AUTONOMOUS ROCK SEGMENTATION FROM LIDAR POINT CLOUDS USING MACHINE LEARNING APPROACHES

Lauren Flanagan[1,2], Levin Gerdes[2,3], Martin Azkarate[2], and Kenneth McIssac[1]

[1]*Department of Electrical and Computer Engineering, University of Western Ontario, Canada*
[2]*ESA-ESTEC, European Space Agency, Keplerlaan 1, 2201 AZ Noordwijk, The Netherlands, Email: firstname.lastname@esa.int*
[3]*Department of Systems Engineering and Automation, University of Málaga, Málaga, Spain*

ABSTRACT

This work demonstrates how Light Detection and Ranging (Lidar) point clouds can be used to autonomously and efficiently segment planetary terrain to identify obstacles for safe rover navigation. Two Lidar datasets which represent planetary environments were used to train a neural network to perform semantic segmentation. The neural network was based on the RandLA-Net architecture that was designed to efficiently perform semantic segmentation on point clouds using a random sampling algorithm without modifying the point cloud structure. Due to the nature of the planetary scenes recorded in the point clouds,the majority of the points represent the ground and the minority of the points represent obstacles. Methods to handle the class imbalance of the datasets were explored to enable the model to learn the minority class and to optimize the model's performance.

1. INTRODUCTION

The motivation of this work is to determine if point cloud data from a Light Detection and Ranging (Lidar) sensor can be used to efficiently segment terrain for safe rover navigation. The goal is to use datasets that represent a planetary environment to identify rock obstacles by performing semantic segmentation of lidar point clouds using machine learning techniques. Due to communication delay, rover operators on Earth cannot operate rovers on other planetary surfaces in real time. For example, the communication delay between Earth and Mars ranges from five to twenty minutes depending on the distance between the planets based on their positions in orbit [2]. Because of the communication delay, the current method of navigating planetary rovers is to use a blind drive method, where goal positions are set by the operators on Earth, and the rover navigates itself. However, this method is not sustainable for future missions with more complex requirements such as increased driving speeds and distances, navigating in permanently shadowed regions, or for groups of rovers working together. Overall, rovers must be capable of detecting natural obstacles including rocks, craters, cliffs, trenches, and sand dunes and navigating through the obstacles on their own.

While fully autonomous navigation approaches exist, they typically display a high computational complexity which requires planetary rovers to periodically stop for processing. The more complex the environmental analysis (for example for localization, mapping, and obstacle detection), the longer and more frequent these stops need to be, reducing navigational efficiency. The type of sensors that provide the underlying data and influences the required computational complexity.

A current method for obstacle detection on planetary surfaces is to use depth cameras to provide information about near by hazards such as large rocks, trenches, or sand dunes, and information about far away obstacles to detect a safe path for blind drives [1]. Problems can arise when using camera images due to the varying lighting conditions: large shadows can be cast by obstacles making it difficult to apply image processing techniques, lens flares can occur when there is a bright light source, and driving at night or in permanently shadowed regions is not possible without an additional light source. Lidar, on the other hand, is unaffected by lighting conditions since the sensor works by sending out pulsed lasers to measure the distance of surrounding surfaces.

Previously, using point cloud data required pre-processing techniques that use a large amount of memory and processing power. When working with planetary rovers, both the memory and processing power are limited, making it unrealistic to use Lidar on a rover. However, with recent advancements in machine learning techniques, point clouds can be fed into a Neural Network (NN) without any pre-processing steps beforehand, reducing the required computational power [5]. These advancements make it feasible to use Lidar with the limited computational resources on board a rover. The ability to autonomously detect rock obstacles using Lidar would allow for safe rover navigation, remove obstacle detection challenges created by lighting conditions, and produce a detailed map of the rover's surroundings.

The sections hereafter describe the datasets, explain the method used for rock detection, and present the segmentation results.

2. DATASETS

This section contains information about two datasets containing Lidar point clouds that were recorded at planetary analogue sites. The processes of data collection and determining ground truth labels for each data set are outlined. It is important for the data to be labelled as either rock or ground in order to use it to train machine learning models to detect rock obstacles in planetary scenes.

