Towards Autonomous Driving: A Machine Learning-based Pedestrian Detection System using 16-Layer LiDAR

Stefan Mihai, Purav Shah, Glenford Mapp, Huan Nguyen, and Ramona Trestian

London Digital Twin Research Centre,
Faculty of Science and Technology,
Middlesex University, London, UK
E-mails: SM3488@live.mdx.ac.uk, {p.shah, g.mapp, h.nguyen, r.trestian}@mdx.ac.uk

***Abstract*—The advent of driverless and automated vehicle technologies opens up a new era of safe and comfortable transportation. However, one of the most important features that an autonomous vehicle requires, is a reliable pedestrian detection mechanism. Many solutions have been proposed in the literature to achieve this technology, ranging from image processing algorithms applied on a camera feed, to filtering LiDAR scans for points that are reflected off pedestrians. To this extent, this paper proposes a machine learning-based pedestrian detection mechanism using a 16-layer Velodyne Puck LITE LiDAR. The proposed mechanism compensates for the low resolution of the LiDAR through the use of linear interpolation between layers, effectively introducing 15 pseudo-layers to help obtain timely detection at practical distances. The pedestrian candidates are then classified using a Support Vector Machine (SVM), and the algorithm is verified by accuracy testing using real LiDAR frames acquired under different road scenarios.**

***Index Terms*—pedestrian detection, LiDAR, interpolation, vehicular networks, Support Vector Machine, Velodyne.**

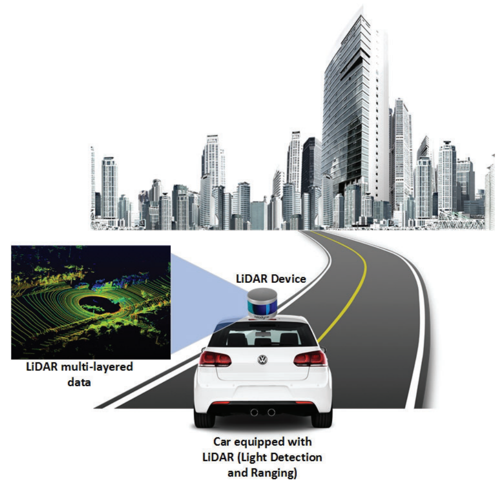


Fig. 1: Car Equipped with LiDAR - Example Scenario

I. Introduction

The significant advancements in sensor, perception and on-board computation technologies has resulted in the rapid development of autonomous vehicles aiming at increasing human life quality by decreasing the number of traffic accidents and reducing the traffic jams. However, one of the major challenges when it comes to autonomous driving networks, is dealing with different uncertainties (e.g., pedestrian/objects detection, emergency braking, etc.). The autonomous vehicles need to be able to identify the surrounding environment and provide an accurate, reliable and quick response to various situations [1].

Over the years, pedestrian detection algorithms have evolved, ever-increasing in complexity and requirements in order to overcome limitations and achieve great accuracy. Advancements in autonomous vehicles and the advent of Intelligent Transportation Systems have increased demand for fail-proof systems that can pinpoint pedestrians in all kinds of driving conditions. Whatever the method used, the algorithms that reached peak accuracy and versatility are understandably either very complex, very expensive, or both, making them unattractive for car manufacturers. As such, simple algorithms that could use cost effective equipment and return reliable results are very much sought after.

The majority of pedestrian recognition approaches are based on camera systems because of their high-resolution, the easily classifiable information they provide (like shape, colour, brightness), and their availability in vehicles, which already use them for multiple other purposes, like lane-assist systems and dashcams. However, these methods increase in complexity and requirements with the increase in demands like spatial positioning and 24/7 availability.

LiDAR (Light Detection and Ranging)-based systems provide cost-efficient and simple solutions to issues left unresolved by image processing. LiDARs are systems that emit layers of lasers with high-directivity that reflect on all surrounding objects and return information about their relative distance and material properties as illustrated in Figure 1. These aspects make LiDARs extremely suitable for pedestrian recognition applications in intelligent vehicles, where correct distance estimation is an important parameter for collision avoidance.

