Can We Detect Pedestrians using Low-resolution LIDAR?
Integration of Multi-frame Point-clouds

Yoshiki Tatebe[1], Daisuke Deguchi[2], Yasutomo Kawanishi[1], Ichiro Ide[1],
Hiroshi Murase[1] and Utsushi Sakai[3]

[1]*Graduate School of Information Science, Nagoya University, Furo-cho, Chikusa-ku, Nagoya-shi, Aichi, Japan*
[2]*Information Strategy Office, Nagoya University, Furo-cho, Chikusa-ku, Nagoya-shi, Aichi, Japan*
[3]*DENSO CORPORATION, 1-1 Showa-cho, Kariya-shi, Aichi, Japan*
tatebey@murase.m.is.nagoya-u.ac.jp, ddeguchi@nagoya-u.jp, {kawanishi, ide ,murase}@is.nagoya-u.ac.jp,
utsushi_sakai@denso.co.jp

Keywords: LIDAR, Pedestrian Detection, Low-resolution.

Abstract: In recent years, demand for pedestrian detection using inexpensive low-resolution LIDAR (LIght Detection And Ranging) is increasing, as it can be used to prevent traffic accidents involving pedestrians. However, it is difficult to detect pedestrians from a low-resolution (sparse) point-cloud obtained by a low-resolution LIDAR. In this paper, we propose multi-frame features calculated by integrating point-clouds over multiple frames for increasing the point-cloud resolution, and extracting their temporal changes. By combining these features, the accuracy of the pedestrian detection from low-resolution point-clouds can be improved. We conducted experiments using LIDAR data obtained in actual traffic environments. Experimental results showed that the proposed method could detect pedestrians accurately from low-resolution LIDAR data.

1 INTRODUCTION

Since 2007, the number of road traffic deaths has not decreased (WHO,), illustrating the needs for further countermeasures against traffic accidents. Furthermore, road traffic deaths among pedestrians are still high, accounting for 22% of the total. In addition, the demand for autopilot system for consumer vehicles has grown considerably in the last decade (Shroff et al., 2013). From these reasons, a system that can recognize the surrounding environment of a vehicle and warn the driver about pedestrians in danger is in strong demand.

In the past few years, pedestrian detection systems have been actively developed. Among such researches, various methods that employ LIght Detection And Ranging (LIDAR) as an in-vehicle sensor have been proposed (Arras et al., 2007; Premebida et al., 2009; Navarro-Serment et al., 2010; Spinello et al., 2011; Kidono et al., 2011; Maturana and Scherer, 2015). Some researchers studied pedestrian detection using LIDAR, and various methods have been proposed. For example, using state-of-the-art Deep Learning, Maturana and Scherer proposed VoxNet (Maturana and Scherer, 2015). They used a three-dimensional convolutional neural network for real-time object recognition. It uses an occupancy grid constructed by three models as input because it can distinguish free space and unknown space. However, VoxNet requires a dense point-cloud that is obtained by the expensive Velodyne LIDAR[1]. For this reason, it cannot be applied to pedestrian detection using cheap low-resolution LIDAR that can be utilized in consumer vehicles.

The low-resolution LIDAR used in our proposed method can scan objects by six horizontal scanning lasers. Therefore, that sensor gives very low vertical resolution, while it is much cheaper than the Velodyne LIDAR. In addition, the number of lasers hitting an object changes due to the distance between the LIDAR and the object. If the object is distant from the LIDAR, only four or less scanning lasers can hit it. In other words, the vertical resolution of the point-cloud is four or less, which is not sufficient to represent the characteristics of pedestrians. For this reason, pedestrian detection using low-resolution LIDAR becomes very difficult, and the conventional methods using the Velodyne LIDAR cannot be applied. For low-resolution point-clouds, Kidono et al. proposed the slice feature and features related to the distribu-

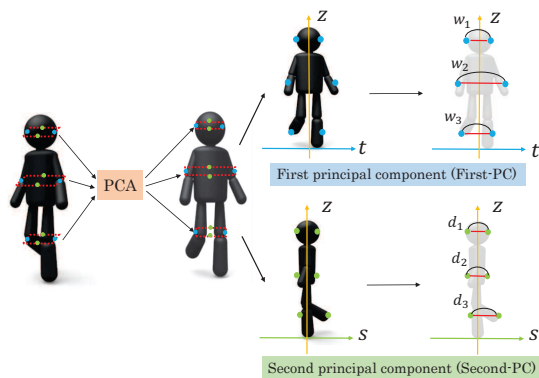---

[1]http://velodynelidar.com/products.html

157

Figure 1: Extraction of the slice feature.