2.1. Analogue Terrain Facility Dataset

2.1.1. Data Collection

Lidar point cloud data was collected at the Canadian Space Agency's Analogue Terrain Facility (ATF) in Saint-Hubert, Quebec. The ATF is approximately 80×100 meters with several different types of terrain, geometry, and rocks. Point cloud scans were collected using an Ouster Lidar OS1-64 which has a $120\,\mathrm{m}$ range and a point cloud scan rate of $20\,\mathrm{Hz}$ with 65536 points per scan. The lidar was attached to a Clearpath Husky Unmanned Ground Vehicle (UGV) and data was collected continuously while driving on a path through a sparse boulder field with small and medium sized boulders, and flat sandy terrain. Overall, 3318 point cloud scans were collected during the traverse through the ATF.

2.1.2. Data Processing

Obtaining ground truth versions of the point clouds, which identify the location of the rocks is an important step to be able to use the data. One key feature of points that represent rocks are their intensity, reflectivity, and normal vector. The intensity and reflectivity of the rocks are greater than that of the surrounding terrain. Another key feature of points that represent rocks is that normal vector to the points which make up the rocks vary from the normal vectors of the terrain. A sharp change in the normal direction can be seen at the base of each rock. By combining the change in the normal direction with the change in the reflectivity and intensity values, the rocks can be separated from the terrain in the point clouds which allows for ground truth labels to be estimated and applied to the ATF dataset. Fig. 1 shows a birds-eye view of the ground truth of the point cloud with the rocks are shown in red.

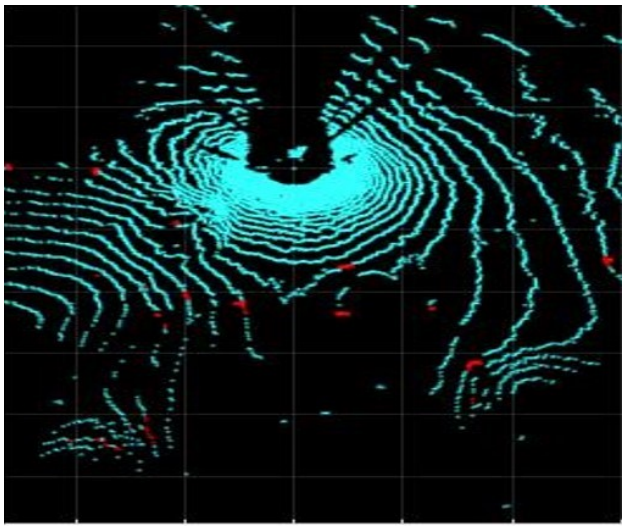


Figure 1: Birds-eye view of a point cloud from the ATF dataset showing rock points in red

2.2. Katwijk Beach Planetary Rover Dataset

2.2.1. Data Collection

The Katwijk Beach Planetary Rover Dataset was collected at a beach near Katwijk in The Netherlands [3]. Three traverses were conducted during the data collection. The first two traverses were approximately $1000\,\mathrm{m}$ through a boulder field made up of two hundred and twelve artificial rocks that were placed to model typical boulder fields seen in Mars Reconnaissance Orbiter images. The third traverse was approximately $200\,\mathrm{m}$ through a boulder field made up of the same two hundred and twelve artificial rocks as in the first traverses, but with the rocks placed to be twice as dense. In all three traverses, three different sized artificial boulders were used. The large boulder was $1.897\,\mathrm{m}$ in diameter, the medium boulder was $1.326\,\mathrm{m}$ in diameter, and the small boulder was $0.737\,\mathrm{m}$ in diameter. Overall, there were 12 large boulders, 100 medium boulders, and 100 small boulders used.

During the data collection process, the rover was equipped with a Velodyne VLP-16 Lidar sensor which has a 100 meter range and a point cloud scan rate of 20 Hz with 28928 points per scan. During the first traverse, 15500 point clouds were collected, in the second traverse 10700 point clouds were collected and in the third traverse 9200 point clouds were collected. Additionally, Differential Global Positioning System (DGPS) data was collected every three seconds for the rover position while driving, and for each rock location.

