IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 16 January 2024, accepted 26 January 2024, date of publication 6 February 2024, date of current version 15 February 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3362804

RESEARCH ARTICLE

Dust De-Filtering in LiDAR Applications With Conventional and CNN Filtering Methods

TYLER PARSONS[1], JAHO SEO[1], BYEONGJIN KIM[2], HANMIN LEE[2], JI-CHUL KIM[2], AND MOOHYUN CHA[2]

[1]Department of Automotive and Mechatronics Engineering, Ontario Tech University, Oshawa, ON L1G 0C5, Canada
[2]Department of Smart Industrial Machine Technologies, Korea Institute of Machinery & Materials (KIMM), Daejeon 34103, South Korea

Corresponding author: Jaho Seo (Jaho.Seo@ontariotechu.ca)

This work was supported in part by the Korea Institute of Machinery & Materials (KIMM).

ABSTRACT Light detection and ranging (LiDAR) sensors can create high-quality scans of an environment. However, LiDAR point clouds are affected by harsh weather conditions since airborne particles are easily detected. In literature, conventional filtering and artificial intelligence (AI) filtering methods have been used to detect, and remove, airborne particles. In this paper, a convolutional neural network (CNN) model was used to classify airborne dust particles through a voxel-based approach. The CNN model was compared to several conventional filtering methods, where the results show that the CNN filter can achieve up to 5.39% F1 score improvement when compared to the best conventional filter. All the filtering methods were tested in dynamic environments where the sensor was attached to a mobile platform, the environment had several moving obstacles, and there were multiple dust cloud sources.

INDEX TERMS Artificial intelligence (AI), autonomous navigation, convolutional neural network (CNN), dust de-filtering, light detection and ranging (LiDAR).

I. INTRODUCTION

Light Detection and Ranging (LiDAR) is commonly used in autonomous navigation applications to create high-resolution maps of the surrounding area [1]. In autonomous vehicles, LiDAR plays a key role in obstacle detection and avoidance since collisions can be extremely dangerous and expensive. Although LiDAR can produce high-resolution point clouds, harsh environmental factors can degrade the quality of the scanned environment. Factors such as dust, snow, rain, and other small airborne particles can be detected because of the short wavelength (900 nm) of the signal [2], thus causing a lot of noise in a point cloud. Because of this, it becomes difficult to distinguish between obstacles and airborne particles.

In literature, there exists several methods to distinguish between airborne particles and solid objects. One approach is to fuse multiple sensors together [3], [4], such as a depth camera and LiDAR [5]. By combining multiple sensors, discrepancies between the sensors can be easily detected. However, hardware and physical limitations (not enough room for more sensors) may not make this approach ideal. Modern technological advances have allowed for smaller sensors and all-in-one packages which can overcome this issue, but this requires additional costs which may not be ideal. So, another approach is to use conventional particle filtering methods on the LiDAR point cloud. In literature, there are several conventional algorithms that can be used, such as the Radius Outlier Removal (ROR) filter [6], [7], Dynamic Radius Outlier Removal (DROR) filter [8], Statistical Outlier Removal (SOR) filter [9], and Low-Intensity Outlier Removal (LIOR) filter [10]. These methods make use of the point cloud's geometry and light intensity to filter outlier points that have properties of airborne particles. However, the conventional methods sometimes remove points that are not airborne particles, which results in the removal of important environmental features such as obstacles.

Another method of airborne particle removal can be with Artificial-Intelligence (AI) techniques. Some well studied AI techniques that have been applied to detect airborne particles are the Random Forest (RF) classifier [11], Support Vector

The associate editor coordinating the review of this manuscript and approving it for publication was Brian Ng.

Machine (SVM) classifier [11], [12], K-Nearest Neighbors (KNN) classifier [12], Neural Network (NN) classifier [11], and Deep Neural Network (DNN) classifier [13]. These approaches require a large amount of data to train a model such that it can accurately make predictions in real-time. Thus, the data and features used to train the model has a large influence on the quality of the predictions. Existing literature states that a combination of geometry features and light-intensity features through a voxel-based approach is ideal for training AI methods for airborne particle detection [1], [11], [13]. Compared to the conventional methods, some AI approaches can remove airborne particles with greater accuracy while still maintaining important environmental information [1], [13].

In the mentioned literature, both conventional and AI filtering methods have been discussed. As explained in the mentioned works, conventional filtering methods perform well under ideal conditions when dust and non-dust particles are easily distinguishable. However, it is speculated that a CNN based approach can outperform the conventional filtering methods, especially in unideal conditions. This assumption is supported by the fact that AI based approaches have been used for airborne particle classification in the mentioned works, and have a good performance. Several research papers have used CNNs for airborne particle filtering with slightly different approaches. For example, Heinzler et al. proposed a CNN model named WeatherNet which is based on LiLaNet to filter fog and rain in point clouds [14]. They also propose a method of automatically labelling experimental data by crossreferencing a sample environment scan under ideal conditions (no rain or fog). They transform the 3D LiDAR data into two 2D images, one for depth and one for intensity. This is done by ''unralvelling'' the 360° scan into a 2D matrix with pixel intensities corresponding to the depth and intensity of the captured points. In the 2D matrix, each row corresponds to one of the 32 vertically stacked send/recieve modules, and each column corresponds to one of the 1800 segments over the whole scanning range. Another study expands on the work of Heinzler et al. by adding a local radius outlier removal (LROR) filter after using a CNN to filter rain and fog [15]. Their CNN model is named SunnyNet and is based upon the WeatherNet model with some minor modifications. Similarily, they also use the 2D depth and one for intensity images for their network. Sebastian et al. conduct an in depth study of snowy, foggy, and rainy conditions in 3D LiDAR applications [16]. What they found was that the eignevectors and eigen values can effectively capture different weather conditions. This concept was extended to not only detecting the atmospheric conditions, but the road conditions as well. Specifically, their model can detect snow, light fog, dense fog, and rain in the atmophere, and full snow coverage, slushy, and wet road conditions. Similarily, they also used the 2D depth image representation of the 3D LiDAR data for their CNN structure named RangeWeatherNet, which is a redesign of the DarkNet archetechture. In [17], the authors propose a different method to represent the LiDAR data: the bird's-eye view. The bird's-eye view is a depth image projected on a different axis compared to other literature. They used this for their CNN model named MobileWeatherNet, which is a modified structure based on the work of Simonyan and Zisserman [18], to detect rain and fog. Their work shows that the bird's-eye view is better for classifying different weather conditions.