This paper proposes a new method for achieving pedestrian detection using low-resolution equipment complemented by simple processing techniques. The system uses a 16-layer Velodyne Puck LITE LiDAR and an algorithm rooted in linear interpolation for data pre-processing and Support Vector Machine (SVM) for classification.

The rest of the paper is divided as follows: Section II explores related work on this topic, Section III provides an overview of the system, then offers an in-depth look into its work-flow, Section IV is focused on a discussion based on the technique's results, and Section V brings our conclusions and insights for future work.

II. Related Work

In the research literature, there are several approaches proposed that aim at achieving object classification and pedestrian detection and tracking using LiDAR technology, some of which will be summarised below. The common challenge researchers face when designing such a system is the optimisation of the trade-off between the pre-processing part, where data is filtered, and the classification model used. Both components, if poorly chosen and configured, can induce undue latency, create a large processing load, or increase the costs of implementation. The LiDAR's specifications also play an important role in the complexity and limitations of the proposed algorithms and their use-cases. High-resolution LiDARs provide large amounts of data per each scan, which requires more processing power to work with to avoid slowing down the algorithm. They are also more expensive.

Ogawa et al. [2] used DENSO's in-vehicle high-resolution LiDAR to capture point clouds, which were then clustered according to the points' position and reflection intensity. The authors also propose a tracking method using an Interactive Multiple Model filter. The pedestrian objects are identified using a Bayesian Classifier. The algorithm has a high accuracy for obstacle detection, before classification, and it obtains better results for long-distance detection than camera-based techniques. However, there is a drop in accuracy after classification due to the weak classification algorithm.

Lin et al. [3] used a 64-layer Velodyne LiDAR to distinguish pedestrian and vehicle objects. The candidate objects are extracted by subtracting the previously obtained background point cloud from the object-populated point cloud. The points are then clustered together and an eight elements long feature vector is formed to train a Support Vector Machine. The algorithm returns excellent results for linearly separable feature vectors, but a previously acquired background scan is not always feasible in driving scenarios. Similarly, Kidono et al. [4] used a 64-layer Velodyne LiDAR and a nine elements long features vector, including the distribution of reflection intensity and an innovative slicing feature. The data is classified using a Support Vector Machine, obtaining good classification results.

Kunisada et al. [5] proposed a pedestrian detection method using one-dimensional Convolutional Neural Networks (CNN), in which the approach is to decide whether each individual point is part of a pedestrian object, instead of first clustering points together and passing clusters through classifiers. This work evaluates the distance information obtained from each virtual Velodyne VLP-16 LiDAR layer as a 1D waveform, and feeds the concatenation of all 16 waveforms into a CNN, obtaining good accuracy for classification at very low latency. The method manages to avoid the problem of conventional approaches that use distance-based clustering, which is that two objects would be clustered together as an individual object because they were too close to each other. However, to achieve this performance, the algorithm requires significant processing power.

While the LiDAR provides excellent ranging, reflectance information and day-long availability, it does not supply enough deterministic information for some classifiers to reliably identify pedestrians. For this reason, research has been carried out to develop joint camera and LiDAR systems that harness the advantages of both devices to obtain better accuracy results in a variety of driving scenarios. Jun et al. in [6] and [7] proposed a camera and LiDAR fusion method in which pedestrian detection is achieved in three stages: LiDAR subsystem, in which a Velodyne HDL-64E LiDAR is used to capture point clouds that are further filtered using a sliding window mechanism scanning across the ground plane, a camera subsystem where image features are extracted and scored, then finally a fusion subsystem that gives a final decision based on the results of both subsystems. Another approach was put forward by Melotti et al. [8] who studied the use of CNN to identify pedestrians using LiDAR-generated depth and reflectance maps. The authors implemented a binary classification algorithm using early and late fusion strategies in CNN and concluded that these fusion schemes greatly increase detection accuracy between the ”pedestrian” and ”non-pedestrian” classes.

Yamamoto et al. [9] studied the use of the Active Scan LiDAR to leverage its ability to control the direction of its lasers in order to develop an algorithm that first estimates a pedestrian candidate's position, then uses those coordinates to concentrate the LiDAR's lasers on the object, obtaining a dense point cloud to classify. However, the paper does not present a method for accurate pedestrian detection.