2.2.2. Data Processing

In order to determine ground truth models, the Lidar scans were aligned with the known location and estimated heading direction of the rover and overlaid on a georeferenced tiff image. Then, since the location and size of each rock was known, the points overlapping the rock locations could be labeled as rock. To achieve this ground truth model, the following steps had to be taken. First, the position of the rover when each Lidar scan was taken had to be interpolated from the known rover locations and the time stamps on the data. The rover moved approximately $1.5\,\mathrm{m}$ between each recorded rover location, so a linear interpolation of the Lidar scan locations between each rover location was found to be acceptable.

The next step in aligning the point clouds with the rover location and heading direction was to apply a series of transformations to transform the data from the local Lidar frame to the location and orientation of the Lidar mounded on the rover in Universal Transverse Mercator (UTM) coordinates taken from the recorded DGPS data. These transformations resulted in the origin of the Lidar

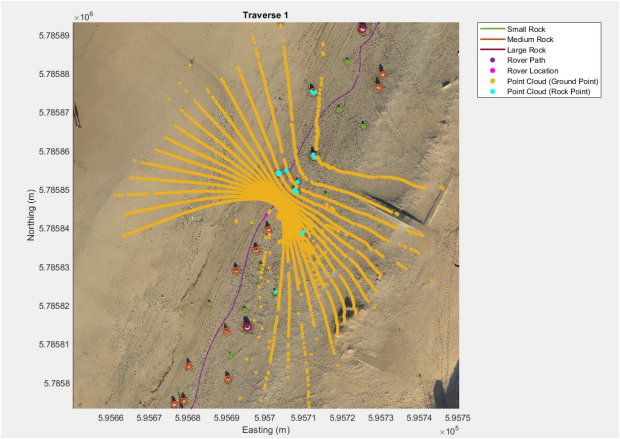


Figure 2: Birds-eye view of a point cloud from the Katwijk Beach Planetary Rover Dataset showing rock points in blue.

data being aligned with the DGPS location of the Lidar sensor as data was collected. Since the location and radius of each rock was also recorded, each point was labeled as rock or ground based on whether or not it overlapped a rock location as shown in Fig. 2.

3. METHODOLOGY

This section outlines the methodology used to perform semantic segmentation of the point clouds contained in the available datasets and the methodology to test the real-time implementation of the developed models.

3.1. Training, Validation, and Testing Split

The data was divided into three parts: the training set, validation set, and testing set. The training set is made up of 60 % of the data and the validation and testing sets are each made up of 20 % of the data. The data from each traverse is sequential, therefore, the data should be split in sequential order to reduce the number of overlapping scans in the training, validation, and testing sets. Each traverse was split so the first 60 % of a traverse was assigned to training, the next 20 % was assigned to validation, and the final 20 % was assigned to testing.

3.2. Handling Class Imbalance

Naturally, each point cloud scan in the datasets contain a large number of ground points and a small number of rock points which creates a class imbalance. To deal with the class imbalance, random under sampling of the ground class and cost-sensitive methods were explored in the experiments.

Random under sampling of the ground class is applied by determining the index of all points corresponding to the ground class, and randomly selecting a number of points to delete, N, determined by subtracting the number of points desired after random under sampling from the total number of points in the point cloud scan as shown in Eq. 1.

Table 1: Percentages of random under sampling of the ground class explored and the number of points remaining in each point cloud after random under sampling

Percentage of Random Under Sampling	20%	40%	60%
Number of Points Remaining in the Point Cloud After Sampling	23142	17557	11571

$$N = len(points_{total}) - len(RUS) \tag{1}$$

Then, N points from the ground class are randomly sampled and deleted. The percentages of randomly sampled points and the corresponding number of points remaining in the point clouds after random under sampling that are explored in this work are summarized in Tab. 1.