The mentioned works highlight the novelty of each respective filter, however, a comparitive study between a selection of conventional filtering methods and CNN based approach have not been explored. Additionally, many of the existing CNN based approaches for LiDAR filtering and weather applicaions do not consider dust as one of the conditions. In this paper, a novel CNN based approach is proposed to classify and filter airborne dust particles in LiDAR point clouds for autonomous excavation applications. In existing studies, CNN based approaches have been used for airborne particle filtering. However, in these studies the voxel-feature tabular data is converted to an image-like structure before being fed into the model. In this study, the tabular data is used as the input to the CNN model, and reshaping functions are used to convert the tabular data into an image-like structure directly within the model. Therefore, the training process can optimize the conversion between the tabular data and image structure. Additionally, this work is applied to dust de-filtering applications only, however, the same methodology can be applied in other airborne particle filtering applications under various adverse weaather conditions such as snow, fog, and rain. The proposed model will be compared to conventional filtering methods with ground-truth data to evaluate the effectiveness of the model. The main contributions of our work can be summarized as follows.

- A large-scale voxel-based dataset was collected and labelled for the CNN training.
- A novel CNN structure was proposed that converts the voxelized features into a image-like structure for feature extraction.
- This study considers several conventional filtering methods, such as the ROR, DROR, SOR, and LIOR, provides a comprehensive analysis on each filter's de-dusting performance compared to the developed CNN model.
- The voxelized classification results from the CNN were converted to a point-based classification as to properly compare to the results generated from the conventional methods.

The remainder of this paper is divided into the following sections. Section II describes the theoretical background and working principles of the conventional filtering methods and the CNN architecture. Section III describes the collection and preparation of the training data, as well as the metrics used to quantify the improvements of the CNN approach. Section IV covers a complete analysis of the classification results of the conventional methods and CNN method, the

improvements made are highlighted in this section. Finally, Section V discusses the concluding remarks, limitations, and future work for the proposed area of research.

II. CONVENTIONAL AND PROPOSED FILTERING METHODOLOGY

This section will present the operating principle and theoretical background regarding the selected conventional filtering algorithms and the proposed CNN-based filtering method.

A. RADIUS OUTLIER REMOVAL FILTER (ROR)

The ROR filter eliminates outlier points in 3D space through a geometrical approach [7]. By iterating though each point in the point cloud, a sphere can be created with a predefined search radius (SR), and the neighboring points can be counted. If the number of neighbors is less than a defined threshold, the point will be classified as an outlier (dust) and removed from the point cloud. The input parameters for this algorithm are the SR and minimum acceptable number of neighboring points. An example of the ROR filter being applied to 2D point data can be seen in Fig. 1. In 3D, the circles created by the SR would simply be modelled as spheres.

The ROR has some drawbacks with LiDAR applications. Specifically, due to the sensor resolution, the density of the point cloud changes with respect to the radial distance. As a result, non-dust points may be classified as dust, resulting in a filtered point cloud that removed important environmental information.

B. DYNAMIC RADIUS OUTLIER REMOVAL FILTER (DROR)

The DROR filter was developed to resolve the problem seen in the ROR regarding the changing point cloud density for further objects [7]. It operates like the ROR, except that the SR is a function of the point's coordinates and the sensor resolution. The SR can be calculated in (1), where x and y are the coordinates of the point, β is a multiplier constant greater than 1, and α is the angular resolution of the LiDAR.

$$SR = \beta * \sqrt{x^2 + y^2} * \alpha \tag{1}$$

There should also be minimum SR (SR_{min}) defined for points that are near the sensor, otherwise no neighboring points will be detected for a SR defined in (1). So, the SR is calculated for all points, but when it is below SR_{min}, the SR is simply set to SR_{min}. The input parameters for the DROR filter are the multiplier constant (β), angular resolution (α), minimum SR (SR_{min}), and the minimum acceptable neighboring points. An example of the DROR filter being applied to 2D point data can be seen in Fig. 2. In 3D, the circles created by the SR would simply be modelled as spheres.

C. STATISTICAL OUTLIER REMOVAL FILTER (SOR)

The SOR filter detects outliers in the point cloud using a statistical approach [19]. First, the point cloud statistics need to be calculated. By iterating through all points in the point cloud, the average distance (d_k) of k nearest neighbors can be calculated. Based on the average distances calculated, the mean average distance (μ) and deviation (σ) can be calculated. With these parameters, the statistical outliers can be identified using (2), where β is a specified multiplier.

$$\mu - \beta * \alpha \leq d_k \leq \mu + \beta * \alpha \tag{2}$$

Based on (2), each points average distance for k nearest neighbors can be calculated, and if it is outside of the statistical range, it is classified as dust. The drawback with this approach is that it is computationally expensive. Since the mean average distance and deviations need to be calculated to classify them, the point cloud needs to be iterated over

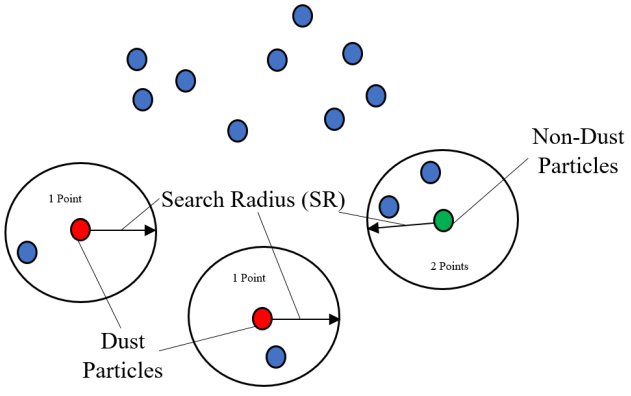

FIGURE 1. ROR example for 2D data. In this example, the non-dust threshold was set to 2.

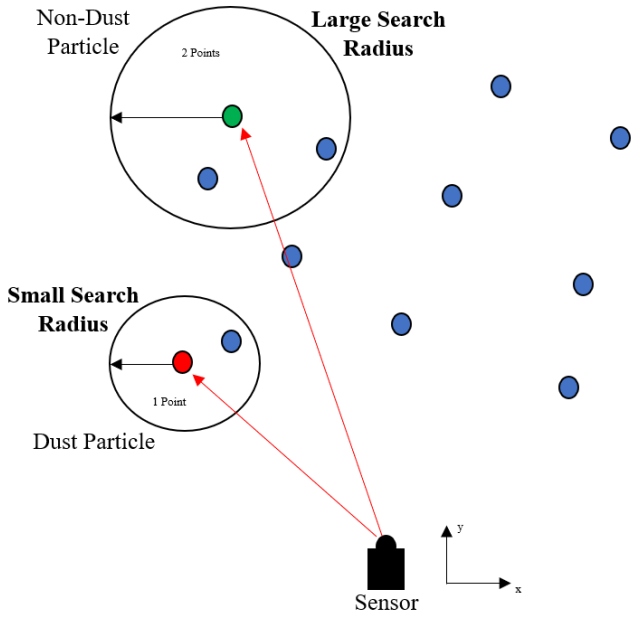

FIGURE 2. DROR example for 2D data. In this example, the non-dust threshold was set to 2.

twice. The input parameters for this approach are the number of neighboring points and the constant multiplier.

D. LOW-INTENSITY OUTLIER REMOVAL FILTER (LIOR)

The ROR, DROR, and SOR all rely strictly on the geometry of the points within the point cloud. However, LiDAR can measure the intensity of the signal obtained reflecting off an object. The LIOR approach uses this valuable information in addition to the geometry of the points [10]. As discussed in [1], the LIOR filter consists of two stages.