tion of reflection intensity (Kidono et al., 2011). The former represents the rough shape of a pedestrian, and the latter represent the differences in reflection characteristics between a pedestrian and other objects. Fig. 1 shows the procedure to extract the slice feature when three horizontal scans hit the object. Three-dimensional point-clouds obtained from LIDAR are divided into slices at multiple heights, and the width and the depth for each slice are calculated to represent the rough shape of a pedestrian. Since each object has its own distribution of reflection intensity, features related to the distribution of reflection intensity work well to classify a pedestrian and other objects. By combining these features, the detection accuracy of a distant pedestrian observed as a low-resolution point-cloud can be improved. However, if a pedestrian is distant from the vehicle, the detection rate is still low because its resolution is very low. In particular, it is very difficult to distinguish a pedestrian from non-pedestrians such as poles, trees, road signs, and so on, because their shapes are roughly similar to that of a pedestrian. In addition, all features proposed by Kidono et al. are extracted only from momentary information (a single frame). They do not consider temporal information. Therefore, by using multi-frame point-clouds, the proposed method tries to detect a pedestrian in a low-resolution point-cloud more accurately.

In this paper, we propose a pedestrian detection method which improves the detection rate observed as a low-resolution point-cloud by utilizing temporal information. Our contributions and novelties of this paper are as follows:

1. Density enhancement by integration of multiple point-clouds: By integrating point-clouds obtained from multiple frames, further details of three-dimensional objects can be obtained because the point density increases.

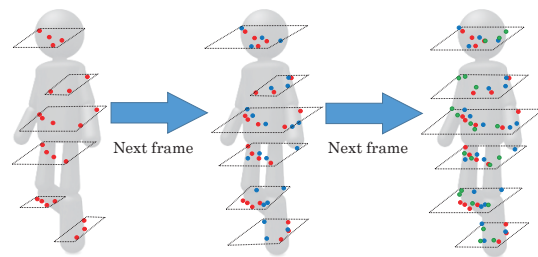2. Consideration of temporal changes of point-

Figure 2: Conceptual diagram of integrating point-clouds.

clouds: By considering temporal changes of point-clouds, objects whose shapes do not change (for example poles, trees, road signs, and so on), can be distinguished.

Section 2 describes the proposed feature extraction method based on the above two ideas, Section 3 describes the proposed pedestrian detection method, evaluation experiments are described in Section 4, and discussions are made in Section 5. We conclude this paper in Section 6.

2 FEATURE EXTRACTION USING TEMPORAL INFORMATION

In this paper, we propose two ideas in order to make use of the temporal information of LIDAR data.

The first idea is to increase the density of point-clouds by integrating multiple frames. In general, the laser beam of a LIDAR hits different positions in each frame even if scanning the same object because of relative movement between the LIDAR and objects. For this reason, more detailed features can be obtained by alignment of multi-frame point-clouds, as shown in Figure 2. By using integrated multi-frame point-clouds for feature extraction, it is expected to allow obtaining more detailed features.

The second idea is to make use of temporal changes of features extracted from LIDAR data. In general, the vehicle will gradually approach objects because the velocity of the vehicle equipped with LIDAR is usually faster than that of a pedestrian. Moreover, the density of three-dimensional point-clouds obtained from LIDAR increases as objects become closer. That is, the density of a newly obtained frame could be higher than that of the older one. The scanning angle also changes due to relative movement of the vehicle and the objects. For this reason, different information about the object can be obtained from each frame.

Applying feature extraction based on the above

two ideas, a pedestrian detection method using a low resolution LIDAR is realized, taking temporal information of LIDAR data into consideration. The following sections 2.1 and 2.2 describe the details for extracting features from multi-frame point-clouds.

2.1 Feature Extraction using Multi-frame Information

In this paper, L represents the number of horizontal scans hitting an object.

