# SEMANTIC SEGMENTATION IN SEARCH AND RESCUE ENVIRONMENTS

**by**

**151220152069 ALPER ERGÜL**

**151220154088 MEHMET ANIL ESEN**

**151220152101 YUNUS BAŞ**

**A Graduation Project Report**

**Electrical Electronics Engineering Department**

**JUNE 2021**

# SEMANTIC SEGMENTATION IN SEARCH AND RESCUE ENVIRONMENTS

**by**

**151220152069 ALPER ERGÜL**

**151220154088 MEHMET ANIL ESEN**

**151220152101 YUNUS BAŞ**

**A Report Presented in Partial Fulfilment of the Requirements for the Degree Bachelor of Science in Electrical Electronics Engineering**

**ESKISEHIR OSMANGAZI UNIVERSITY**

**JUNE 2021**

# SEMANTIC SEGMENTATION IN SEARCH AND RESCUE ENVIRONMENTS

**by**


**151220152069 ALPER ERGÜL**

**151220154088 MEHMET ANIL ESEN**

**151220152101 YUNUS BAŞ**


**has been approved by**

**Supervisory Committee**

**Asst. Prof. Dr. Burak KALECİ**

**Asst. Prof. Dr. Helin DUTAĞACI**

**Research Assistant Kaya TURGUT**

**Prof. Dr. Gökhan ÇINAR**

# ABSTRACT

Robots must have situational awareness in search and rescue environments to perform their duties. Some methods have been developed for robots to have this awareness and to recognize their environments. One of these methods is semantic segmentation with deep learning. There are different methods and different types. Semantic segmentation has surpassed many other techniques among these methods. It is critical to adopt these works for search and rescue operations after disasters happen to reduce the risks for human and animals. To achieve this, the point-based deep learning architectures were considered. MS + CU, Tangent Convolution, RSNet, PointGrid, and PointWeb architectures were implemented for the ESOGU RAMPS dataset. The dataset contains point clouds that captured from a simulated environment; they include points belong to inclined and straight ramps, walls, and terrain. Then, the performance of these architectures was examined with the aid of metric and visual results. Besides, advantages and disadvantages of these architectures for semantic segmentation of search and rescue environments were discussed.

**Keywords:** *point cloud, deep learning, search and rescue, semantic segmentation.*

# ÖZET

Robotların görevlerini yerine getirebilmeleri için arama kurtarma ortamlarında durumsal farkındalığa sahip olmaları gerekmektedir. Robotların bu farkındalığa sahip olmaları ve çevrelerini tanımaları için bazı yöntemler geliştirilmiştir. Bu yöntemlerden biri de derin öğrenme ile anlamsal bölütlemedir. Farklı yöntemler ve farklı türleri vardır. Semantik bölütleme, bu yöntemler arasında diğer birçok tekniği geride bırakmıştır. İnsan ve hayvanlar için riskleri azaltmak için bu çalışmaların afetler meydana geldikten sonra arama kurtarma operasyonlarında benimsenmesi kritik önem taşımaktadır. Bunu başarmak için nokta tabanlı derin öğrenme mimarileri düşünülmüştür. ESOGU RAMPS veri seti için MS + CU, Tangent Convolution, RSNet, PointGrid ve PointWeb mimarileri uygulandı. Nokta bulutu, simüle edilmiş bir ortamdan yakalanan noktaları içerir; eğimli ve düz rampalara, duvarlara ve araziye ait noktaları içerir. Daha sonra bu mimarilerin performansı metrik ve görsel sonuçlar yardımıyla incelenmiştir. Ayrıca bu mimarilerin arama kurtarma ortamlarının anlamsal bölütleme için avantaj ve dezavantajları tartışılmıştır.



**Anahtar Kelimeler:** *nokta bulutu, derin öğrenme, robot, anlamsal bölütleme.*

# ACKNOWLEDGEMENT

We would like to thank our supervisor Asst. Prof. Burak Kaleci, for his useful comments and on-going support. We are also very grateful Research Assistant Kaya Turgut, for his guidance on how to solve problems about run codes and for his comments on several other aspects of the project; and to Ali Fuat Çalık, for his suggestion regarding to solving errors and comments on early drafts of our thesis.



Ali Fuat ÇALIK

Computer Engineer

Profession: Computer Engineering Student,

Organization: Eskisehir Osmangazi University Computer Engineering Faculty, Eskisehir

# TABLE OF CONTENTS

# LIST OF FIGURES

# LIST OF TABLES

# LIST OF SYMBOLS AND ABBREVIATIONS

| **Abbreviation** | **Explanation** |
|---|---|
| RSNet: | Recurrent Slice Networks |
| MS+CU: | Multi Scale Consolidation Unit |
| Acc: | Accuracy |
| R: | Recall |
| P: | Prediction |
| IoU: | Interception Over Union |
| MIoU: | Mean Interception Over Union |
| RGB: | Red Green Blue |
| LIDAR: | Light Detection and Ranging |