Two cost sensitive methods were explored: a cost matrix, and inverse cost frequency. For this work, false negative classifications are much more serious than false positive classifications. A false negative could lead to the rover hitting a rock and getting damaged, whereas a false positive will result in the rover avoiding areas that did not necessarily need to be avoided. The cost matrices explored in the experiments are displayed in Tab. 2. In both matrices the cost of a false negative is set higher than the other costs and there is no cost for a correct classification. Cost matrix A assigns a cost of ten for a false negative classification and a cost of one for a false positive classification. Cost matrix B assigns a cost of fifty for a false negative classification and a cost of one for a false positive classification. The inverse cost frequency method takes into account the ratio of the total points in each class relative to the total number of points in the dataset, as shown in Eq. 2.

$$weight = \frac{num_{class}}{num_{total}} \tag{2}$$

Then the cost of predicting a false negative and false positive are calculated as the inverse of the weight plus a scaling factor, as shown in Eq. 3. The scaling factor is used to limit maximum cost of misclassifying the minority class. The resulting costs calculated from the total class imbalance in the Katwijk Beach Planetary Rover Dataset are 29 for a false negative and 1 for a false positive.

$$cost = \frac{1}{weight + 0.02} \tag{3}$$

The cost of predicting a false negative is calculated using the weight of the positive class, in this case the rock class. The cost of predicting a false positive is calculated using the weight of the negative class, in this case the ground class.

3.2.1. Iterating the Cost Sensitive Methods

After analyzing the results of the initial grid search presented in Section 4.1, the models had a significantly

Table 2: Initial cost matrices explored in experiments

(a) Cost Matrix A

Cost Matrix A	Actual Class	
Predicted Class	0	1
	10	0

(b) Cost Matrix B

Cost Matrix B	Actual Class	
Predicted Class	0	1
	50	0

Table 3: Cost matrices explored in iteration one

(a) Cost Matrix A

Cost Matrix A	Actual Class	
Predicted Class	0	5
	30	0

(b) Cost Matrix B

Cost Matrix B	Actual Class	
Predicted Class	0	5
	50	0

(c) Cost Matrix C

Cost Matrix C	Actual Class	
Predicted Class	0	10
	30	0

(d) Cost Matrix D

Cost Matrix D	Actual Class	
Predicted Class	0	10
	50	0

higher number of false positives than false negatives. The cost sensitive methods were tuned by increasing the cost of a false positive to see if the false positive rate could be decreased while keeping the false negative rate low. The second set of cost matrices considered are given in Tab. 3.

3.3. Semantic Segmentation of the Katwijk Beach Planetary Rover Dataset

The ability to achieve a low computational complexity and accurate segmentation results is important to allow for model deployment on-board a rover. The network RandLA-Net was selected as a starting point for training a model to detect rock obstacles from point clouds in planetary scenes because the network was able to achieve results of $53.9\,\%$ Mean Intersection Over Union (mIOU) when performing semantic segmentation of the SemanticKITTI dataset, while keeping the computational complexity low: achieving an approximate frame rate of 22 frames per second on an NVIDIA RTX2080Ti GPU [4]. The original RandLA-Net model uses the pre-processing techniques of grid sub-sampling and random cropping to reduce the size of the point clouds used to train the model. The random cropping algorithm selects a point at random and keeps the K-nearest neighbours to that point. The point clouds used for this work contain approximately one third the number of points that were used to train the original RandLA-Net model and only two classes; therefore, it is not necessary to reduce the number of points in each point cloud. Further, the grid sub-sampling method and random cropping method could both lead to rock points being removed from the point clouds which should be avoided because there is already a large class imbalance in the datasets. For these reasons, the pre-processing steps of grid sub-sampling and random cropping are not applied in this work.

Table 4: Hyperparameters explored while tuning the model (* with early stopping)

Hyperparameter	**Value**
Number of Layers	2, 3, 4
Batch Size (Training)	6, 12
Learning Rate	1e-2, 1e-4
Number of Epochs	200*

A grid search of hyperparameters and methods for handling the class imbalance of the datasets was performed to find the best model for performing semantic segmentation of the planetary datasets. All of the parameter combinations were trained five times and the results were averaged to get a better idea of each model's performance because the random sampling method adds variation to each training set. Early stopping was used while training the models: if the model had not improved with respect to the mIOU within the last 25 epochs, then the training process was stopped. The tuned hyperparameters, summarized in Tab. 4 include the number of layers, the batch size, the learning rate and the number of epochs.