III. Pedestrian Detection Algorithm

A. Overview

An overview of the proposed pedestrian detection algorithm is depicted in Figure 2. The model is split into several functional components: (1) *LiDAR Measurements* - LiDAR data is acquired and usually represented by a cloud of 3D points dense enough to provide the shapes of the surrounding environment (e.g., cars, pedestrians, objects, curbs, etc.); (2) *the pre-processing block* - where the LiDAR point cloud is clustered, filtered and interpolated, and (3) *the detection block* - in which the feature vector is extracted and used to train a Support Vector Machine classifier. A detailed explanation of these functional blocks is given in subsequent subsections.

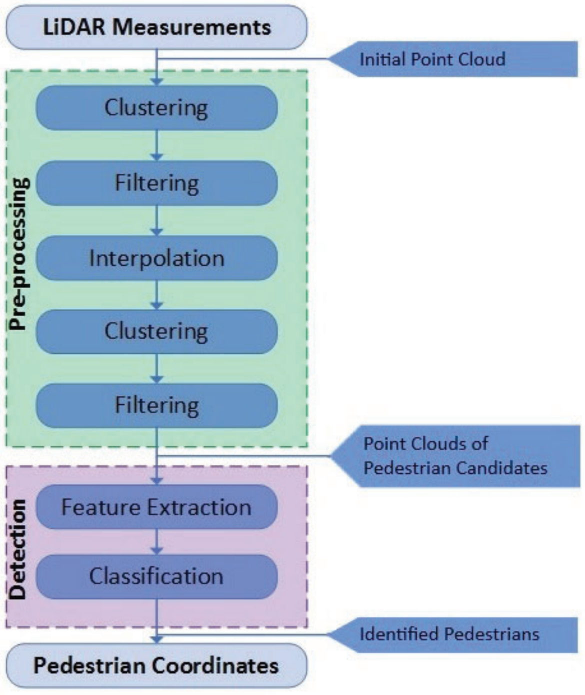


Fig. 2: Algorithm overview

B. Pre-processing

The pre-processing block focuses on extracting pedestrian candidates from the point cloud measured by the LiDAR. The block takes as input a frame of LiDAR data, where each point is characterized by location data expressed in a three dimensional Cartesian coordinates system with the point of origin around the LiDAR. Additionally, the points contain information about the intensity of the reflected laser light, a parameter that depends on the material properties of the reflective object.

In order to make sense of the data, the point cloud is segmented into three classes: ego-vehicle points, ground points and obstacle points. The ego-vehicle is extracted by removing all the points in the close vicinity of the LiDAR corresponding to the vehicle's real dimensions. The ground points are the points which are located below a certain elevation angle difference threshold. The remaining points belong to obstacles.

The obstacle points have been *clustered* according to a given minimum Euclidean distance. Furthermore, since the distance between layers increases with the distance from the LiDAR, the chosen minimum Euclidean distance used for clustering needs to be at least equal to the maximum vertical distance between two adjacent layers at the minimum detection range. In other words, it needs to be large enough that all the points from layers reflected off a single obstacle at the minimum detection distance are clustered together. The minimum detection range is defined as the distance necessary for the vehicle to completely stop moving once a pedestrian appeared in front of it. In order to obtain the minimum detection range, one must consider the following:

- the ego-vehicle speed;
- the braking deceleration, which depends on the vehicle's characteristics and road conditions;
- the reaction time (processing time) of the algorithm.

Figure 3 illustrates the assumptions, limitations and logic behind this process, where ω is the maximum angular resolution between layers, a is the braking deceleration, v is the vehicle's speed, R is the minimum detection range, and d is the minimum Euclidean distance for clustering.

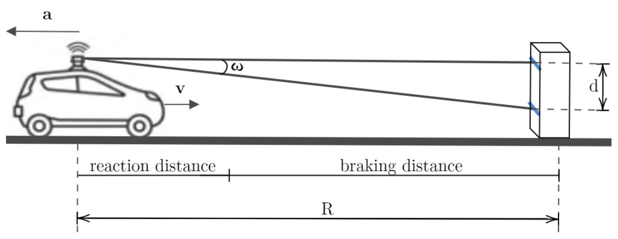


Fig. 3: Detection limit scenario

Once distance-based clustering is applied, the obtained clusters are *filtered* according to the number of points they contain. The threshold is chosen as the number of points that would normally be reflected off a pedestrian candidate object that is R meters away from the vehicle.