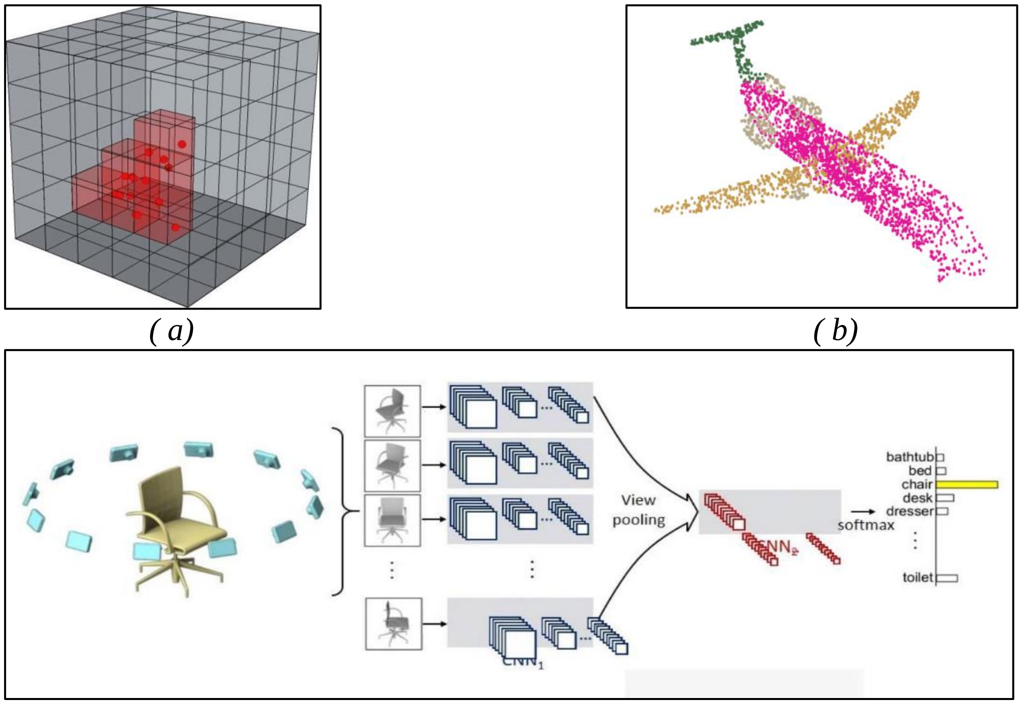
# 1. INTRODUCTION

Robots have been employed to make life easier and eliminating dangerous consequences with the ever-developing technology. Thus, the application domains where robots are used have been increased rapidly. One of these is search and rescue operations. Robots must have situational awareness in search and rescue environments to perform their duties. Some problems have been considered for robots to have this awareness and to recognize their environments. One of these is semantic segmentation. In this study, it is aimed to examine performance of point-based deep learning architectures for semantic segmentation of ramps, walls, and terrain in a search and rescue environment.

Deep learning methods were preferred in order to recognize the ramps, walls, and the terrain observed by the robot to be developed and to raise the awareness of the environment. There are different methods (Voxelization, Multiview, Point Cloud, etc. (Fig. 1)) for deep learning and semantic segmentation [1]. In order to make semantic segmentation, data's that produced from point clouds were preferred.

Semantic segmentation has surpassed many other techniques. With the different architectures created using the point cloud data, the deficiencies of the point clouds have been closed and developed day by day. Therefore, this thesis is aimed to use the cloud point cloud data in the project.

Every day, preventions are taken to protect against natural disasters. It is known how important search and rescue efforts are to prevent casualties. It is critical to develop that prevention works and to eliminate accidents and hazardous environments in search and rescue operations. For this reason, it is of great importance that the robot to be designed in the field of search and rescue is aware of the situation and the correct segmentation and techniques of ramps, walls, lands are developed.




( a )　　( b )

(c)

***Figure 1.*** *Voxelization, point cloud, multiview* [1]

For the efficient operation of the robot, we need to choose one of the most accurate point-based deep learning architecture. Therefore, first of all, training was carried out on different architectures using the original data of the architectures. The results obtained by testing at the end of the training are compared. Then, by analyzing the functioning of the code and how the data is used, arrangements are made on the code in accordance with the dataset named ESOGU RAMPS (Fig. 2). By comparing the results obtained by training different architectures with the same datasets, the appropriate architecture is selected for the semantic segmentation part of the robot.

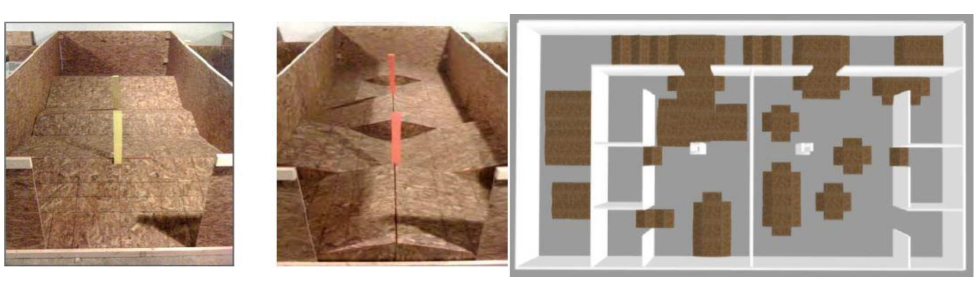


***Figure 2.*** *ESOGU's ramps*

In search and rescue operations, a robot can take much more risk than a human by moving correctly. Thus, accidents and injuries can be minimized in search and rescue operations. Human burden and chaos can be reduced in such studies. At the same time, the deep learning architecture designed and used can be improved and made more efficient. It is not only limited in this area, but it can facilitate human life by using it in different areas or it can eliminate the dangers as in this project. It is possible to encounter some difficulties during design or development.