3.4. Semantic Segmentation on the Analogue Terrain Facility Dataset

Significant differences exist between the two datasets: each Lidar has different parameters resulting in different density representations of the scenes; the ATF dataset has $3\,\%$ the number of point clouds and covers $2\,\%$ of the distance compared to the Katwijk Beach Planetary Rover Dataset, resulting in the ATF dataset being underrepresented; and the artificial obstacles in the Katwijk Beach Planetary Rover Dataset have consistent shape, size, and texture compared to the real rocks in the ATF that vary in shape, size, and texture. For all of these reasons, the datasets were not mixed together for training. Instead, after training on the Katwijk Beach Planetary Rover Dataset, the top three performing combinations of model parameters, presented in Tab. 5, were applied to the ATF dataset to test the model's ability to learn representations of real rocks with varying size, shape, and texture which will examine how well the model performs on a more complex dataset.

3.5. Real-time Inference of Point Clouds

The ability of the model to segment point clouds in real time with limited computational power is important so

Table 5: Parameters of the models to be trained and tested on the ATF dataset

Model Number	Number of Layers	Batch Size	Learning Rate	Cost Method
1	4	6	1e-2	[0, 5; 30, 0]
2	4	6	1e-2	[0, 10; 30, 0]
3	4	12	1e-2	[0, 10; 50, 0]

the model is able to be deployed on-board a planetary rover. To test the ability to deploy the model on board a rover, the top three performing models from Experiment One were tested to be used in a real-time simulation using an Intel Xeon E5-2665 CPU. The time to perform scene segmentation on the 480 point clouds in the test set of the Katwijk Beach Planetary Rover Dataset was recorded and averaged. The results were compared with the segmentation frame rate required for implementing the model on board a rover, and the frame rate which the Lidar used to create the dataset publishes point clouds.

4. RESULTS AND DISCUSSION

This section presents the results of the experiments outlined in section 3 and discusses the implications of the results.

4.1. Semantic Segmentation on the Katwijk Beach Planetary Rover Dataset

The first step consisted of tuning the number of layers while holding the other hyperparameters constant. The model was trained with four, three, and two layers and the average validation performance metrics are given in Tab. 6. Since there is a large class imbalance, the model naturally achieves very high accuracy by correctly predicting the ground class, so the recall and precision scores are used to measure the model's ability to predict the minority class. The recall score gives the ratio of correctly predicted rock class labels to the true total number of rock class labels. The recall scores were similar between each model, with a decrease of only $0.62\,\%$ as the number of layers decreased from four to two. The precision score gives the ratio of correctly predicted rock class labels to the total number of predicted rock class labels. The precision scores decrease by $8.24\,\%$ as the number of layers were reduced from four to two layers, and indicates that number of layers has a greater impact on the number of false positive predictions. Since the model performance was found to decrease as the number of layers decreased, the remainder of the experiments are performed on models with four layers only.

A grid search was performed on the remaining hyperparameters and class imbalance techniques outlined in section 3.3. While performing the grid search with $20\,\%$ random under sampling applied, the model was either not able to learn, or it would overfit to the training set. Random under sampling likely did not perform well because

Table 6: Comparison of average performance metrics of models trained with two, three, and four layers

Number of Layers	4	3	2
Recall (%)	92.11	91.86	91.49
Precision (%)	73.13	69.61	64.89
F1 (%)	81.26	80.20	75.43

it changes the density of the ground plane of the point clouds resulting in the network learning an incorrect representation. The network works by taking a random sample of points and using the Local Spatial Encoding unit to create a feature vector for each point that encodes information about the surrounding points. Random under sampling of the ground class would result in the network learning a less dense representation of the ground during training, resulting in the model being unable to properly classify the original point cloud. Because of these results, no further exploration into random under sampling was done.

The grid search was continued on the rest of the hyperparameters and cost sensitive methods for handling class imbalance. The average performance metrics for each combination of parameters are given in Tab. 7. The following conclusions about the effect of the hyperparameters and class balancing methods were drawn from the results: decreasing the learning rate resulted in a decrease in the performance of the model; increasing the cost of a false negative resulted in the number of false negatives decreasing, but the number of false positives tended to increase; increasing the batch size when the learning rate was 1e-2 resulted in the number of false negatives and false positives decreasing.