Depending on the boundary conditions chosen for the experiment, the resulting minimum Euclidean distance for clustering may be unreasonably large due to the low vertical resolution of the LiDAR. This may result in unreliable clustering, affecting the reliability of the pedestrian detection algorithm. To fix this issue, one can either impose new boundary conditions for the experiment, i.e. have no objects within a d radius of the pedestrians, or compensate through software improvement.

In order to preserve the minimum detection range needed for collision avoidance and obtain accurate distance-based clustering in reasonable conditions, this paper proposes *linearly interpolating* the LiDAR layers as illustrated in Figure 4.

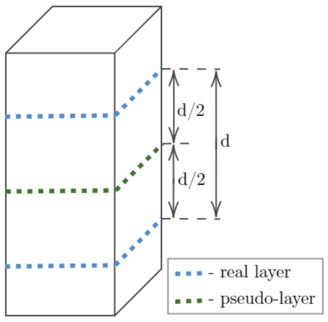


Fig. 4: Layer interpolation

This results in additional pseudo-layers which are placed at half the distance between the real layers, effectively reducing the minimum Euclidean distance needed for clustering by half. The algorithm then applies distance-based *clustering* again on the remaining point cloud using the new Euclidean distance as a radius. Then, it *filters* the obtained clusters according to the new number of points a pedestrian candidate object at the distance *R* would reflect. Each cluster is then encapsulated in a bounding box with corresponding height, length and width to fit all its points.

C. Detection

The Detection block focuses on the training of a Support Vector Machine model to classify the pedestrian candidates identified in the pre-processing phase into two categories: pedestrian and non-pedestrian. Each pedestrian candidate gets described by a set of features which are defined in Table I.

TABLE I: Features selection

Feature	Description
f_1	Number of points on the pedestrian candidate object
f_2	Distance from the LiDAR to the pedestrian candidate object
f_3	Body proportion - Height/Length
f_4	Body proportion - Height/Width
f_5	Bounding box volume
f_6	Mean of the distribution of points intensities for the object
f_7	Standard deviation of the distribution of points intensities for the object

The SVM is a supervised machine learning model that analyzes an object's feature vector to classify it into two different classes. In this case, the feature vector is defined as $\boldsymbol{f_i} = (f_1, f_2, f_3, ...f_7)^T$, where i is an iterative index for the pedestrian candidates. For training, individual frames of LiDAR data are captured and filtered in the Pre-processing phase of the algorithm, obtaining a corresponding number of pedestrian candidates per frame. These are further labeled as *pedestrian* or *non-pedestrian* with the help of a classifying vector $\boldsymbol{y}$, which contains values of $+1$ for the *pedestrian* class and -1 for the *non-pedestrian* class for each observation.

Consequently, the decision boundaries are given by:

$$y_i(\boldsymbol{\omega} \cdot \boldsymbol{f_i} + b) - 1 = 0 \tag{1}$$

where $\boldsymbol{\omega}$ is a parameter used to describe the margin between the two classes, and b is a real number.

In order to maximize the optimal margin between the classes, the non-separable character of the feature vector has been taken into consideration, resulting in a maximization problem as defined below:

$$\max_{\alpha} \left[\sum_i \alpha_i - \frac{1}{2} \sum_i \sum_j \alpha_i \alpha_j y_i y_j \boldsymbol{f_i} \cdot \boldsymbol{f_j} \right] \tag{2}$$

s.t.

$$\sum_i \alpha_i y_i = 0 \tag{3}$$
$$0 \leqslant \alpha_i \leqslant C$$

where C is the *box constraint*, a parameter equal to the maximum penalty for features that violate margins. In this case, the SVM uses a soft margin, meaning that it will allow some misclassification to happen in order to obtain a better classification accuracy overall.

The Support Vector Machine separates the observations in two classes with the help of a hyperplane, or a flat affine subspace, bounded by the decision boundaries described above. In cases where the chosen features do not clearly divide the two classes, with many overlaps between observations, the SVM can use *kernel functions* to describe the observations in a higher dimensional feature space, where separation by a hyperplane is feasible. There are many popular kernels, but for this work, the Radial Basis Function (RBF) proved to greatly increase accuracy.