The point cloud method that emerged to improve deep learning and semantic segmentation methods was developed with the PointNet architecture and showed itself in this field in a very open way to development. It has been observed that some of the architectures that are tried to be developed based on this architecture can be used in this project. Training and test results obtained in previous studies were compared. The architectures, which are thought to be used to increase the efficiency in terms of the project, have been taken into research and trials.

In this study, we choose to use point cloud data. Point clouds have ability to model complex objects with finite number of points. Besides, point clouds easy to produce. Point clouds commonly produced LIDAR (light detection and ranging) technology. Point clouds increase accuracy of the data and reduces errors on the data. Also, point clouds minimizes the disruption of the data. For these reasons, point clouds placed in ESOGU RAMP dataset were used (Fig. 2). We use point cloud format in ESOGU RAMP dataset as XYZ RGB L. XYZ describes the coordinates of the surface geometry. RGB is color of the surface and L is labels of the classes. Point-based deep learning architectures are selected for this experiment. Evaluate the recall, precision, accuracy and IoU values then compare these architectures each other’s.

In order to this project to work efficiently and correctly, some architecture was investigated based on PointNet [2] and PointNet ++ [3]. The methodology of the architectures below is as follows; MS + CU [4], Tangent Convolution [5], RSNet [6], PointGrid [7], and PointWeb [8].

# 2. METHODOLOGY

In this section, main characteristics of MS + CU, Tangent Convolution, RSNet, PointGrid, and PointWeb architectures are described.

## 2.1 MS+CU

MS+CU focuses on spatial context to improve Point Net architecture. It introduces two mechanisms which are input level and output level context. The first mechanism (Input level context) includes processed neighbourhood information multiple scales or adjacent regions. The second one (Output level context) operates predicted points and consolidate them over large spatial neighbourhood. Input level context increases the context that processed simultaneously. By doing this, it increases the number of blocks processed at a given time. Context are selected from same position but different scale or selected from neighbourhood cells in a regular grid. Scale dependent features are obtained from block descriptor for the multi scale approach. Output level context uses CU (Consolidation Units). In CU, block features are consolidated which are constructed in input level context. It takes a set of point features and applies MLP to these features. Thereafter, block features are concatenated again with high dimensional input features. CUs could chain together. Also, it could apply multiple times. After that, each point became informed each other.

## 2.2 Tangent Convolution

Tangent convolution architecture aims efficiently evaluating the large scale, unstructured and noisy real-world data. Tangent convolution works on the surface geometry of point cloud. Points that are orthogonal to the virtual cameras of image plane are named as tangent image. In order to evaluate tangent image, local covariance is made for each point. Image signals are estimated by using the point signals. Points that are selected spherical from point cloud project to the tangent image. Thus, image signals are constructed. This process is named as tangent convolution. Architecture derived from tangent convolution. Purpose of tangent convolution is producing feature map using by input feature map with set of weight. In architecture tangent images described as discreate function. Convolution kernel was applied to the discreate kernel tangent images. The function (g(u)) which returns the point that projectile to nearest neighbourhood of the image plane depends on only point cloud geometry. This dependency gives opportunity of precomputation. Before starting the precomputation process, N x L dimensioned matrix I was constructed. Indexes of this matrix are output of the g(u) function. This output is called as tensor M, dimension of the tensor M is N x L x Cin. Tensor M is flattened to the set of kernels W that gives output feature map. To construct convolution network for point clouds, extra functions are needed. Pooling layer one of these functions. Networks generally aggregate signals with pooling. In this architecture, pooling layers are getting harsher when the pooling is applied. Pooling starts with the 5cm, after each successful pooling layer, layer grid size is increasing. This increasement solves the non-uniform point density problem. Input feature map (Fin) that size is C x Ni is processed by grid hashing and obtained I matrix. I matrix contains Nout x 8 point that hashed to same grid. Resolution is decreased for each dimension of each pooling layer. To construct output feature map (Fout), tensor is pooled. The second extra function is un-pooling. Un-pooling process is vice versa of the pooling process. It produces high resolution feature map from low resolution feature map. In the tangent convolution, scalar function for each point in point cloud holds features about color, intensity, abstract ConvNet features. However, distance features need extra arrangements. Distance feature is evaluated by tangent plane between neighbour point. To arrange distance feature, each distance image is precomputed then concatenate them with tensor M.