After analyzing the results of the grid search, a question arose as to whether the false positive rate could be decreased by increasing the cost of false positives in the cost matrices. The results of the second grid search, performed with the modified cost matrices, are given in Tab. 8 and show that increasing the cost of a false positive resulted in fewer false positive classifications as seen through the increase in the precision scores. When the cost of a false positive was adjusted to 5 is had little effect on the recall scores. However, when the cost of a false positive was increased to 10, the recall scores decreased, indicating that increasing the cost too much had a negative impact on the number of false negatives.

4.1.1. Performance on the Test Set

The top three performing models based on the F1-scores from training and validation, whose parameters are summarized in Tab. 9, were run on the test set of data to determine the unbiased performance of each model. The results of the test are given in Tab. 10, and are broken down by test area, where test area one and two are from traverse one, test area three is from traverse two and test area four is from traverse three of the dataset. Additionally, the average performance results over all four test sets of data are provided for each tested model. An important result is that the models all achieved high recall scores,

Table 7: Performance results of models tested during a grid search of the hyperparameters and methods for handling class imbalance

Model Description			**Performance**		
Learning Rate	Cost Method	Batch Size	Recall (%)	Precision (%)	F1 (%)
1e-2	Inverse	6	92.11	73.13	81.26
1e-2	Inverse	12	96.52	74.54	84.12
1e-2	[0, 1; 10, 0]	6	89.66	50.17	64.34
1e-2	[0, 1; 10, 0]	12	93.19	66.60	77.69
1e-2	[0, 1; 50, 0]	6	90.47	53.21	67.01
1e-2	[0, 1; 50, 0]	12	94.89	55.96	70.40
1e-4	Inverse	6	93.49	40.04	56.07
1e-4	Inverse	12	90.80	26.64	41.19
1e-4	[0, 1; 10, 0]	6	92.13	57.21	70.59
1e-4	[0, 1; 10, 0]	12	84.33	54.31	66.07
1e-4	[0, 1; 50, 0]	6	92.22	33.21	48.83
1e-4	[0, 1; 50, 0]	12	88.64	24.43	38.30

Table 8: Performance results of models tested with the iterated set of cost methods for handling class imbalance

Model Description			**Performance**		
Learning Rate	Cost Method	Batch Size	Recall (%)	Precision (%)	F1 (%)
1e-2	[0, 5; 30, 0]	6	92.55	84.04	88.09
1e-2	[0, 5; 30, 0]	12	87.22	88.82	88.01
1e-2	[0, 10; 30, 0]	6	91.68	87.45	89.51
1e-2	[0, 10; 30, 0]	12	83.20	91.11	86.98
1e-2	[0, 5; 50, 0]	6	92.45	79.86	85.69
1e-2	[0, 5; 50, 0]	12	90.57	76.01	82.65
1e-2	[0, 10; 50, 0]	6	84.19	89.59	86.80
1e-2	[0, 10; 50, 0]	12	88.24	88.27	88.26

Table 9: Parameters of the top three performing models during training and validation on the Katwijk Beach Planetary Rover Dataset

Model Number	Number of Layers	Batch Size	Learning Rate	Cost Method
1	4	6	1e-2	[0, 5; 30, 0]
2	4	6	1e-2	[0, 10; 30, 0]
3	4	12	1e-2	[0, 10; 50, 0]

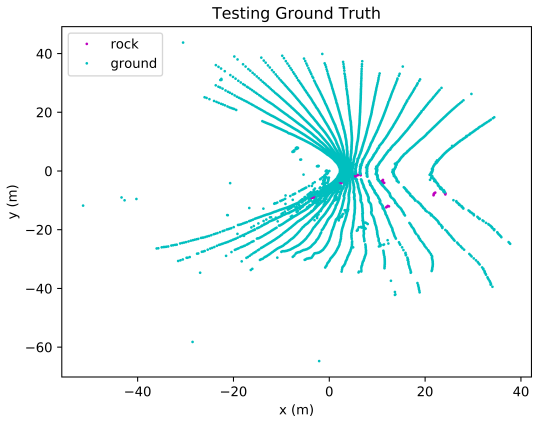
Table 10: Performance of the top three models on the test set of the Katwijk Beach Planetary Rover Dataset