In the first stage, the points within the point cloud are filtered with respect to a predefined intensity value. The second stage uses the geometry information of the points removed in the first stage. The second stage can use either the ROR or the DROR filter previously discussed, thus creating the LIOR-ROR or LIOR-DROR filters. If the point is classified as an outlier in the first and second stage, then it is classified as dust. If the point is calculated as an outlier in the first stage but not the second stage, then it is not classified as dust and is kept in the original point cloud.

Since LIOR is paired with either ROR or DROR, the input parameters are the same except for the intensity threshold used in the first stage. To properly select the intensity threshold, a study should be conducted with ground-truth labelled data. The LIOR filter adds an additional measure to the ROR and DROR filters by identifying possible dust points, then the second stage can preserve non-dust points that may have a relatively smaller intensity value.

E. PROPOSED FILTERING METHODOLOGY: CONVOLUTIONAL NEURAL NETWORK FILTER (CNN)

In literature, the CNN [20] is commonly used in object detection and recognition in images. The CNN can be applied in many applications ranging from medical image classification for disease diagnosis [21] to facial recognition [22]. As seen in a majority of applications, the CNN model consists of an input layer, several hidden layers, and an output layer used for classification [23]. The operation types within the hidden layers consist of the convolutional layer (feature extraction), pooling layer, and fully connected layer [23]. The structure of the CNN, the order in which the layers occur, and the parameters selected have a large influence on the performance of the model. Thus, these metrics need to be carefully selected to yield a high-quality model.

Sometimes, a CNN may be applied to tabular data rather than images, as seen in [24] and [25]. In these cases, the tabular data is converted into an image-like structure that can be used by the CNN. As explained by Zhu et al. [25], the tabular data is converted to an image where each entry in the tabular data is converted to a pixel. They state that the location of the pixels should be optimized such that similar features are close together in the image.

As seen in literature, LiDAR data can be voxelized [26], and features can be calculated using a principle component analysis (PCA), which will be discussed in detail in the next section. This results in a collection of tabular data where each voxel consists of several metrics calculated using the PCA. So, the input of the CNN model is tabular data representing a voxel containing several LiDAR points. As described in literature, the tabular data should be converted into an image-like structure for the CNN to function as intended.

The proposed CNN model takes tabular data as the input and is immediately passed through a dense layer. Rather than converting the tabular data to an image before being trained, the proposed CNN model converts the tabular data to an image after the first dense layer. This process increases the dimensionality of the input vector, then the data can simply be rearranged into a 2D image. In doing so, the model can train the dense layer weights such that the optimal tabular data-to-image conversion is obtained.

After the first dense layer, the resulting vectors can be flattened and reshaped into an image. Then, the first 2D convolution layer is applied. For this process, a ReLU activation function was selected, which can be formulated in (3). The ReLU function outputs the input (x) if it is positive, and 0 otherwise [27]. The ReLU function was selected because of its simple implementation, yet effective performance. The ReLU function was used for each convolutional layer.

$$f(x) = \max(0, x) \tag{3}$$

After the first convolutional layer, a 2D max pooling function is used on the tensors produced. 2D max pooling works by creating a pool (typically a 2D matrix of a specified size) and sliding it over a tensor in the X and Y directions. The maximum value found within a pool will be added to output tensor [28]. This process down samples the input tensor and extracts important features. A visual representation of the 2D max pooling operation can be seen in Fig. 3. The 2D max pooling operation is done after every convolutional layer.

In total, there are 3 2D convolutional layers paired with 2D max pooling in series. After the final 2D max pooling layer, the tensor is flattened and used as the input to the final dense layer with a sigmoid activation function, which is ideal for binary classification. The sigmoid activation function can be formulated in (4), where it takes any real number as an input,

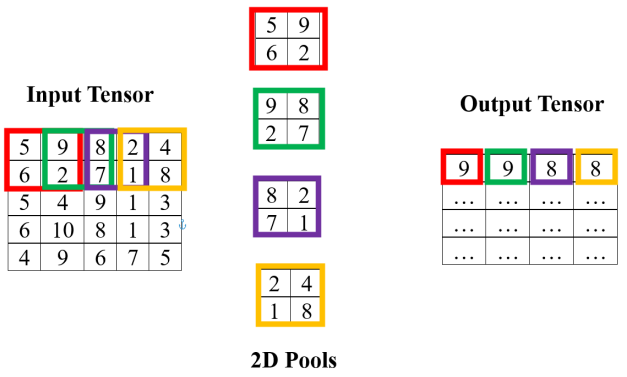

FIGURE 3. Example of a 2D max pooling operation.

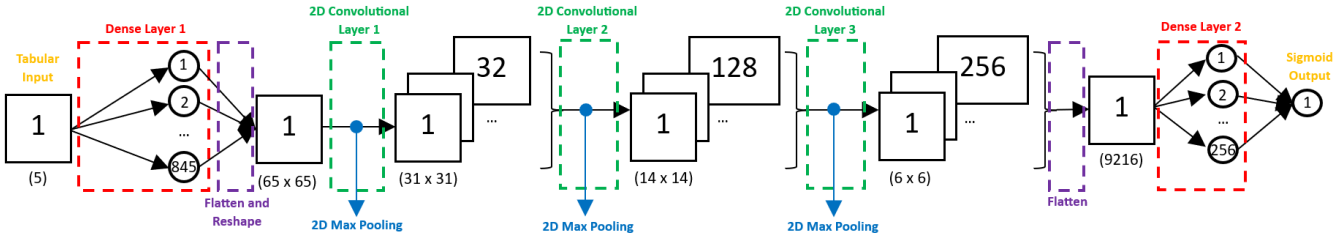

FIGURE 4. The proposed structure for dust de-filtering.

and the output is in the range of 0 to 1 [29]. This models the probability of the voxel being either dust or non-dust.

$$f(x) = \frac{1}{1 + e^{-x}} \tag{4}$$

A graphical representation of the proposed CNN structure can be seen in Fig. 4, where the dimension of the tensors is highlighted at each layer. The first convolutional layer has 32 3 × 3 filters, the second convolutional layer has 128 3 × 3 filters, and the third convolutional layer has 256 3 × 3 filters. Each 2D max pooling layer has a pool size of 2 × 2. The proposed CNN model was assembled using the Python TensorFlow library [30].

III. EXPERIMENT

To train any AI model, a large labelled dataset is a necessity. So, the team conducted several experiments and test runs to gather enough data to train a robust model.