### 2.3 RSNet

RSNet takes raw point clouds as input and performs 3D segmentation operations by extracting semantic labels for each point. RSNet has two blocks which extract independent feature. One of them is input feature block and the other is output feature block. The core part of RSNet is located between these two blocks. This part is the local dependency module, which is the result of the combination of a slice pooling and un-pooling layers and RNN layers First, the input feature block consumes every point in the point cloud and then extracts features based on the points it consumes. These point features are not sequential. With the slice pool layer, sequential feature vectors are saved as an sequence. Then RNNs are applied. Finally, the slice un-pooling layer does the opposite of the operations and extracts each point with its new features.

## 2.4 PointGrid

Local geometry is a structure with a 3D convolution mesh that can better display shape details, allowing learning of high-level local approximation functions. It contains a constant number of points in each grid cell so that the PointGrid can provide them. PointGrid is analyzed in three main parts. First comes the input layer, then the classification network, and finally the segmentation network. In the input layer, the coordinates of the points are stacked as features for each 3D grid cell. An efficient sampling strategy called Point Quantization is used to keep a constant number of points in each cell for train and test. This strategy works as follows. If the cells have more than K points, K of these points is randomly sampled. If cells have less than K points, samples are added. If there are no points in the cells, they are filled with zeros. The value of K is obtained by dividing the number of points for train and test by the number of grid cells. With the classification network, features are extracted from the input grid. PointGrid consists of several convolution blocks. Each convolution layer contains 3 to 3 to 3 kernel, normalization and ReLU (Rectified Linear Unit). The segmentation network shares the features that come out of the classification network and decodes them to form the segmentation. With the segmentation network, K+1 tags are created for each cell in the 3d grid. K of these tags represent K points in their cells. 1 represents an additional cell level label. To get the baseline accuracy labels, the majority of the labels of the points in the cells are taken. Cells without points, so cells which filled with zero, are labelled as no label. Likewise, points in these no label cells are labelled no label. In test operations, if the cells have points equal to or less than K, the K label corresponding to each point is used. If there are more than K points, they are labelled with cell-level labels. During training, before sampling, objects are rotated to reproduce the point cloud by vibrating the position of each point. The sampling part of PointGrid also has a data augmentation function.

## 2.5 PointWeb

PointWeb architecture combines global and local features of points within each region, built on the PointNet++ architecture. As a hierarchical architecture, PointNet++ includes global and local features to identify point clouds in each local region. Unlike PointNet++, PointWeb extracts and combines the labels of the region with the point networks it creates between the points. Another feature used for the development of the PointWeb architecture is the Adaptive Feature Adjustment (AFA) module. With this module, point networks are collected and the effect of any point on other points is observed. With this module, the performance of recognizing the local area increases with the points taken. Based on the principle of effectively extracting the local area by creating a network with dense points, "PointWeb" is formed.

# 3. ESOGU RAMP DATASET

ESOGU Dataset consists of 681 scenes. 581 of these scenes are used for the train process. The remaining 100 are used for testing. These scenes are all point clouds in .txt format. Each scene contains 4 different classes: wall, terrain, inclined ramp and flat ramp. Point clouds contain the point coordinates, the RGB codes of the classes in the scene for each point, and the label of each point. In figure 3, examples for the RGB values are given.

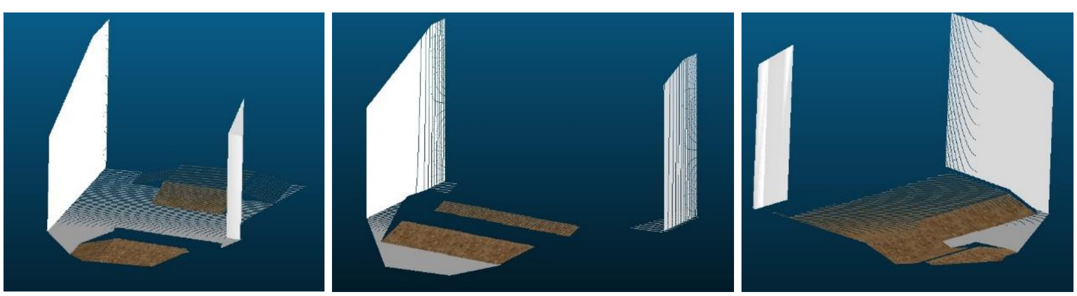


***Figure 3.*** *ESOGU ramps point cloud visuals*

# 4. EXPERIMENTS

## 4.1 Experimental Setup

In this research MS + CU, Tangent Convolution, RSNet, PointGrid, and PointWeb architectures were implemented by using TensorFlow [9] and Pytorch [10] modules in python programming language for semantic segmentation of ESOGU Ramp dataset. ESOGU Ramp dataset was integrated to the architectures and brought into the appropriate format for each architecture. Data's that divided blocks were feed into the architecture. Number of points in each block were selected as 4096 points for each architecture. To train ESOGU Ramp dataset, hyperparameters were regulated for each architecture. In MS+CU, down sampling prefix was selected as 0.03. Exponential decay Adam optimizer was used for reducing loss and regulating weight and learning rate**.** Chebyshev metric algorithm was used to evaluation of distance parameter. Epoch number is selected as 100. In Tangent Convolution, Maximum iteration number was selected as 5000 and to make data processing OPEN3D [11] was implemented. In RSNet architecture, Pytorch module was used for semantic segmentation. 100 epochs and batch size 8 were selected for train. The train process, which took too long due to insufficient equipment, was left unfinished at the end of 21 epochs. In PointGrid architecture, mat files were used for data processing. 100 epochs, 32 batch sizes were used for the train. In PointWeb architecture, Pytorch module was used for semantic segmentation. Epoch number is selected 100 and batch size is selected 8 for train and test.