Model Number	Test area	Recall (%)	Precision (%)	F1 (%)
1	1	89.67	96.71	93.06
	2	98.13	77.27	86.46
	3	100.00	74.27	85.24
	4	93.75	57.69	71.43
	Average	**95.39**	**76.49**	**84.05**
2	1	88.50	89.76	89.13
	2	95.59	89.08	92.22
	3	100.00	92.70	96.21
	4	93.75	68.18	78.95
	Average	**94.46**	**84.93**	**89.13**
3	1	94.13	87.75	90.83
	2	98.30	78.24	87.13
	3	100.00	64.80	78.64
	4	93.75	50.85	65.93
	Average	**96.54**	**70.41**	**80.63**

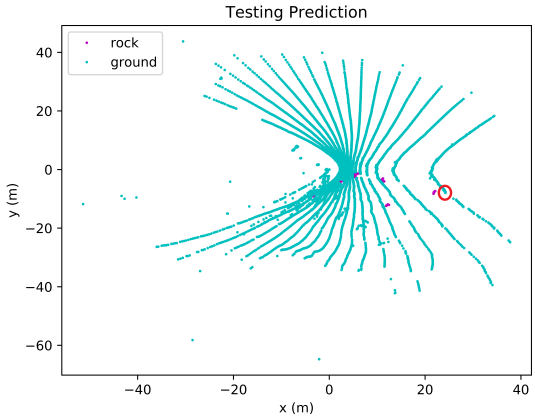
with scores ranging from 88.50 to $100.00\,\%$, but the precision scores were lower and had a larger range of 50.85 to $96.71\,\%$. A high recall score indicates a low false negative rate which is very important because it indicates that rocks are not being missed in the point cloud scenes. The low precision scores indicate that a high number of false positives are occurring; however, false positives are not a always a bad thing depending on where they occur. If a false positive occurs next to a true rock location, the false positive can be considered padding around the rock; this will be discussed further in Section 4.4. Overall, the model with the best performance based on the F1-score is model number two which achieved an average F1-score of $89.13\,\%$. The performance of the model can be seen in Fig. 3 which shows a comparison of ground truth scans to the models predictions from a bird's-eye view of point clouds from the test sets.

4.2. Semantic Segmentation on the Analogue Terrain Facility Dataset

The validation results obtained after training the top three models on the ATF dataset, given in Tab. 11, show that the models achieved good precision and recall scores, indicating that the models are capable of learning more complex representations of obstacles in planetary scenes.



(a) Ground truth of point cloud from test area one



(b) Prediction of point cloud from test area one

Figure 3: Comparison of ground truth and predicted labels for four point clouds from the test set of the Katwijk Beach Planetary Rover Dataset where misclassifications are circled in red

The ATF dataset contains real rocks, so their size, shape, and texture are different than the artificial rocks placed in the Katwijk Beach Planetary Rover Dataset. The artificial rocks repeat in size and shape throughout the dataset, whereas in the ATF dataset each rock is unique. The performance of the models on the training set are given in Tab. 12 and show that all three models achieved high recall and precision scores, but model number two performed the best with an F1-score of $91.40\,\%$.

4.3. Real-Time Implementation Analysis

The average time required for the models to perform semantic segmentation of a point cloud scene is presented in Tab. 13. The results show that average time per point cloud required for each model to perform semantic segmentation on the test set is very similar: $0.0068\,\mathrm{s}$ separate the fastest model from the slowest model. The small time difference between models is likely because the only variation between the tested models is the cost matrix applied to handle the class imbalance. Changing the cost matrix should not affect the models speed, but only the model's ability to predict a correct classifications. If the trained model was applied to the rover, with similar compute capabilities as the tested CPU, it would provide the ability to perform scene segmentation at a rate of approximately $0.63\,\mathrm{s}$ per point cloud. Further, the rate Velodyne VLP-16 Lidar publishes point clouds at a rate of 10 point clouds per second, so the model would be capable of segmenting approximately every sixth point cloud recorded by the Lidar in real-time.