The training data used was gathered from 2 experiments. In both experiments, a VLP-16 LiDAR sensor was used [31]. The first experiment consisted of a stationary sensor with a non-dust target and dust cloud that varied in distance from the sensor. The distances of the dust cloud and non-dust target for the first experiment can be seen in Table 1. For the first experiment, only 1 leaf blower was active for all test runs. The second experiment was a more realistic scenario with dynamic environmental factors. Specifically, the VLP-16 was mounted on a mobile platform and there were dynamic objects in the environment (people walking), as well as a dust cloud generated from up to two leaf blowers. This results in a dust cloud that varied in density depending on how many leaf blowers (i.e., 1 or 2 blowers) were active. The initial distances of the dust cloud and non-dust target for the second experiment can also be seen in Table 1. For the second experiment, the number of active leaf blowers was changed for each test run. Meaning that each test run had 1 leaf blower active, then an additional leaf blower was activated to create the dense dust cloud. Also, the distance of the dynamic non-dust obstacle was varied as it moved through the environment. An example of the data collection environment can be seen in Fig. 5. To label the LiDAR data collected, the LiDAR labeller app in MATLAB was used [32].

TABLE 1. First and second experiment dust and non-dust target distances.

Experiment	Test Run	Dust Cloud Distance	Non-Dust Target Distance
First	1	4 m	5 m
	2	5 m	10 m
	3	8 m	10 m
	4	10 m	15 m
Second	1	4 m	5 m
	2	6 m	5 m
	3	7 m	10 m
	4	11 m	10 m

As seen in Fig. 5, only the data directly in front of the sensor was considered. There are two reasons for this. The first reason is that the dust clouds were generated only in front of the sensor. If the full field of view of the LiDAR sensor was considered, a significant amount of non-dust particles would be captured from behind the sensor. If this dataset was used for training, it would result in a biased model due to the imbalanced classes. The second reason is that the proposed dust de-filtering technology was developed for vehicles that only travel in the forwards direction. Meaning that only the environment directly in front of the vehicle needs to be considered. Additionally, having less points to process can reduce the processing time, which is ideal for real-time applications.

To extract features in the point cloud to use for training, the point clouds were voxelized. Specifically, an octree structure [33] was used for the voxelization. The octree decomposition begins with a region of interest (ROI), and the ROI is continuously segmented into children voxels given that some condition is met. Some conditions may include the maximum number of children partitions, the minimum voxel dimension, or the voxel capacity. In some cases, the voxels may be different sizes.

For this application, each voxel must contain at least 4 points to do the PCA. The PCA of a point distribution in a voxel is derived from the least squares estimation [26]. Single value decomposition can be used to derive the normal vector of the best-fit plane by solving an eigensystem [34]. The eigenvalues found using the PCA can be used to compute several important geometrical traits of the voxelized data. The calculated eigenvalues can be ordered as follows, $\lambda_0 \geq \lambda_1 \geq \lambda_2$ [35]. The geometric features calculated using the eigenvalues are based on the roughness [11], planarity [36],

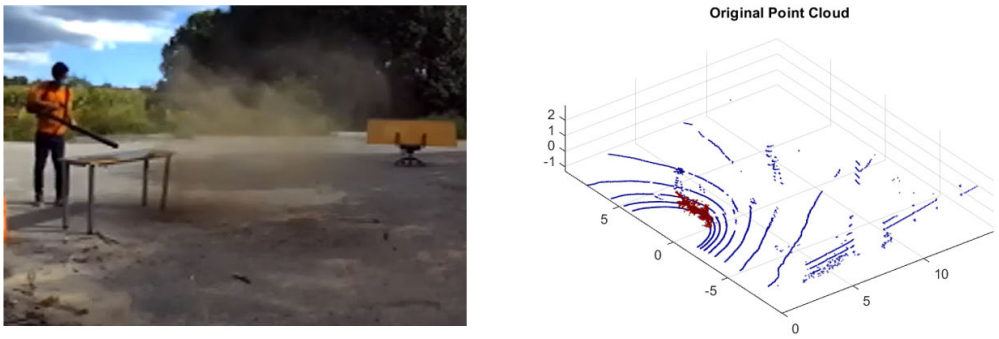

FIGURE 5. The environment in which the training and testing LiDAR data was collected. The MATLAB LiDAR labeler app was used to label the dust points seen in red.

and curvature [36]. The selected geometric features can be formulated in (5) to (7).

$$f_1 = \frac{\lambda_2}{\lambda_0} \tag{5}$$

$$f_2 = \frac{\lambda_1}{\lambda_0} \tag{6}$$

$$f_3 = \frac{\lambda_0}{\lambda_0 + \lambda_1 + \lambda_2} \tag{7}$$

In addition to the geometric features in (5) to (7), intensity features were also used. The mean and deviation intensities of the points in each voxel were calculated as the final two features used for training. The mean and deviation of intensity calculations can be seen in (8) and (9), where int_i is the intensity of point i in the voxel, n is the number of points in a voxel, and $\overline{int}$ is the mean intensity seen within the voxel.

$$f_4 = \frac{1}{n}\sum_{i=1}^{n} int_i \tag{8}$$

$$f_5 = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(int_i - \overline{int})^2} \tag{9}$$

A summary of the training dataset can be seen in Table 2. When training the CNN, the dataset is stratified and split 70% for training, and 30% validation.

A. METRICS FOR IMPROVEMENT

To quantify the performance of the proposed CNN model, it will be compared to the conventional filtering methods. Since the conventional filtering methods are applied to the points, and the CNN filtering method is applied to the voxelized point cloud, some conversions are needed as to properly compare the results side-by-side. To do so, the resulting voxel classifications are applied to all points within the voxel. For example, if a voxel is classified as dust by the CNN, then all points within the voxel are dust.

To quantify performance of the discussed filters, the F1 score will be computed. The F1 score is defined as the harmonic mean of precision and recall [37]. The F1 score was selected since it considers true positives, false positives, and false negatives in its calculation. The mathematical formulation of precision, recall, and the F1 score can be seen in (10) to (12). In (10) and (11), *TP* represents the true positives, which are the ground-truth dust particles that have been successfully classified as dust, *FP* represents the false positives, which are the non-dust particles that are classified as dust, and *FN* represents the false negatives, which are the dust particles that are classified as non-dust. The F1 score formulated in (12) considers both the precision (p) and recall (r).

$$p = \frac{TP}{TP + FP} \tag{10}$$

$$r = \frac{TP}{TP + FN} \tag{11}$$

$$F1 = \frac{2}{r^{-1} + p^{-1}} \tag{12}$$

TABLE 2. Voxelized training data for the CNN model.

Non-Dust Voxels	Dust Voxels
253, 362	10, 742

IV. RESULTS AND DISCUSSION

In this section, a complete analysis of the conventional methods and CNN model will be conducted for airborne dust particle filtering. To compare the performance of each filtering method, 4 frames worth of LiDAR data will be tested. The previous research only tested the conventional methods with the static environment [1], so the 4 frames will be selected from the dynamic experiment since it is closer to the real-world application of the proposed research.

A. CONVENTIONAL METHODS

As previously discussed, the ROR, DROR, SOR, and LIOR filtering methods will be selected to compare against the proposed CNN model. The parameters selected for the conventional filtering methods can be summarized in Table 3.