A pedestrian candidate point-clouds sequence is represented hierarchically as follows:

$$\mathcal{P} = \{\boldsymbol{P}(t)\}_{t=1}^{T}, \tag{1}$$
$$\boldsymbol{P}(t) = \{\boldsymbol{p}_l\}_{l=1}^{L},$$
$$\boldsymbol{p}_l = \{p_{l,i}\}_{i=1}^{I},$$

where $\mathcal{P}$ is a candidate point-clouds sequence with a length of T frames, $\boldsymbol{P}(t)$ is the point-cloud of t-th frame, $\boldsymbol{p}_l$ is a point-cloud obtained by a l-th horizontal scan, and $p_{l,i}$ is the i-th distant data of l-th scan with coordinates (x, y, z) obtained from LIDAR. Figure 3 shows the graphical representation of this hierarchical structure. In the proposed method, features are extracted from continuous M frames in the candidate point-clouds sequence $\mathcal{P}$. If $\mathcal{P}$ includes K frames, the feature vectors are extracted $K - M + 1$ times for each kind of feature as

$$\boldsymbol{f}(j) = g\left(\{\boldsymbol{P}(k)\}_{k=j}^{j+M-1}\right), \tag{2}$$

where $\boldsymbol{f}(j)$ is the j-th feature vector extracted from $\mathcal{P}$ using M frames, and $g\left(\{\boldsymbol{P}(k)\}_{k=j}^{j+M-1}\right)$ is a feature extraction function from multiple frames. Finally, $K - M + 1$ feature vectors $\boldsymbol{F} = \{\boldsymbol{f}(j)\}_{j=1}^{K-M+1}$ are obtained from $\mathcal{P}$.

Features extracted from multi-frame point-clouds are hereinafter referred to as "multi-frame features", and those extracted from each single-frame point-cloud are referred to as "single-frame features". There are two benefits derived from extracting features from multi-frame point-clouds: (1) the density of point-clouds can be increased, and (2) the changes of the data between frames can be taken into account.

2.2 Multi-frame Features

In this section, a method for extracting multi-frame features from candidate point-clouds sequences are described. Before extracting each feature, pre-processing is performed. First, M-frame candidate point-clouds are projected onto the road surface, and Principal Component Analysis (PCA) is applied to
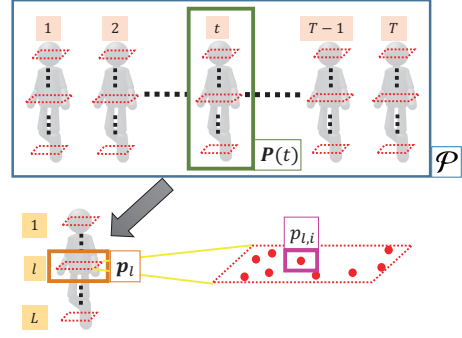

Figure 3: Structure of the candidate point-clouds sequence.

Table 1: Multi-frame features and their dimensions.

Feature's name		Dim.
f_1	Approximated volume of a point-cloud	L
f_2	Maximum value of normalized reflection intensity voxel	$4L$
f_3	Mean of normalized reflection intensity voxel	$4L$
f_4	Maximum value of reflection intensity voxel	$4L$
f_5	Weighted mean of slice feature	$2L$
f_6	Weighted mean of relative slice position	$4(L-1)$
f_7	Weighted mean of maximum value of normalized reflection intensity	1
f_8	Weighted mean of variance of normalized reflection intensity	1

them in each frame. Next, the center-of-gravity of each point-cloud is aligned to share the origin of the axes composed of two principal components obtained by PCA (the first principal component x represents the width of a point-cloud, and the second principal component y represents the depth of a point-cloud). By this pre-processing, the orientation and the position of the each point-cloud are normalized.

Next, the method for extracting multi-frame features from the normalized point-clouds will be described. Table 1 shows the list of features used in the proposed method and the number of dimensions of each feature when the candidate point-cloud is hit by L horizontal scans.