## 4.2 Experimental Results

The metric and visual results were obtained from ESOGU Ramp dataset are shown in Table 1, and Figure 4, respectively. Four scenes were selected to analyze positive and negative sides of the architectures. R metric in the Table 1 represents the recall parameter. Recall value gives answer for how much prediction was done correctly for that class. P metric in the Table 1 represents the precision parameter. Precision parameter gives answer for how much prediction was done correctly from predicted class. IoU metric in the Table 1 represents the intersection over union parameter. IoU parameter gives us intersection rate which how much intersection was done with targeted class field. Acc metric in the Table 1 represents the accuracy value. Accuracy is the rate of the true predicted to total prediction.

***Table 1.*** *Metric results*

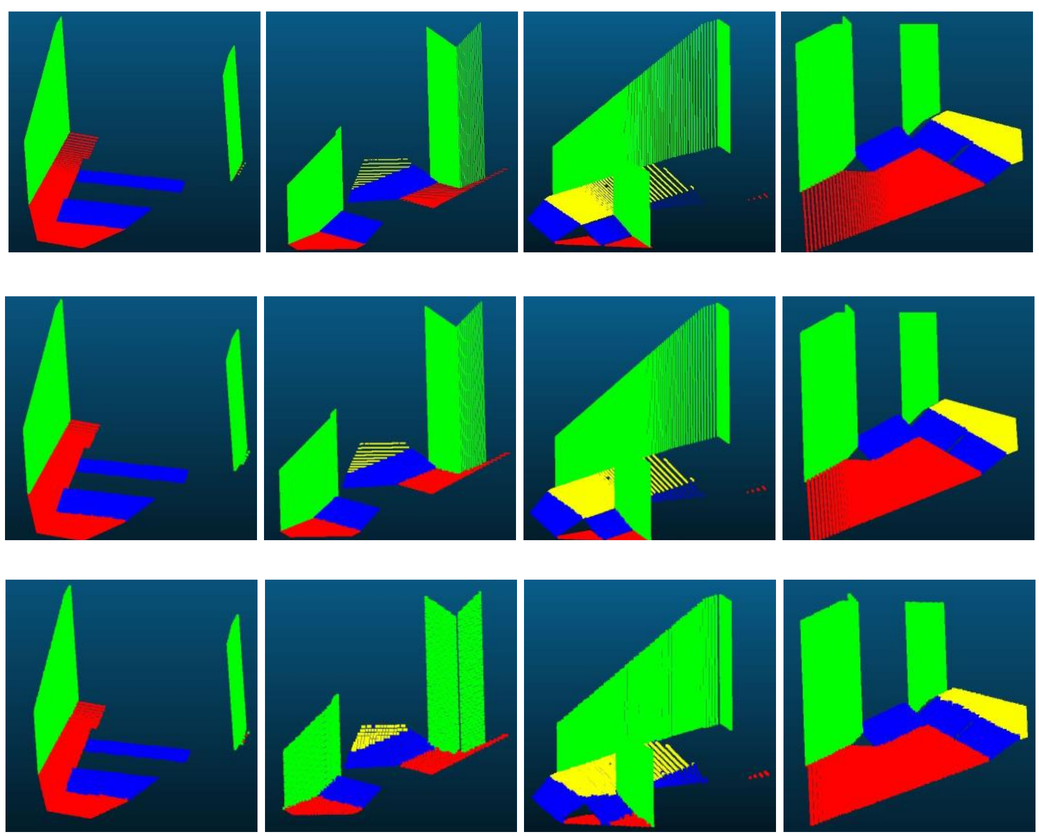
| | Inclined Ramp | | | Wall | | | Straight Ramp | | | Terrain | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | **R** | **P** | **IoU** | **R** | **P** | **IuO** | **R** | **P** | **IuO** | **R** | **P** | **IoU** | **MIoU** | **Acc** |
| MS+CU | **99.5** | **98.3** | **97.8** | **99.9** | **99.8** | **99.7** | 98.2 | **99.5** | **97.7** | 98.3 | **99.4** | 97.8 | **98.3** | **99.0** |
| Tangent Conv. | 99.2 | 91.5 | 90.9 | 99.7 | 99.6 | 99.3 | 97.9 | 99.4 | 90.2 | 99.2 | 99.3 | 97.3 | 94.4 | 96.9 |
| RSNet | 95.4 | 98.1 | 93.7 | 99.7 | 99.7 | 99.5 | **98.9** | 97.9 | 96.8 | **99.4** | 98.4 | **97.9** | 97.0 | 98.4 |
| PointGrid | 62.0 | 53.2 | 40.1 | 97.0 | 95.6 | 92.9 | 47.8 | 67.1 | 38.7 | 70.7 | 70.2 | 54.4 | 56.5 | 69.4 |

In Figure 4, colors represent the label that contain classes points, green points represent the wall class, red points represent the terrain class, blue points represent the inclined ramp class and the yellow points represent the straight ramp class.