In Table 3, all values were selected based on trial and error to achieve the best performance for the respective filtering

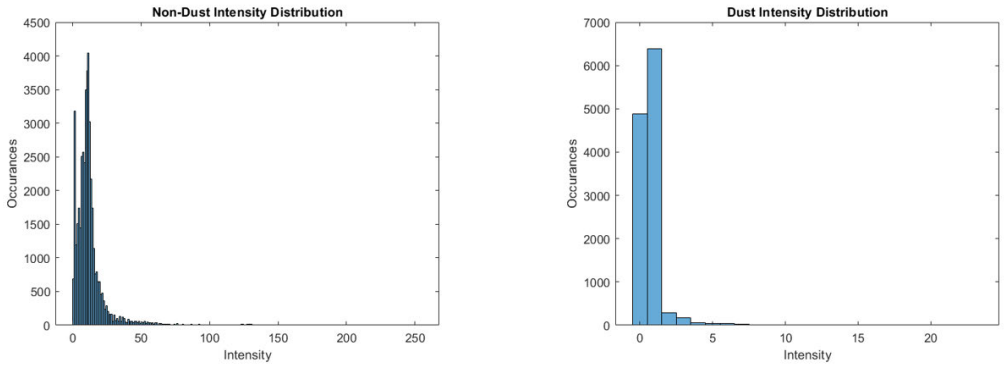

FIGURE 6. Intensity distribution plots for the non-dust (a) and dust (b) particles.

TABLE 3. Parameters for the ROR, DROR, SOR, and LIOR conventional filters.

Filter	Parameter	Value
ROR	SR	0.04 m
	Minimum Acceptable Points	3
DROR	SR_{min}	0.04 m
	Minimum Acceptable Points	3
	Multiplier Constant (β)	0.05
	Angular Resolution (α)	0.25
SOR	k Nearest Neighbors	3
	Multiplier Constant	0.2
LIOR-ROR	SR	0.043 m
	Intensity Threshold	3
	Minimum Acceptable Points	6
LIOR-DROR	SR_{min}	0.05 m
	Intensity Threshold	3
	Minimum Acceptable Points	4
	Multiplier Constant (β)	0.03
	Angular Resolution (α)	0.25

TABLE 4. F1 score for the conventional and CNN filters on 4 different LiDAR frames from the dynamic environment.

	Testing Frame			
Filter	**1**	**2**	**3**	**4**
ROR	26.65%	17.10%	13.99%	25.91%
DROR	72.39%	75.69%	80.10%	45.28%
SOR	27.62%	48.97%	64.64%	28.69%
LIOR-ROR	64.04%	41.98%	30.55%	57.29%
LIOR-DROR	82.76%	85.93%	91.07%	81.85%
CNN	82.16%	88.87%	95.57%	87.24%

algorithm, except for the intensity threshold in the LIOR filters. For this, a study was conducted using the ground-truth labelled data. The intensity distribution was examined for the 4 test frames, and the threshold was selected based on these results. The intensity distribution plots can be seen in Fig. 6. From the plots seen in Fig. 6, the recommended threshold intensity for identifying dust particles is about 3. Some non-dust particles do go less than 3. However, since the LIOR filters are paired with ROR and DROR, this portion of the algorithm aims to preserve these outliers.

The performance of the conventional filters can be summarized in Table 4. Examining the results presented in Table 4, the best conventional airborne dust filtering algorithm is the LIOR-DROR, whereas the best conventional filter that does not use the point intensity is the DROR filter. The worst conventional filter is the ROR filter, with an F1 score ranging from 13.99% to 26.65%. The results show that when a conventional filter is paired with a two stage LIOR approach, the F1 score can improve. This is seen when comparing the ROR filter to the LIOR-ROR, and the DROR filter to the LIOR-DROR.

Since the LIOR-DROR filter was the best amongst the conventional filters, the LIOR-DROR F1 scores will be used to compare to the CNN filter.

B. CONVOLUTIONAL NEURAL NETWORK

The CNN model was trained using the voxelized dataset discussed in Section III in about 4 hrs. Examining Table 2, the dataset consists of approximately 4% dust voxels. Meaning that the training dataset is extremely biased towards non-dust voxels. Unfortunately, there are no publically available dust LiDAR datasets, so the training data was limited to the data collected in the discussed experiments. Additionally, manually labelling all the point clouds in the MATLAB LiDAR labeller app was a time-consuming process.

So, a few different data processing techniques were applied to observe which method yielded the best F1 score in the validation stage of training. First, the minority class (dust voxels) was oversampled using the synthetic minority oversampling technique (SMOTE) [38]. The SMOTE works by operating in the feature space of the minority class to generate synthetic data. The synthetic data is created by selecting features along line segments that join any/all of the minority class's nearest neighbors [38].

The other approach was to stratify the dataset. By allowing the same fraction of the minority class to be in the training

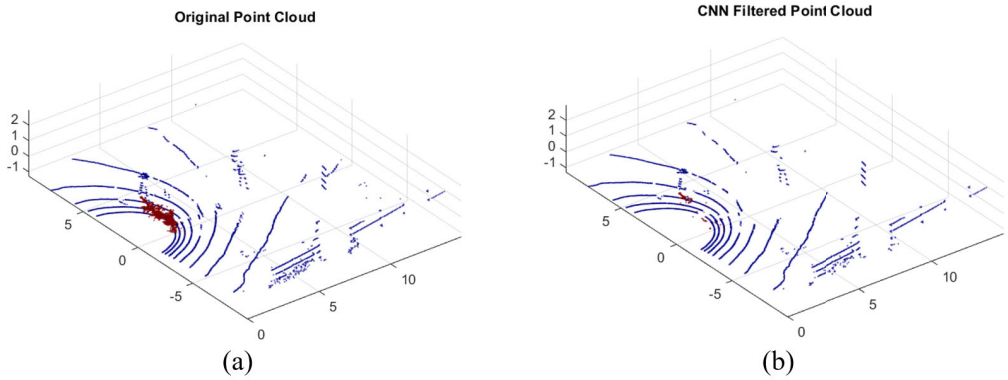

FIGURE 7. Testing frame 1 before (a) and after (b) CNN airborne dust filtering. The red points are ground truth labelled dust points.

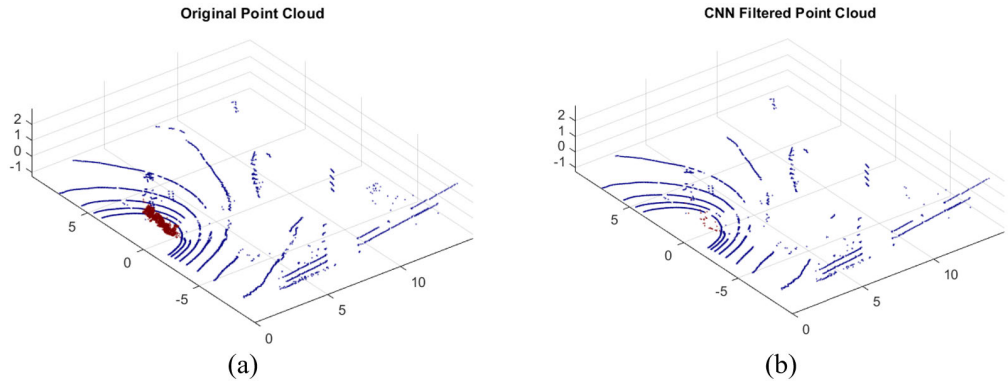

FIGURE 8. Testing frame 2 before (a) and after (b) CNN airborne dust filtering. The red points are ground truth labelled dust points.