Reflection Intensity. The reflection intensity that appears in Table 1 is the maximum value of the waveform representing the intensity of the reflected light at each point obtained from LIDAR. The reflected light is attenuated in inverse proportion to the square of the distance to each point. Because of this, the reflection intensity value is preferably normalized by multiplying by the square of distance. This normalized value
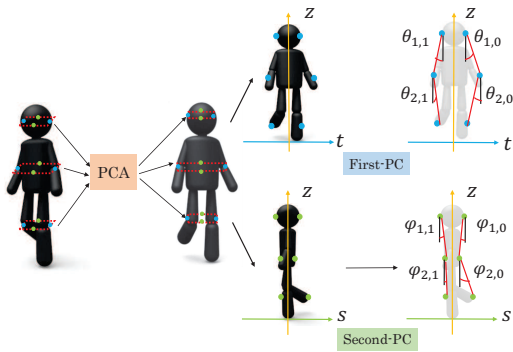
Figure 4: Extraction of the relative slice position ($L = 3$).
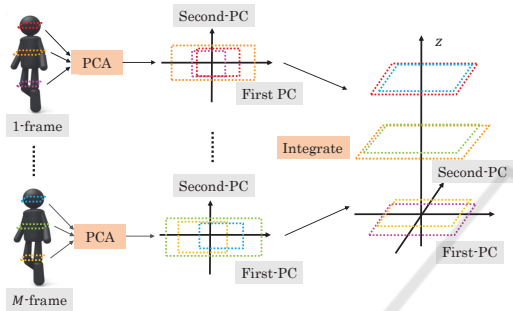
Figure 5: Integration of the point-cloud ($L = 3$).

is called normalized reflection intensity.

Relative Slice Position. The relative slice position that appears in Table 1 represents the outline of a candidate point-cloud. Figure 4 shows the procedure to extract the relative slice position when $L = 3$. First, after calculating a slice feature (Kidono et al., 2011), the end points of the slices of each point-cloud p_l ($l = 1, 2, \cdots, L$) are connected in a straight line. Then, angles of the straight lines with respect to the vertical direction are obtained as the slice position. In other words, by capturing relative position between slices, the relative slice position represents the rough outline of a pedestrian and other objects.

In this paper, features inspired by the first idea of integration of point-clouds to increase their densities ($f_1 \sim f_4$), and the second idea of temporal change of features ($f_5 \sim f_8$) are explored. The calculation procedure for each feature is described in the following sections.

2.2.1 Integration of Point-clouds $f_1 \sim f_4$

In this section, multi-frame features extracted after increasing the density of point-clouds by integrating them based on the first idea are described. Here, the procedure of integrating point-clouds from M frames

for multi-frame feature extraction is described. Figure 5 shows the procedure integrating point-clouds from M frames when $L = 3$. Here, M point-clouds go through pre-processing before being integrated by overlapping them along their coordinate axes.

Two kinds of features are extracted from the point-clouds obtained by the above integration procedure. Each feature proposed here can use more points for feature extraction than a single-frame feature such as slice feature which only uses four points in a point-cloud p_l, because these features are extracted by using point-clouds after their densities are increased.

- Approximated volume of a point-cloud: f_1
- Feature voxel: $f_2 \sim f_4$

However, these features can also be extracted from a single-frame point-cloud. For convenience, the following explanation assumes that the extraction was performed from a single-frame point-cloud.

Approximated Volume of a Point-Cloud (f_1). Approximated volume of a point-cloud represents the shape of point-clouds by a relative ratio of the points constituting each point-cloud p_l of an object. The extraction procedure for this feature is as follows. First, c_l ($l = 1, 2, \cdots, L$), that is the number of points constituting a point-cloud p_l, are calculated. Next, this number of points is normalized by the number of all points from the input point-cloud, and the normalized value is referred to as α_l ($l = 1, 2, \cdots, L$). The entire process is formulated as

$$\alpha_l = \frac{c_l}{\sum_{l=1}^{L} c_l}. \tag{3}$$

Finally, the vector of normalized points is obtained by concatenating all the normalized values. The number of dimensions of the feature vector will vary depending on the number of scan hits L of the input point-cloud. One feature is extracted from the point-cloud p_l, and the feature vector f_1 which is L-dimensions, is obtained as

$$f_1 = \{\alpha_1, \cdots, \alpha_l, \cdots, \alpha_L\}.$$

Feature Voxel ($f_2 \sim f_4$). The feature voxel is a feature obtained by extracting a single-frame feature such as the maximum value, or the mean of normalized reflection intensity, or the maximum value of reflection intensity from divided regions of point-clouds. The following describes the procedure of extracting the feature voxel. First, point-clouds are divided into sub-regions as shown in Figure 6.