GROUND TRUTH

MS + CU

TANGENT C.



RSNET

POINT GRID

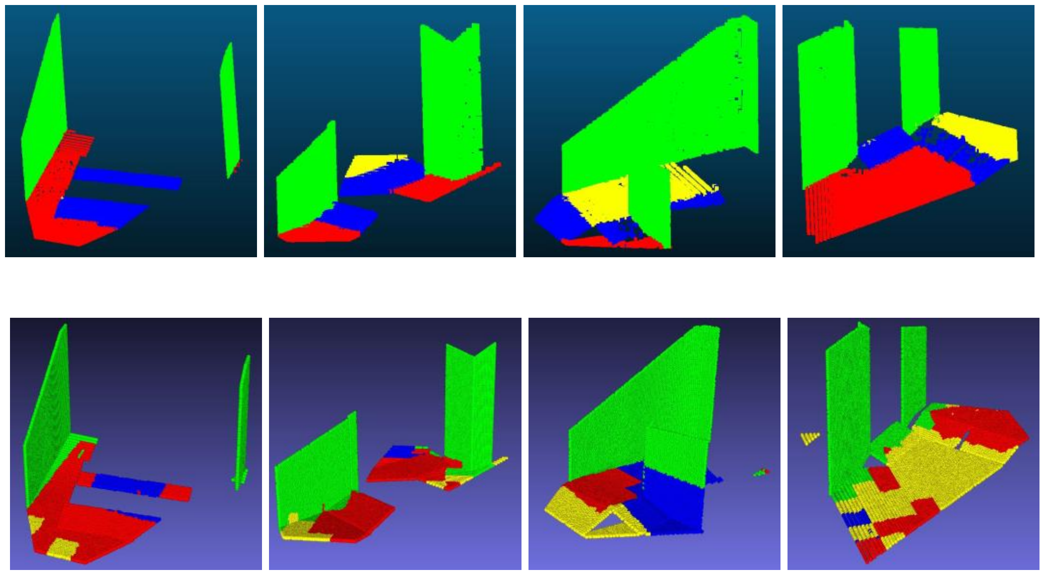


***Figure 4.*** *Visual results*

### 4.2.1 Experimental Results of MS+CU

MS+CU was done successfully test ESOGU RAMP dataset. Results were accomplished. MS+CU confuses between floor, inclined ramp, and straight ramp. Wall class was successfully predicted. However, in some scenes wall class confused between ramp and floor classes. Reason of this confused results is the group processing of the blocks in the input level context also consolidation unit makes wrong predicted points worse after each usage.

### 4.2.2 Experimental Results of Tangent Convolution

Tangent Convolution architecture test was done successfully on ESOGU RAMP dataset. The architecture was struggled to find the flat ramps. Flat ramps were confused with straight ramps. TC seems to have trouble about finding ramp and flat ramp. IoU value show us, there is trouble between ramp and flat ramp. This trouble source may be surface geometry of the junction between ramp and flat ramp. Architecture uses the surface geometry norms and convolutions to find true points. The points projected into the nearest neighbour tangent image may be mixed with the ramp or the flat ramp. This projectile process may cause that failure.

### 4.2.3 Experimental Results of RSNet

RSNet was generally successful and gave good results. RSNet classifies wall and terrain classes above 97% in all measurements. However, there are errors in some scenes where the inclined ramp and the flat ramp intersect. Likewise, in some scenes, it could not distinguish terrain and inclined ramp. The reason for these problems is the low number of epochs in the train process. Because the test measurements made at the end of each epoch showed continuous improvement. It is foreseen that these problems will be solved with a full train.

### 4.2.4 Experimental Results of PointGrid

In general, PointGrid gave unsuccessful results, as can be seen from the Table 1 and Figure 4. The reason of this is that the number of points in the point cloud is high, the grid cell and the fixed number of K points taken by each cell have not been changed, so the number of points in the cells is more than K and they are labeled directly at the cell level. Labeling at the cell level during the test phase caused many errors as can be seen from the samples. In order to eliminate these errors, a full training and testing is required by making the grid cell numbers and the number of fixed points in each cell suitable for the number of points in the point cloud. Another solution is to separate point clouds into blocks and load the data. It is estimated that the results will be much better if one of these processes is performed.