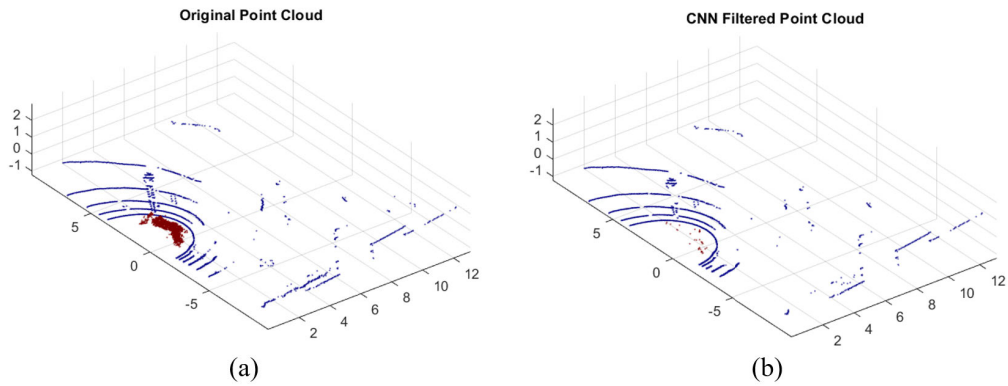

FIGURE 9. Testing frame 3 before (a) and after (b) CNN airborne dust filtering. The red points are ground truth labelled dust points.

and validation partitions, the imbalance can be minimized. Through experimentation, it was found that the SMOTE technique could not accurately produce synthetic dust voxel data, resulting in a CNN model that had a low F1 score when tested with the mentioned 4 frames. So, the CNN model was trained using the stratified dataset. The testing results for the CNN model can be seen in Table 4, along with the conventional filtering results.

Examining the F1 scores in Table 4, the CNN model outperformed the best conventional filter (LIOR-DROR) for 3 out of 4 of the testing point clouds. The performance of the CNN was comparable to that of the LIOR-DROR filter for the first testing dataset with a difference of 0.6%. However, the other 3 frames saw an improvement of up to 5.39%, thus validating the CNN filtering method as an effective means of classifying airborne dust particles. A side-by-side

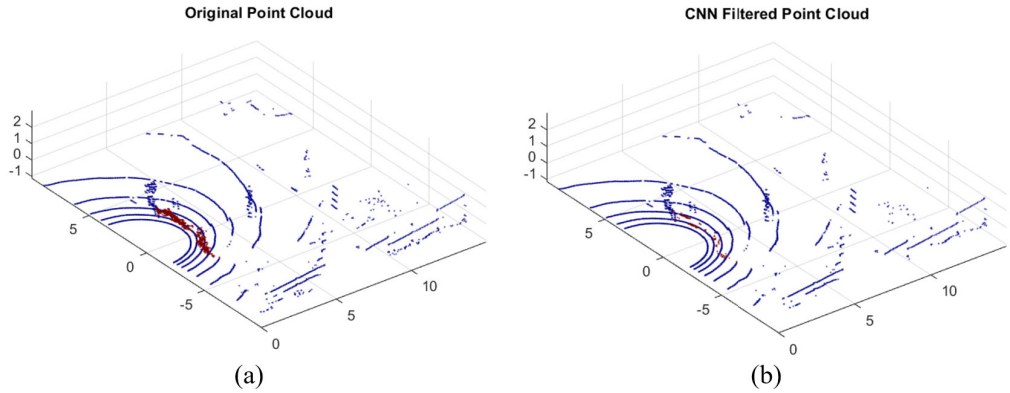

FIGURE 10. Testing frame 4 before (a) and after (b) CNN airborne dust filtering. The red points are ground truth labelled dust points.

comparison of each testing frame before and after applying the CNN filter can be seen in Fig. 7 to Fig. 10. Examining Fig. 7 to Fig. 10, most of the dust (red LiDAR points) were effectively eliminated while still preserving important environmental information, such as the ground and solid obstacles. This performance is also reflected in the F1 scores found in Table 4.

V. CONCLUSION

In this paper, a CNN model was proposed to classify airborne dust particles in autonomous excavation applications. Since CNNs are commonly applied to image-based classification, the tabular data (voxel features) was converted to an image-like structure by using the CNNs dense layer and reshaping function. In doing so, the proposed model can optimize the weights of the dense layer such that the best image structure is obtained. This differs from existing research where the tabular data is converted to an image-like structure before being fed to the CNN model. In existing studies, extensive research was conducted to find the optimal tabular data-to-image conversion method. In this study, this process is encorporated into the model itslef, so the training process can find the appropriate conversion. Additionally, this study conducted a comprehensive analysis of the proposed CNN model and several conventional de-dust filtering methods. This comparison provides insight regarding the performance of each filtering method in de-dusting applications.

The proposed CNN model was compared to several conventional filtering methods, namely the ROR, DROR, SOR, LIOR-ROR, and LIOR-DROR for 4 frames selected from the dynamic environment experiment. The results show that the CNN model can outperform the best conventional filtering method in 3 of the test frames, whereas the performance was comparable in the one without improvement.

Some drawbacks of this research include the limited training data for dust classification. Since there are no publicly available dust LiDAR datasets, the training data needs to be collected and labelled manually through the discussed experiments. This can be done in the MATLAB LiDAR labeller app; however, it is time consuming to gather enough data for training a CNN model. Additionally, the amount of dust samples in the training data was very limited (approximately 4% dust samples). If more dust samples were collected, the F1 score of the CNN model may increase in the proposed testing cases.

Future work for this research includes applying the CNN model in real-time. This paper mainly focuses on the validation of the proposed CNN model, which was conducted offline after the data was collected. Additionally, the performance may be improved by fusing multiple sensors together. For example, LiDAR data can be fused with depth camera data, which is what will be achieved in the future work as well. Finally, the background environment can be added after the removal of the dust particles. In the proposed research, the airborne dust particles are removed from the point cloud. However, since non-dust objects may exist behind a dust cloud, the dust cloud can be blocking the non-dust objects. When this happens, the dust cloud can be removed through filtering, but the non-dust objects behind the dust cloud are not detected. This presents a safety concern since information is missing regarding the obstacles that exist behind dust clouds. So, future work aims to fuse a static environment scan with the filtered point cloud such that the environment behind the dust cloud is maintained. This can be done by filtering the point cloud captured in real-time, and adding non-dust objects from the static environment scan that were blocked by the dust cloud. However, this poses an additional concern regarding the dynamic non-dust objects. So, additional methods may be used to track dynamic non-dust objects in real time and add them to the filtered environment (in their predicted location) if they are blocked by a dust cloud.