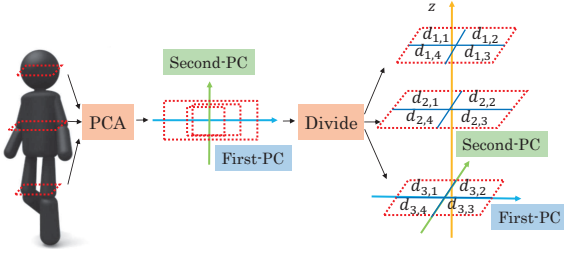Along coordinate axes obtained by preprocessing, a point-cloud is divided into four

Figure 6: Division of a point-cloud for the extraction of feature voxel ($L = 3$).

sub-regions at the center-of-gravity of each point-cloud $\boldsymbol{p}_l$. By concatenating features extracted from sub-regions, a feature vector is obtained. The number of dimensions of the feature vector will vary depending on the number of scan hits L of the input point-cloud. Features $(\beta_{l,1}, \beta_{l,2}, \beta_{l,3}, \beta_{l,4})$ are extracted from four sub-regions $(d_{l,1}, d_{l,2}, d_{l,3}, d_{l,4})$ of the $\boldsymbol{p}_l$ ($l = 1, 2, \cdots, L$). Here, $\beta_{l,1}, \beta_{l,2}, \beta_{l,3}, \beta_{l,4}$ are conventional features extracted from four sub-regions of the $\boldsymbol{p}_l$. Thus, $4L$ dimensions feature vectors $\boldsymbol{f}_2 \sim \boldsymbol{f}_4$ are obtained as

$$\boldsymbol{f}_i = \{\beta_{1,1}, \beta_{1,2}, \beta_{1,3}, \beta_{1,4}, \cdots, \beta_{l,1}, \beta_{l,2}, \beta_{l,3}, \beta_{l,4},$$
$$\cdots, \beta_{L,1}, \beta_{L,2}, \beta_{L,3}, \beta_{L,4}\} \ (i = 2, 3, 4).$$

2.2.2 Temporal Changes of Point-clouds $\boldsymbol{f}_5 \sim \boldsymbol{f}_8$

In this section, to utilize the second idea of the temporal changes of point-clouds, $\boldsymbol{f}_5 \sim \boldsymbol{f}_8$ are extracted by calculating the weighted mean of some single-frame features (slice feature, relative slice position, maximum value of the normalized reflection intensity, and variance of the normalized reflection intensity) extracted from continuous M frames of $\mathcal{P}$ so that the newer frame's feature has a higher weight. The weighted mean of continuous point-clouds from M frames is calculated by a recurrence relation:

$$\overline{\boldsymbol{s}(k)} = \begin{cases} \boldsymbol{s}(1) & (k = 1), \\ 0.4\overline{\boldsymbol{s}(k-1)} + 0.6\boldsymbol{s}(k) & (k = 2, 3, \cdots, M), \end{cases}$$

where $\boldsymbol{s}(k)$ is a single-frame feature extracted from the k-th frame of $\mathcal{P}$, and $\overline{\boldsymbol{s}(k)}$ is the weighted mean calculated from 1 to the k-th frames. For example, a weighted mean $\overline{\boldsymbol{s}(2)}$ is calculated by using $\boldsymbol{s}(1)$ and $\boldsymbol{s}(2)$, and a weighted mean $\overline{\boldsymbol{s}(3)}$ is calculated by using $\overline{\boldsymbol{s}(2)}$ and $\boldsymbol{s}(3)$. This calculation is repeated until $\overline{\boldsymbol{s}(M)}$, and it becomes multi-frame feature vectors $\boldsymbol{f}_5 \sim \boldsymbol{f}_8$. The dimensionality of the final feature vector is equal to that of the original single-frame features, as shown in Table 1.
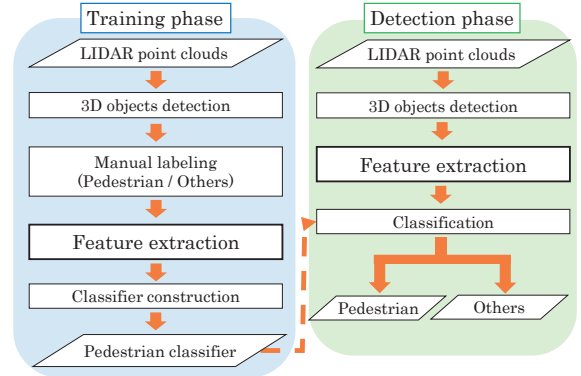

Figure 7: Process flow of the proposed method.

3 PEDESTRIAN DETECTION USING A LOW-RESOLUTION LIDAR

Figure 7 shows the process flow of the proposed method. The proposed method is the training phase for building a classifier to be used for pedestrian detection in advance, and the detection phase for detecting pedestrians from the LIDAR data using the classifier constructed in the training phase.

