiy, A., & Funkhouser, T. (2009, September). Min-cut based segmentation of point clouds. *In 2009 IEEE 12th International Conference on Computer Vision Workshops, ICCV Workshops* (pp. 39-46).

[20] Rusu, R. B., & Cousins, S. (2011, May). 3d is here: Point cloud library (pcl). *In 2011 IEEE international conference on robotics and automation* (pp. 1-4).

[21] Eruyar, E. E., Yılmaz, M., Yılmaz, B., Akbulut, O., Turgut, K., & Kaleci, B. A Comparative Study for Indoor Planar Surface Segmentation via 3D Laser Point Cloud Data. *Black Sea Journal of Engineering and Science*., 3(4), 128-137.

[22] Deng, W., Huang, K., Chen, X., Zhou, Z., Shi, C., Guo, R., & Zhang, H. (2020). Semantic RGB-D SLAM for Rescue Robot Navigation. *IEEE Access.*, 8, 221320-221329.

[23] Zhang, J., Zhao, X., Chen, Z., & Lu, Z. (2019). A review of deep learning-based semantic segmentation for point cloud. *IEEE Access*., 7, 179118-179133.

[24] Guo, Y., Wang, H., Hu, Q., Liu, H., Liu, L., & Bennamoun, M. (2020). Deep learning for 3d point clouds: A survey. *IEEE transactions on pattern analysis and machine intelligence*.

[25] Qi, C. R., Su, H., Mo, K., & Guibas, L. J. (2017). PointNet: Deep learningon point sets for 3D classification and segmentation. InCVPR.

[26] Qi, C. R., Yi, L., Su, H. & Guibas, L. J. (2017). PointNet++: Deephierarchical feature learning on point sets in a metric space. InNeurIPS.

[27] Zhang, Z., Hua, B. S. & Yeung, S. K.(2019). ShellNet: Efficient Point Cloud Convolutional Neural Networks using Concentric Shells Statistics. International Conference on Computer Vision (ICCV).

[28] Zhao, H., Jiang, L., Fu,, C.-W. & Jia, J. (2019). PointWeb: Enhancing lo-cal neighborhood features for point cloud processing, InCVPR.

[29] Thomas, H., Qi, C. R., Deschaud,, J.-E., Marcotegui, B., Goulette, F., & Guibas, L. J. (2019). KPConv: Flexible and deformable convolutionfor point clouds. InICCV.

[30] Wu, W., Qi, Z., & Fuxin, L. (2019). PointConv: Deep convolutionalnetworks on 3D point clouds. InCVPR.

[31] Li, Y., Bu, R., Sun, M., Wu, W., Di, X. & Chen, B. (2018). PointCNN:Convolution on x-transformed points. InNeurIPS.

[32] Landrieu , L. & Simonovsky, M. (2018). Large-scale point cloud seman-tic segmentation with superpoint graphs. InCVPR.

[33] Wang, Y., Sun, Y., Liu, Z., Sarma, S. E., Bronstein, M. M., & Solomon, J. M. (2019). Dynamic graph CNN for learning on point clouds. `ACM Transactions on Graphics.`

[34] Ye, X., Li, J., Huang, H., Du, L., & Zhang, X. (2018). 3D recurrentneural networks with context fusion for point cloud semanticsegmentation. InECCV.

[35] Liu, X., Han, Z., Liu , Y. S., & Zwicker, M. (2019). Point2Sequence:Learning the shape representation of 3D point clouds with anattention-based sequence to sequence network. InAAAI.

[36] Huang, Q., Wang, W., & Neumann, U. (2018). Recurrent slice networksfor 3D segmentation of point clouds. InCVPR.

[37] Turgut, K., & Kaleci, B. (2019). A PointNet Application for Semantic Classification of Ramps in Search and Rescue Arenas. *International Journal of Intelligent Systems and Applications in Engineering.*, 7(3), 159-165.

[38] Turgut, K., & Kaleci, B. (2020, October). Comparison of Deep Learning Techniques for Semantic Classification of Ramps in Search and Rescue Arenas. *In 2020 Innovations in Intelligent Systems and Applications Conference (ASYU)* (pp. 1-6).

[39] The ESOGU RAMPS dataset, ESOGU, https://ai-robotlab.ogu.edu.tr/Sayfa/Index/11, (March,2021).

[40] Gazebo, Open source robotics foundations (OSRF), https://gazebosim.org/, (March,2021).

[41] Tensor Flow Library, https://www.tensorflow.org/, (March,2021).