ACKNOWLEDGMENT

The authors would like to thank Ali Afzalaghaeinaeini for his previous research in this area. Afzalaghaeinaeini helped to capture and label the experimental data used in this study. He also has conducted detailed studies of the conventional de-dust filtering methods, thus paving the way for their research.

REFERENCES

[1] A. Afzalaghaeinaeini, J. Seo, D. Lee, and H. Lee, "Design of dust-filtering algorithms for LiDAR sensors using intensity and range information in off-road vehicles," *Sensors*, vol. 22, no. 11, p. 4051, May 2022, doi: 10.3390/s22114051.

[2] T. G. Phillips, N. Guenther, and P. R. McAree, "When the dust settles: The four behaviors of LiDAR in the presence of fine airborne particulates," *J. Field Robot.*, vol. 34, no. 5, pp. 985–1009, Feb. 2017, doi: 10.1002/rob.21701.

[3] T. Peynot and A. Kassir, "Laser-camera data discrepancies and reliable perception in outdoor robotics," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Oct. 2010, pp. 2625–2632.

[4] M. P. Gerado-Castro, T. Peynot, and F. Ramos, "Laser-radar data fusion with Gaussian process implicit surfaces," in *Field and Service Robotics*, 105th ed. New York, NY, USA: Springer, 2015, pp. 289–302, doi: 10.1007/978-3-319-07488-7_20.

[5] G. Xie, J. Zhang, J. Tang, H. Zhao, N. Sun, and M. Hu, "Obstacle detection based on depth fusion of LiDAR and radar in challenging conditions," *Ind. Robot: Int. J. Robot. Res. Appl.*, vol. 48, no. 6, pp. 792–802, Nov. 2021, doi: 10.1108/ir-12-2020-0271.

[6] A. Aldoma, Z.-C. Marton, F. Tombari, W. Wohlkinger, C. Potthast, B. Zeisl, R. B. Rusu, S. Gedikli, and M. Vincze, "Tutorial: Point cloud library: Three-dimensional object recognition and 6 DOF pose estimation," *IEEE Robot. Autom. Mag.*, vol. 19, no. 3, pp. 80–91, Sep. 2012.

[7] N. Charron, S. Phillips, and S. L. Waslander, "De-noising of LiDAR point clouds corrupted by snowfall," in *Proc. 15th Conf. Comput. Robot Vis. (CRV)*, Toronto, ON, Canada, May 2018, pp. 254–261.

[8] Y. Cao, H. Huang, and D. Yu, "Filter methods for removing falling snow from light detection and ranging point clouds in snowy weather," *Sensors Mater.*, vol. 34, no. 12, p. 4507, Dec. 2022, doi: 10.18494/sam4047.

[9] R. B. Rusu and S. Cousins, "3D is here: Point cloud library (PCL)," in *Proc. IEEE Int. Conf. Robot. Autom.*, Shanghai, China, May 2011, pp. 1–4.

[10] J.-I. Park, J. Park, and K.-S. Kim, "Fast and accurate desnowing algorithm for LiDAR point clouds," *IEEE Access*, vol. 8, pp. 160202–160212, 2020, doi: 10.1109/ACCESS.2020.3020266.

[11] L. Stanislas, N. Suenderhauf, and T. Peynot, "LiDAR-based detection of airborne particles for robust robot perception," in *Proc. Australas. Conf. Robot. Autom. (ACRA)*, 2018, pp. 1–8.

[12] A. U. Shamsudin, K. Ohno, T. Westfechtel, S. Takahiro, Y. Okada, and S. Tadokoro, "Fog removal using laser beam penetration, laser intensity, and geometrical features for 3D measurements in fog-filled room," *Adv. Robot.*, vol. 30, nos. 11–12, pp. 729–743, Jun. 2016, doi: 10.1080/01691864.2016.1164620.

[13] A. Afzalaghaeinaeini, "Design of dust-filtering algorithms for LiDAR sensors in off-road vehicles using the AI and non-AI methods," M.S. thesis, Dept. Auto. Mech. Eng., Ont. Tech., Oshawa, ON, Canada, 2022.

[14] R. Heinzler, F. Piewak, P. Schindler, and W. Stork, "CNN-based LiDAR point cloud de-noising in adverse weather," *IEEE Robot. Autom. Lett.*, vol. 5, no. 2, pp. 2514–2521, Apr. 2020, doi: 10.1109/LRA.2020.2972865.

[15] Z. Luo, J. Ma, G. Xiong, X. Hu, Z. Zhou, and J. Xu, "Semantic segmentation based rain and fog filtering only by LiDAR point clouds," in *Proc. IEEE Int. Conf. Unmanned Syst. (ICUS)*, Guangzhou, China, Apr. 2022, pp. 90–95.

[16] G. Sebastian, T. Vattem, L. Lukic, C. Burgy, and T. Schumann, "RangeWeatherNet for LiDAR-only weather and road condition classification," in *Proc. IEEE Intell. Vehicles Symp. (IV)*, Nagoya, Japan, Jul. 2021, pp. 777–784.

[17] M. P. D. Silva, D. Carneiro, J. Fernandes, and L. F. Texeira, "Mobile WeatherNet for LiDAR-only weather estimation," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Gold Coast, QLD, Australia, 2023, pp. 1–8.

[18] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *Proc. Int. Conf. Learn. Represent.*, 2015, pp. 1–14.

[19] R. B. Rusu, Z. C. Marton, N. Blodow, M. Dolha, and M. Beetz, "Towards 3D point cloud based object maps for household environments," *Robot. Auto. Syst.*, vol. 56, no. 11, pp. 927–941, Nov. 2008.

[20] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, May 2015, doi: 10.1038/nature14539.

[21] S. S. Yadav and S. M. Jadhav, "Deep convolutional neural network based medical image classification for disease diagnosis," *J. Big Data*, vol. 6, no. 1, pp. 1–18, Dec. 2019, doi: 10.1186/s40537-019-0276-2.

[22] Q. Sun and A. Redei, "Knock knock, who's there: Facial recognition using CNN-based classifiers," *Int. J. Adv. Comput. Sci. Appl.*, vol. 13, no. 1, pp. 9–16, 2022, doi: 10.14569/ijacsa.2022.0130102.

[23] X. He, Y. Wang, X. Wang, W. Huang, S. Zhao, and X. Chen, "Simple-encoded evolving convolutional neural network and its application to skin disease image classification," *Swarm Evol. Comput.*, vol. 67, Dec. 2021, Art. no. 100955, doi: 10.1016/j.swevo.2021.100955.