IV. Experimental results

A. Implementation and evaluation

The algorithm has been developed using Matlab, running on a machine using an Intel(R) Core(TM) i7-6700HQ @ 2.60 GHz CPU with 8 GB RAM, and Windows 10 Pro operating system.

The proposed algorithm takes as input a frame of LiDAR data, a point cloud structured in 16 layers of 1806 points each, captured using a Velodyne Puck LITE installed on top of a vehicle at the height of 1.8 meters. The device has a range of 100 meters, a 30° vertical field of view distributed amongst the 16 layers, resulting in a 2° vertical angular resolution. The horizontal angular resolution varies between 0.1° and 0.4°, and the rotations frequency is equal to 10Hz, which translates into 10 frames per second. The captured frames are fed into the Pre-processing block which extracts the pedestrian candidates.

Before the first filtering, the minimum detection range, *R* has been determined by taking the following considerations into account:

- the ego-vehicle moves at a maximum speed of 50 km/h;
- the braking deceleration is equal to 9.8 m/s^2 [10];
- the reaction time (processing time) of the algorithm is at most 1s, which is the average reaction time of a normal driver attentive to the road, driving at the aforementioned speed [11].

The resulting value for *R* in these conditions is 22 meters and it serves as a reference distance at which to compute the vertical distance between points belonging to two different adjacent layers, previously defined as the minimum Euclidean distance for clustering, *d*. This distance is equal to 0.76 meters. As such, at this point in the algorithm, all the points belonging to objects that are at least 0.76 meters apart are clustered together. The resulting clusters are filtered according to the number of points they have, in order to remove the outlier data.

After the *layer interpolation*, the remaining data points are clustered again, this time with a minimum Euclidean distance of 0.38 meters, and filtered yet again.

Pedestrian candidates are extracted from the remaining clusters. Here, a pedestrian candidate object is defined as a cluster of points that is situated at a certain maximum height above the ground level and would fit in a bounding box of the following dimensions: 200x100x100cm.

For the purpose of this work, approximately 10,000 consecutive Velodyne Puck LITE LiDAR frames were recorded in a busy intersection, near a school and a university, capturing plenty of pedestrians passing by, as shown in Figure 5.

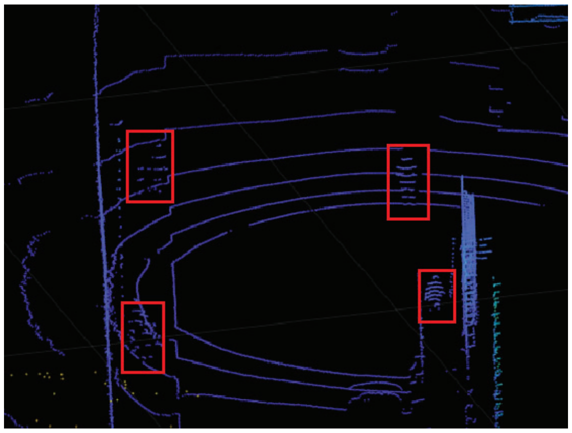
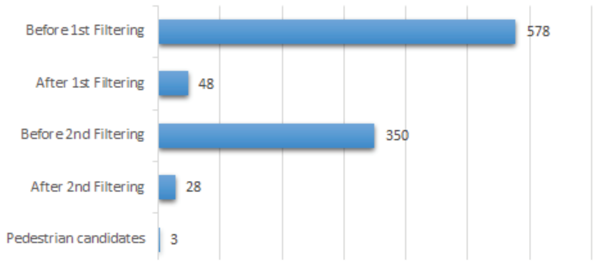


Fig. 5: LiDAR recording with four pedestrians in the frame.

Furthermore, to obtain a diverse range of non-pedestrian objects, the Velodyne Puck sample data set [12] has been used, with an additional 5,000 frames.

Out of these consecutive frames, 600 unique frames were extracted to build a data set of different scenarios, with various pedestrians and non-pedestrians.