## 5.PROJECT PLAN

- **Work Package 1 – Literature Research:** Deep learning on point clouds
- **Work Package 2 – Studying Architectures:** PointWeb, MS + CU, Point Grid, Tangent Convolution and RSNet
- **Work Package 3 – Analysis of Architecture:** Learning how architectures work
- **Work Package 4 – Establishing Necessary Environments for Architectures:** Ubuntu, CUDA, Modules were established

- **Work Package 5 – Works with Original Data of Architectures:** S3DIS, ShapeNet were used
- **Work Package 6 – Analyzing the ESOGU Ramp Dataset:** Contents of ESOGU Ramp Dataset were learned
- **Work Package 7 – Integrating Architectures with ESOGU Ramp Dataset:** Integration of dataset to architectures were regulated
- **Work Package 8 – Testing and Reporting Architectures with ESOGU Ramp Dataset:** Train and test processes were done and reported
- **Work Package 9 – Literature Research:** Different architectures were searched
- **Work Package 10 – Studying Architectures:** PointWeb, MS + CU, Point Grid, Tangent Convolution and RSNet
- **Work Package 11 – Solving Problems:** Learning how architectures work
- **Work Package 12 – Establishing Necessary Environments for Architectures:** Modules were established
- **Work Package 13 – Works with Original Data of Architectures:** S3DIS, ShapeNet were used
- **Work Package 14 – Report-1:** Report about architectures was presented advisor
- **Work Package 15 – Integrating Architectures with ESOGU Ramp Dataset:** Integration of dataset to architectures were regulated
- **Work Package 16 – Report-2: :** Report about architectures was presented advisor
- **Work Package 17 – Testing Architectures with ESOGU Ramp Dataset:** Train and test processes were done
- **Work Package 18 – Solving Problems:** Some problems about training and testing were fixed

***Table 2.*** *Resource assignments for work packages*

| Work Package | Resource | Duration (weeks) |
|---|---|---|
| 1 | Alper Ergül, Mehmet Anıl Esen, Yunus Baş | 2 |
| 2 | Alper Ergül, Mehmet Anıl Esen, Yunus Baş | 1 |
| 3 | Alper Ergül, Mehmet Anıl Esen, Yunus Baş | 1 |
| 4 | Alper Ergül, Mehmet Anıl Esen, Yunus Baş | 1 |
| 5 | Alper Ergül, Mehmet Anıl Esen, Yunus Baş | 4 |
| 6 | Alper Ergül, Mehmet Anıl Esen, Yunus Baş | 1 |
| 7 | Alper Ergül, Mehmet Anıl Esen, Yunus Baş | 2 |
| 8 | Alper Ergül, Mehmet Anıl Esen, Yunus Baş | 1 |
| 9 | Alper Ergül, Mehmet Anıl Esen, Yunus Baş | 4 |
| 10 | Alper Ergül, Mehmet Anıl Esen, Yunus Baş | 1 |
| 11 | Alper Ergül, Mehmet Anıl Esen, Yunus Baş | 1 |
| 12 | Alper Ergül, Mehmet Anıl Esen, Yunus Baş | 1 |
| 13 | Alper Ergül, Mehmet Anıl Esen, Yunus Baş | 3 |
| 14 | Alper Ergül, Mehmet Anıl Esen, Yunus Baş | 1 |
| 15 | Alper Ergül, Mehmet Anıl Esen, Yunus Baş | 2 |
| 16 | Alper Ergül, Mehmet Anıl Esen, Yunus Baş | 1 |
| 17 | Alper Ergül, Mehmet Anıl Esen, Yunus Baş | 1 |
| 18 | Alper Ergül, Mehmet Anıl Esen, Yunus Baş | 2 |
| PROJECT COMPLETITION TIME | | 30 |

Table 2 shows the work package, resource due to duration table.