[24] M. Artzi, E. Redmard, O. Tzemach, J. Zeltser, O. Gropper, J. Roth, B. Shofty, D. A. Kozyrev, S. Constantini, and L. Ben-Sira, "Classification of pediatric posterior fossa tumors using convolutional neural network and tabular data," *IEEE Access*, vol. 9, pp. 91966–91973, 2021, doi: 10.1109/ACCESS.2021.3085771.

[25] Y. Zhu, T. Brettin, F. Xia, A. Partin, M. Shukla, H. Yoo, Y. A. Evrard, J. H. Doroshow, and R. L. Stevens, "Converting tabular data into images for deep learning with convolutional neural networks," *Sci. Rep.*, vol. 11, no. 1, p. 11325, May 2021, doi: 10.1038/s41598-021-93376-5.

[26] M. Wang and Y. Tseng, "Incremental segmentation of LiDAR point clouds with an octree-structured voxel space," *Photogramm. Rec.*, vol. 26, no. 133, pp. 32–57, Mar. 2011, doi: 10.1111/j.1477-9730.2011.00624.x.

[27] C. Banerjee, T. Mukherjee, and E. Pasiliao, "An empirical study on generalizations of the ReLU activation function," in *Proc. ACM Southeast Conf.*, Kennesaw, GA, USA, Apr. 2019, pp. 164–167.

[28] B. Zhao, Y. S. Chong, and A. Tuan Do, "Area and energy efficient 2D max-pooling for convolutional neural network hardware accelerator," in *Proc. 46th Annu. Conf. IEEE Ind. Electron. Soc.*, Singapore, 2020, pp. 423–427.

[29] G. Mourgias-Alexandris, A. Tsakyridis, N. Passalis, A. Tefas, K. Vyrsokinos, and N. Pleros, "An all-optical neuron with sigmoid activation function," *Opt. Exp.*, vol. 27, no. 7, p. 9620, Apr. 2019, doi: 10.1364/oe.27.009620.

[30] *TensorFlow*. Accessed: Jun. 4, 2023. [Online]. Available: https://www.tensorflow.org/

[31] *Velodyne LiDAR Puck*. Accessed: Jun. 4, 2023. [Online]. Available: https://www.amtechs.co.jp/product/VLP-16-Puck.pdf

[32] *Getting Started With LiDAR Labeler*. Accessed: Jun. 4, 2023. [Online]. Available: https://www.mathworks.com/help/lidar/ug/lidar-labeler-get-started.html

[33] Y.-T. Su, J. Bethel, and S. Hu, "Octree-based segmentation for terrestrial LiDAR point cloud data in industrial applications," *ISPRS J. Photogramm. Remote Sens.*, vol. 113, pp. 59–74, Mar. 2016, doi: 10.1016/j.isprsjprs.2016.01.001.

[34] N. Lord, "Matrix computations, 3rd edition," *Math. Gazette*, vol. 83, no. 498, pp. 556–557, Nov. 1999, doi: 10.2307/3621013.

[35] J.-F. Lalonde, N. Vandapel, D. F. Huber, and M. Hebert, "Natural terrain classification using three-dimensional ladar data for ground robot mobility," *J. Field Robot.*, vol. 23, no. 10, pp. 839–861, 2006, doi: 10.1002/rob.20134.

[36] Y. Lil, L. Ma, Z. Zhong, F. Liu, M. A. Chapman, D. Cao, and J. Li, "Deep learning for LiDAR point clouds in autonomous driving: A review," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 32, no. 8, pp. 3412–3432, Aug. 2021, doi: 10.1109/TNNLS.2020.3015992.

[37] K. Takahashi, K. Yamamoto, A. Kuchiba, and T. Koyama, "Confidence interval for micro-averaged f1 and macro-averaged f1 scores," *Appl. Intell.*, vol. 52, no. 5, pp. 4961–4972, Jul. 2021, doi: 10.1007/s10489-021-02635-5.

[38] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "SMOTE: Synthetic minority over-sampling technique," *J. Artif. Intell. Res.*, vol. 16, pp. 321–357, Jun. 2002, doi: 10.1613/jair.953.

TYLER PARSONS received the Bachelor of Engineering (Hons.) and Master of Applied Science degrees from Ontario Tech University, ON, Canada, in 2021 and 2023, respectively. His main area of research for graduate studies was in route optimization and global path planning. Since graduating, he has been an Associate Researcher with the Autonomous Vehicle and Electro-Hydraulic Control (AVEC) Laboratory, Ontario Tech University. His current research interests include advance sensing technology and path planning optimization.

JAHO SEO received the B.S. degree in agricultural machinery and process engineering from Seoul National University, Seoul, South Korea, in 1999, the M.E. degree in mechanical engineering from the University of Quebec (École de Technologie Supérieure), Montreal, Canada, in 2006, and the Ph.D. degree in mechanical engineering from the University of Waterloo, Waterloo, Canada, in 2011. He was with the Department of Mechanical and Mechatronics Engineering, University of Waterloo, as a Postdoctoral Fellow, in 2011; the Department of System Reliability, Korea Institute of Machinery & Materials (KIMM), as a Senior Researcher, from 2012 to 2016; and the Department of Biosystems Machinery Engineering, Chungnam National University, South Korea, as an Assistant Professor, from 2016 to 2017. Since 2017, he has been an Assistant Professor with the Department of Automotive and Mechatronics Engineering, Ontario Tech University, where he has been involved in research on the development of autonomous control systems for intelligent mobile machines.

BYEONGJIN KIM received the B.S. degree in electrical engineering and the Ph.D. degree in convergence IT engineering from the Pohang University of Science and Technology (POSTECH), in 2015 and 2021, respectively. Currently, he is a Senior Researcher with the Department of Smart Industrial Machine Technologies, Korea Institute of Machinery & Materials (KIMM). His research interests include sensing systems under extreme environment and autonomous driving of unmanned systems and field robotics.

HANMIN LEE received the B.S., M.S., and Ph.D. degrees in mechanical engineering from the Korea Advanced Institute of Science and Technology (KAIST), in 1998, 2000, and 2005, respectively. He is currently a Principal Researcher and the Head of the Department of Smart Industrial Machine Technologies, Korea Institute of Machinery & Materials (KIMM). His current research interest includes autonomous driving and manipulation in off-road environments.

JI-CHUL KIM received the bachelor's degree in mechanical engineering from Yonsei University, in 2007, and the master's and Ph.D. degrees in mechanical engineering from KAIST with a focus on robotics, in 2009 and 2014, respectively. Currently, he is a Senior Researcher with the Department of Smart Industrial Machine Technologies, Korea Institute of Machinery & Materials (KIMM). His main research interests include unmanned systems technology and automatic control.

MOOHYUN CHA received the B.S. degree from the Pohang University of Science and Technology (POSTECH) and the M.S. degree from KAIST. Currently, he is a Principal Researcher with the Department of Smart Industrial Machine Technologies, Korea Institute of Machinery & Materials (KIMM). His research interests include various virtual reality (VR) applications for engineering and training and computer graphics and data processing for very large engineering datasets.