3.1 Training Phase

3.1.1 3D Object Detection and Manual Labeling

The proposed method extracts point-clouds of three-dimensional objects from LIDAR data by applying a three-dimensional object detection and tracking algorithm (Ogawa et al., 2011) for LIDAR data collected in real-world environments. In this paper, the point-clouds obtained by three-dimensional object detection are regarded as the candidate point-clouds and a series of them are regarded as the candidate point-clouds sequence. The candidate point-clouds contained pedestrians, trees, poles, traffic signs, and so on. These point-clouds are labeled as "pedestrian" or "other objects" manually.

3.1.2 Feature Extraction

The proposed method extracts features related to the shape and the reflection intensity of three-dimensional objects described in Section 2.2 from successive multi-frame point-clouds in a series of point-clouds obtained by tracking three-dimensional objects. The proposed feature extraction method is based on two ideas using the temporal information of multi-frame point-clouds.

3.1.3 Classifier Construction

In the proposed method, a pedestrian classifier is constructed by inputting the features calculated from point-clouds of pedestrians (positive samples) and other objects (negative samples) to kernel-SVM, which generally shows high performance. Here, the RBF (Radial Basis Function) kernel is used.

3.2 Detection Phase

3.2.1 3D Object Detection

Candidate point-clouds sequences are obtained by applying the same three-dimensional object detection and tracking algorithm (Ogawa et al., 2011) on point-clouds obtained from LIDAR.

3.2.2 Feature Extraction

The same features as the training phase are extracted from candidate point-clouds sequences.

3.2.3 Classification

By inputting these features to the classifier constructed in the training phase, the candidate point-clouds are classified as either a pedestrian or not. This output is the detection result of the proposed method.

4 EXPERIMENTS

To confirm the effectiveness of the proposed method against pedestrian detection using a low-resolution LIDAR, experiments using in-vehicle low-resolution LIDAR point-clouds taken in real-world environments were conducted. The low-resolution LIDAR used in the experiments was equipped on the top of the room mirror. The LIDAR could obtain depth data at 6 vertical directions and 401 horizontal directions at 10 fps. The vertical detection angle was 6 degrees (1 degree pitch), and the horizontal detection angle was 40 degrees (0.1 degree pitch).

4.1 Experimental Procedure

In this experiment, point-clouds of pedestrians and other objects were collected by the low-resolution LIDAR in real-world environments. By applying the three-dimensional object detection and tracking algorithm (Ogawa et al., 2011) to the collected data, the candidate point-clouds sequences were obtained.

Table 2: Number of point-clouds for each number of scan hits L.

	Number of samples by L			
	$L = 3$	$L = 4$	$L = 5$	$L = 6$
Pedestrians	1,002	5,269	6,934	8,189
Others	5,726	5,904	7,236	13,535

Table 3: Single-frame features and their dimensions.

Feature's name		Dim.
f_9	Slice feature	$2L$
f_{10}	Relative slice position	$4(L-1)$
f_{11}	Maximum value of normalized reflection intensity	1
f_{12}	Mean of normalized reflection intensity	1
f_{13}	Variance of normalized reflection intensity	1
f_{14}	Maximum value of reflection intensity	1

All positive samples and hard negative samples (pole, tree and so on) extracted from these sequences were input of the proposed method. Table 2 shows the result of aggregating the data by each number of scan hits L used in this experiment. Dividing the candidate point-clouds sequences by L, the performance of the proposed method was evaluated by five-fold cross-validation. To ensure fair comparison, the point-clouds obtained from the same object were not included in both training samples and test samples.