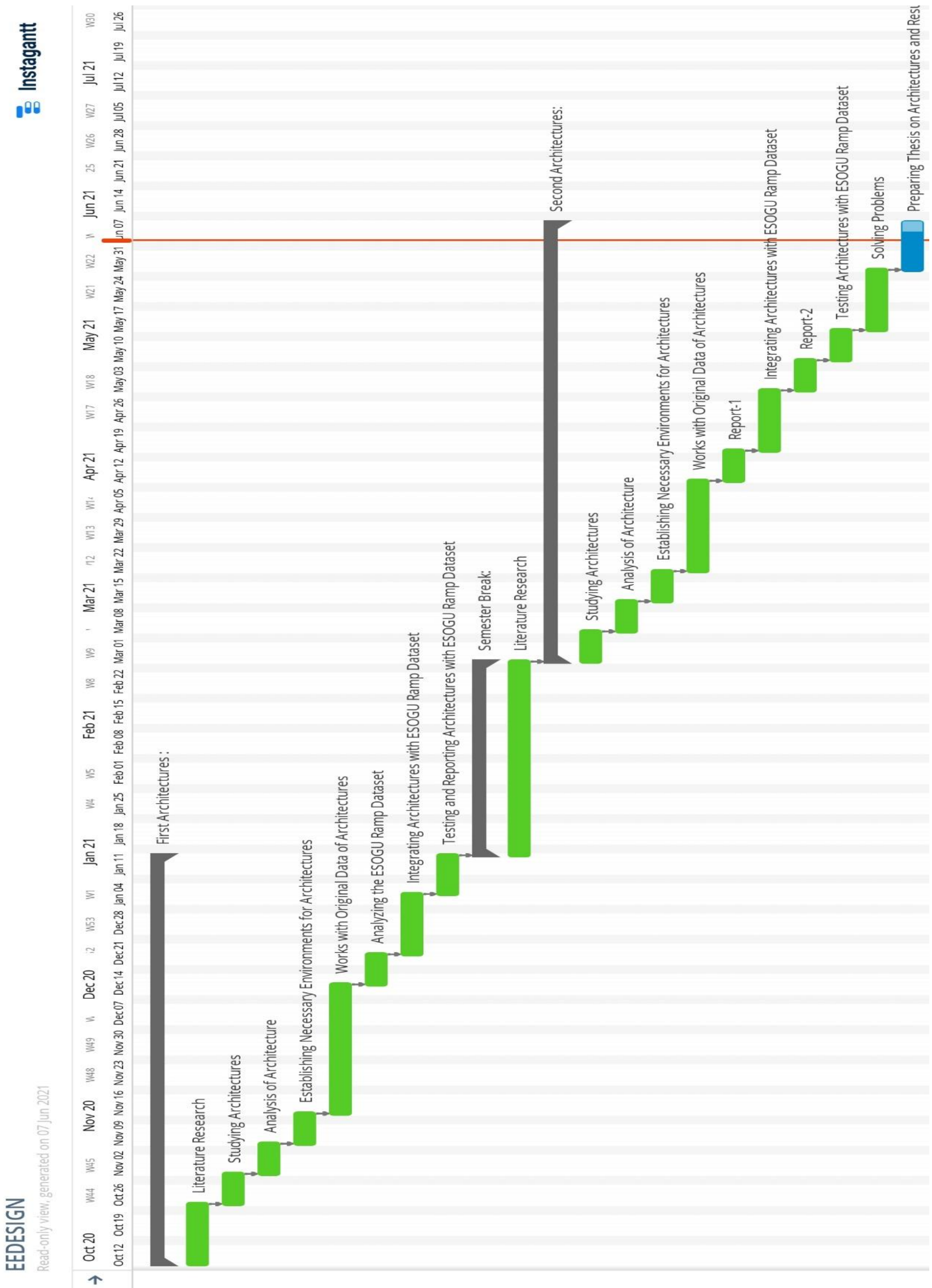



***Figure 5.*** *Gantt diagram of the project*

## 6. CONCLUSION

Purpose of this study is applying the semantic segmentation to the walls, straight ramps, terrain, and inclined ramps. MS + CU, Tangent Convolution, RSNet, PointGrid, and PointWeb architectures were run by us. Train and test processes were applied to the ESOGU Ramp dataset. We can say, MS+CU has the best test results from other architectures. The wall class was found highly correct ratio in all architectures.  All of the architectures were confused between ramp and flat ramp classes. It has been seen that RSNet architecture can give high precision results if enough training is done. As a result of the examination, it was seen that the most efficient architecture on the ESOGU Ramp dataset was MS+CU.

## REFERENCES


[1] J. Zhang, X. Zhao, Z. Chen and Z. Lu, "A Review of Deep Learning-Based Semantic Segmentation for Point Cloud," in IEEE Access, vol. 7, pp. 179118-179133, 2019, doi: 10.1109/ACCESS.2019.2958671.

[2] R. Q. Charles, H. Su, M. Kaichun and L. J. Guibas, "PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation," 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017, pp. 77-85, doi: 10.1109/CVPR.2017.16.

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

[4] Engelmann, F., Kontogianni, T., Hermans, A., & Leibe, B. (1970, January 1). Exploring Spatial Context for 3D Semantic Segmentation of Point Clouds. https://openaccess.thecvf.com/content_ICCV_2017_workshops/w13/html/Engelmann_Exploring_Spatial_Context_ICCV_2017_paper.html.

[5] Tatarchenko, M., Park, J., Koltun, V., & Zhou, Q.-Y. (1970, January 1). Tangent Convolutions for Dense Prediction in 3D. html/Tatarchenko_Tangent_Convolutions_for_CVPR_2018_paper.html

[6] Huang, Q., Wang, W., & Neumann, U. (2018, March 29). Recurrent Slice Networks for 3D Segmentation of Point Clouds. arXiv.org. https://arxiv.org/abs/1802.04402.

[7] Le, T., & Duan, Y. (1970, January 1). PointGrid: A Deep Network for 3D Shape Understanding. https://openaccess.thecvf.com/content_cvpr_2018/html/Le_PointGrid_A_Deep_CVPR_2018_paper.html.

[8] Zhao, H., Jiang, L., Fu, C. and Jia, J., 2021. PointWeb: Enhancing Local Neighborhood Features for Point Cloud Processing. Available at: https://openaccess.thecvf.com/content_CVPR_2019/html/Zhao_PointWeb_Enhancing_Local_Neighborhood_Features_for_Point_Cloud_Processing_CVPR_2019_paper.html

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

[10] Pytorch Library, https://pytorch.org/ ,(June,2021).

[11] OPEN3D Library http://www.open3d.org/, (June,2021).