	Before 1st Filtering	After 1st Filtering	Before 2nd Filtering	After 2nd Filtering	Pedestrian candidates
Clusters	578	48	350	28	3

Fig. 6: Average number of clusters per frame at different filtering stages.

Figure 6 shows the effect of the filtering actions taken in the Pre-processing block, illustrating the mean number of clusters per frame at different stages. On average, 578 clusters were formed according to a minimum Euclidean distance of 0.76 meters, out of which 48 were extracted after the first filtering. After the layer interpolation, the remaining points were clustered again with the Euclidean distance of 0.38 meters, resulting in 350 clusters. Out of the 28 clusters that resulted after the second filtering, an average of 3 pedestrian candidates were extracted per frame.

Figure 7 illustrates the effects of the filtering stages on the point cloud, while Figure 8 shows the effect of the layers interpolation on a couple of pedestrian candidates.

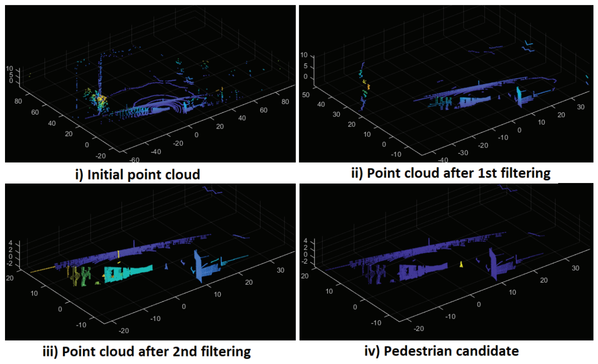
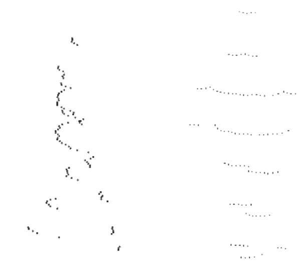
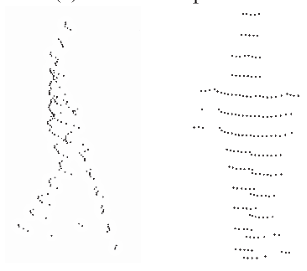

i) Initial point cloud
ii) Point cloud after 1st filtering
iii) Point cloud after 2nd filtering
iv) Pedestrian candidate

Fig. 7: Effects of the filtering stages on the point cloud.



(a) Before interpolation.



(b) After interpolation.

Fig. 8: Effects of the layer interpolation on pedestrian candidates.

After passing the 600 frames through the Pre-processing block, 1,610 pedestrian candidate were obtained. After removing duplicate objects, 1,015 pedestrian candidates remained to train and test the Support Vector Machine, which were distributed as shown in Table II.

TABLE II: Data set for SVM

Total	Pedestrians	Non-pedestrians
1015	518	497

The data set has been used to train and validate a Support Vector Machine with and without RBF, using 5-fold cross-validation. Additionally, the model has been validated using random frames extracted from the original recordings, other than the ones used to train and test the SVM. Table III shows the resulting accuracy of the model with and without the kernel, a comparison between the theoretical detection range and the empirical range, and the expected processing time versus the real processing time.

TABLE III: Results

Description	Results
Accuracy of the model without RBF	91.53%
Accuracy of the model with RBF	95.86%
Theoretical detection range, R	22 m
Experimental detection range	21.54 m
Assumed processing time	1 s
Real processing time	3.1 s

B. Challenges

The main challenge with the proposed method is overcoming the limitations imposed by the low-resolution LiDAR in order to obtain a working algorithm, with a good detection range, and low latency. This work introduces the use of linear interpolation to reduce the requirement of minimum distance between objects for correct identification and extraction. Thus, the algorithm manages to detect singular objects 21.54 meters away from the LiDAR, as long as there is no other object within a 0.38 meters radius. Another way to put it is that the method uses pseudo-layers as a means to connect the distant real layers reflected off an object at the edge of the detection range. However, it is worth mentioning that in some instances this restriction is violated, because the interpolated layers manage to connect layers belonging to two different objects, thus allowing the algorithm to cluster them together. Then, the resulting cluster gets dropped because it does not fit in the bounding box dimensions, which could mean that the model ignores some objects that do not respect the boundary conditions.