We evaluated the proposed method by using ROC (Receiver Operating Characteristic) curve and its AUC (Area Under the Curve). In this experiment, three methods, that is the proposed method 1, the proposed method 2 and the comparison method, were evaluated. The proposed method 1 used the multi-frame feature $f_1 \sim f_4$ extracted from three frames ($M = 3$) for pedestrian classification, and the proposed method 2 used the multi-frame feature $f_1 \sim f_8$ extracted from three frames ($M = 3$) for pedestrian classification. This method detected pedestrians every three frames because three frames were required for the multi-frame feature extraction. For the comparison method, the average of three ROC curves and AUCs obtained in each of the three frames were used for the evaluation. In the comparison method, conventional single-frame features were used for the classification (Kidono et al., 2011). Table 3 shows the list of single-frame features, and Table 4 shows the features used in each method. We implemented feature extraction and classification methods using MATLAB, and used a conventional PC (Intel Core i7 4790 3.6 GHz) in the experiment.
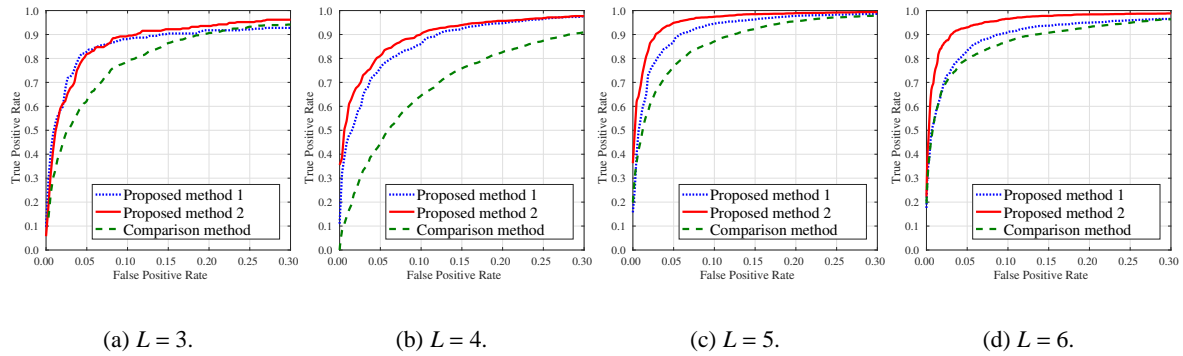
(a) $L = 3$.   (b) $L = 4$.   (c) $L = 5$.   (d) $L = 6$.

Figure 8: ROC curves of each method by the number of scan hits L.

Table 4: Evaluated methods and features.

	Prop. 1	Prop. 2	Comp.
$f_1 \sim f_4$	✓	✓	
$f_5 \sim f_8$		✓	
$f_9 \sim f_{14}$			✓

Table 5: AUC of each method.

Method	AUC by L			
	$L = 3$	$L = 4$	$L = 5$	$L = 6$
Proposed 1	0.937	0.953	0.971	0.960
Proposed 2	**0.952**	**0.963**	**0.986**	**0.983**
Comparison	0.926	0.891	0.958	0.954

Table 6: Rank of greedy algorithm.

Order	Rank by L			
	$L = 3$	$L = 4$	$L = 5$	$L = 6$
1	f_2	f_7	f_4	f_4
2	f_1	f_1	f_5	f_6
3	f_4	f_2	f_6	f_5
4	f_6	f_4	f_1	f_1
5	f_8	f_3	f_7	f_2
6	f_3	f_5	f_2	f_3
7	f_5	f_6	f_3	f_8
8	f_7	f_8	f_8	f_7

4.2 Results & Discussions

Figures 8(a)–8(d) show the ROC curves of each method obtained in this experiment. When False Positive Rate (FPR) was close to 5%, True Positive Rate (TPR) of the proposed methods 1 and 2 were higher than that of the comparison method for all numbers of scan hits L. Note that the detection rate in low FPR is important for pedestrian detection, where the proposed method outperforms the comparative method. In addition, the proposed method 2 had higher detection accuracy than that of the proposed method 1.

Table 5 shows the AUC obtained in this experiment by each method. The AUC values of the proposed method 2 were higher than those of the proposed method 1 and the comparison method for all L, showing that the proposed method 2 is superior than the two other methods in overall performance.

The computation time of feature extraction and classification for each candidate are 16.4 ms and 1.6 ms, respectively.

4.2.1 Comparison of Detection Accuracy

As shown in Figures 8(a)–8(d), by using a combination of features discussed in this paper, the proposed

method could detect pedestrians more accurately than using conventional single-frame features.

Furthermore, the proposed method 1 using $f_1 \sim f_4$ based on point-clouds density enhancement much improved the pedestrian detection accuracy especially in a very low-resolution condition ($L = 3, 4$). Therefore, idea 1 of integrating point-clouds to increase its density was effective when the point-clouds density was very low. On the other hand, the proposed method 2 could detect pedestrians much more accurately than the proposed method 1 in relatively high-resolution condition ($L = 5, 6$). When $L = 5$ and 6, more detailed shape and reflection intensity of objects could be obtained by LIDAR point-clouds, so $f_5 \sim f_8$ based on the temporal change worked well.