Table 11: Validation performance of the models on the ATF dataset

Model Description			**Performance**		
Learning Rate	Cost Method	Batch Size	Recall (%)	Precision (%)	F1 (%)
1e-2	[0, 5; 30, 0]	6	93.26	87.46	90.27
1e-2	[0, 10; 30, 0]	6	92.43	90.69	91.55
1e-2	[0, 10; 50, 0]	12	91.11	90.78	90.94

Table 12: Performance of the models on the test set of the ATF dataset

Model Description			**Performance**		
Learning Rate	Cost Method	Batch Size	Recall (%)	Precision (%)	F1 (%)
1e-2	[0, 5; 30, 0]	6	98.27	81.75	89.25
1e-2	[0, 10; 30, 0]	6	96.28	86.99	91.40
1e-2	[0, 10; 50, 0]	12	98.17	81.38	88.99

4.4. Impact of False Negative and False Positive Classifications

The seriousness of false negative and false positive classifications depends on two factors: the location of the point and the classification of the neighboring points. If a false negative occurs but some or all other points belonging to that rock are correctly identified then the false negative is not very serious because the identified rock area will be padded and avoided. When a false negative occurs and no other points in that rock are correctly identified then the false negative is a very serious misclassification be-

Table 13: Average time per point cloud required to perform semantic segmentation on the Katwijk Beach Planetary Rover Dataset test set

Model Number	Time (Seconds/Point Cloud)
1	0.6306
2	0.6238
3	0.6275

cause a rock has been missed and may damage the rover if it runs into the missed rock. Additionally, false negatives are more likely to occur farther away from the rover because the scan density of Lidar point clouds decrease with distance; therefore, far away rocks have a lower density representation than close rocks and are easier to misclassify. However, false negatives which occur far away from the rover are not as serious because the rover is not close enough to be damaged by the missed rock and as the rover moves closer to the rock, the rock density in the point cloud will increase and the rock will be more likely to be detected. If a false positive occurs and its neighboring points belong to a rock then the misclassification is not very serious because the rock will be padded and avoided anyways. If a false positive occurs and its neighboring points do not belong to a rock, it will result in the rover unnecessarily avoiding the area and can be serious if the rover drives on a longer path than necessary, or if too many false positives occur that are not near true rocks, making the terrain seem untraversable.

5. CONCLUSION

This work presents a way to use NNs to efficiently segment obstacles from Lidar point clouds and proves the viability of performing the segmentation on-board a rover. The scope of this work was to use pre-processed point cloud data with per-point labels to train NN machine learning models to perform semantic segmentation of point cloud scenes that represent planetary environments, and to detect obstacles within the point clouds. Modifications to a NN called RandLA-Net, including modifying the data processing, tuning the hyperparameters, and applying methods to handle the class imbalance, resulted in the best overall segmentation accuracy of $99.68\,\%$, where each point was identified as either the rock class or the ground class. The model achieved a recall score of $94.46\,\%$ and a precision score of $84.93\,\%$. A segmentation rate of 0.6238 seconds per point cloud was achieved by the model on an Intel Xeon E5-2665 CPU, indicating that the model could be deployed on-board a rover with similar processing capabilities. Overall, the resulting technique developed could be confidently used to inform a rover's path planner of the location of surrounding obstacles in environments with sandy terrain with rock obstacles.

REFERENCES

[1] The cameras on the Mars 2020 Perseverance rover, . URL https://mars.nasa.gov/mars2020/spacecraft/rover/cameras/#Engineering-Cameras.

[2] Communications, . URL https://mars.nasa.gov/mars2020/spacecraft/rover/communications/.

[3] R. Hewitt, E. Boukas, M. Azkarate, M. Pagnamenta, J. Marshall, A. Gasteratos, and G. Visentin. The Katwijk beach planetary rover dataset. International Journal of Robotics Research, Dec 2017. doi: 10.1177/0278364917737153.

[4] Q. Hu, B. Yang, L. Xie, S. Rosa, Y. Guo, Z. Wang, N. Trigoni, and A. Markham. RandLA-Net: Efficient semantic segmentation of large-scale point clouds. 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2020. doi: 10.1109/cvpr42600.2020.01112.

[5] C. R. Qi, L. Yi, H. Su, and L. J. Guibas. PointNet : Deep hierarchical feature learning on point sets in a metric space. Jun 2017.