Another challenge is minimising the processing time, which could be done by running the model on machines that have better specifications. The algorithm involves performing several operations on large point clouds which can be computationally taxing, so another direction would be optimising the code to minimise the order of complexity. Furthermore, the model was developed in Matlab, using interpreted code, which could be a source of latency. Translating the algorithm into a compiled language would reduce run time.

V. Conclusion

This paper proposed a method for achieving accurate pedestrian detection using a 16 layer LiDAR by compensating for the device's low resolution leveraging simple processing and data manipulation techniques, such as linear interpolation and distance-based clustering, to successfully extract pedestrian candidates under reasonable constraining conditions. Experiments have been run on point clouds captured using a Velodyne Puck LITE LiDAR in a busy intersection, as well as the Velodyne Puck Sample Data, from where diverse pedestrian candidates have been extracted to train, test, and validate a Support Vector Machine classifier, obtaining great accuracy. The theoretical expectations for the detection range of the system have been matched by empirical results, while the algorithm's shortcomings in latency are caused by the lack of processing power of the host machine and the nature of the development environment.

Further work includes experiments with larger data sets, testing the system on high-performance machines, comparing the performance of the proposed solution with others from the industry or literature, and optimising the algorithm to minimise latency and processing requirements.

References

[1] R. Hussain and S. Zeadally, "Autonomous Cars: Research Results, Issues, and Future Challenges," *IEEE Communications Surveys and Tutorials*, vol. 21, no. 2, pp. 1275-1313, Secondquarter 2019.

[2] T. Ogawa, H. Sakai, Y. Suzuki, K. Takagi and K. Morikawa, "Pedestrian detection and tracking using in-vehicle lidar for automotive application," *2011 IEEE Intelligent Vehicles Symposium (IV)*, Baden-Baden, 2011, pp. 734-739.

[3] Z. Lin, M. Hashimoto, K. Takigawa and K. Takahashi, "Vehicle and Pedestrian Recognition Using Multilayer Lidar based on Support Vector Machine," *2018 25th International Conference on Mechatronics and Machine Vision in Practice (M2VIP)*, Stuttgart, 2018, pp. 1-6.

[4] K. Kidono, T. Miyasaka, A. Watanabe, T. Naito and J. Miura, "Pedestrian recognition using high-definition LIDAR," *2011 IEEE Intelligent Vehicles Symposium (IV)*, Baden-Baden, 2011, pp. 405-410.

[5] Y. Kunisada, T. Yamashita and H. Fujiyoshi, "Pedestrian-Detection Method based on 1D-CNN during LiDAR Rotation," *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, Maui, HI, 2018, pp. 2692-2697.

[6] W. Jun and T. Wu, "Camera and lidar fusion for pedestrian detection," *2015 3rd IAPR Asian Conference on Pattern Recognition (ACPR)*, Kuala Lumpur, 2015, pp. 371-375.

[7] Wang Jun, Tao Wu and Zhongyang Zheng, "LIDAR and vision based pedestrian detection and tracking system," *2015 IEEE International Conference on Progress in Informatics and Computing (PIC)*, Nanjing, 2015, pp. 118-122.

[8] G. Melotti, A. Asvadi and C. Premebida, "CNN-LIDAR pedestrian classification: combining range and reflectance data," *2018 IEEE International Conference on Vehicular Electronics and Safety (ICVES)*, Madrid, 2018, pp. 1-6.

[9] T. Yamamoto, Y. Kawanishi, I. Ide, H. Murase, F. Shinmura and D. Deguchi, "Efficient pedestrian scanning by active scan LIDAR," *2018 International Workshop on Advanced Image Technology (IWAIT)*, Chiang Mai, 2018, pp. 1-4.

[10] N. Kudarauskas, "Analysis of emergency braking of a vehicle", *Transport*, 2007, vol. 22.3, pp. 154-159.

[11] R.S. Jurecki, T. Stanczyk, "Analyzing driver response times for pedestrian intrusions in crash-imminent situations", *2018 XI International Science-Technical Conference Automotive Safety. IEEE*, 2018, pp. 1-7.

[12] "Sample data for VeloView (Puck)", Velodyne LiDAR, Accessed on 24th of February, 2020 [Online]. Available: https://velodynelidar.com/downloads/