4.2.2 Contribution Analysis of Features

To analyze the contribution of each feature towards the detection accuracy, the ranking of all features was calculated by using the greedy algorithm. Table 6 shows the selection order (rank) of features by the greedy algorithm. As shown in Table 6, the order was different for each number of scan hits L.

Table 7 shows a rearranged version of Table 6 according to the proposed ideas (idea 1: integration of point-clouds to increase their density and idea 2: temporal changes of features). In Table 7, the rank of the features based on idea 1 was relatively higher than that of the features based on idea 2. This trend

Table 7: Rank of greedy algorithm (rearranged according to the fundamental ideas).

Ideas	Feature's name	Rank				Ave.
		$L = 3$	$L = 4$	$L = 5$	$L = 6$	
Density	Maximum value of reflection intensity voxel	3	4	1	1	2.3
	Approximated volume of a point-cloud	2	2	4	4	3.0
	Maximum value of normalized reflection intensity voxel	1	3	6	5	3.8
	Mean of normalized reflection intensity voxel	6	5	7	6	6.0
Change	Weighted mean of relative slice position	4	7	3	2	4.0
	Weighted mean of slice feature	7	6	2	3	4.5
	Weighted mean of maximum value of normalized reflection intensity	8	1	5	8	5.5
	Weighted mean of variance of normalized reflection intensity	5	8	8	7	7.0

suggests that the idea 1 was more effective than the idea 2. This is because it is likely that the movement of pedestrians and vehicles in the range of three frames used for multi-frame feature extraction was very small.

5 CONCLUSIONS

In this paper, we proposed a pedestrian detection method using multi-frame features extracted from low-resolution LIDAR data. We introduced the multi-frame features extracted by combining point-clouds over multiple frames to increase its resolution and capturing temporal changes of the point-clouds. The proposed method detected pedestrians using the classifier trained by inputting the LIDAR data divided by their numbers of scan hits L.

Using the data collected in real-world environments, experiments showed the proposed method using a combination of proposed multi-frame features, could detect pedestrians more accurately than using conventional single-frame features. We also analyzed the contribution of each feature to the performance improvement. The results showed the idea of integrating point-clouds to increase their density was effective for pedestrian detection from low resolution LIDAR.

Future work includes improvement of the proposed method considering the combination of single-frame features and multi-frame features simultaneously, construction of the classifier using partial AUC (Narasimhan and Agarwal, 2013), and comparison of features learned by Deep Learning.

ACKNOWLEDGEMENTS

REFERENCES

World Health Organization. (2015). Global status report on road safety 2015.

Arras, K. O., Mozos, O. M., and Burgard, W. (Apr. 2007). Using boosted features for the detection of people in 2D range data. In *Proc. 2007 IEEE Int. Conf. on Robotics and Automation*, pages 3402–3407.

Kidono, K., Miyasaka, T., Watanabe, A., Naito, T., and Miura, J. (June 2011). Pedestrian recognition using high-definition LIDAR. In *Proc. 2011 IEEE Intelligent Vehicles Symposium*, pages 405–410.

Maturana, D. and Scherer, S. (Sept. 2015). Voxnet: A 3D convolutional neural network for real-time object recognition. In *Proc. 2015 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pages 922–928.

Narasimhan, H. and Agarwal, S. (Aug. 2013). SVM pAUC tight: A new support vector method for optimizing partial AUC based on a tight convex upper bound. In *Proc. 19th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining*, pages 167–175.

Navarro-Serment, L. E., Mertz, C., and Hebert, M. (Oct. 2010). Pedestiran detection and tracking using three-dimensional LADAR data. *Int. J. of Robotics Research, vol.29, no.12*, pages 1516–1528.

Ogawa, T., Sakai, H., Suzuki, Y., Takagi, K., and Morikawa, K. (June 2011). Pedestrian detection and tracking using in-vehicle LIDAR for automotive application. In *Proc. 2011 IEEE Intelligent Vehicles Symposium*, pages 734–739.

Premebida, C., Ludwig, O., and Nunes, U. (Oct. 2009). Exploiting LIDAR-based features on pedestrian detection in urban scenarios. In *Proc. 2009 IEEE Int. Conf. on Intelligent Transportation Systems*, pages 1–6.

Shroff, D., Nangalia, H., Metawala, A., Parulekar, M., and Padte, V. (Jan. 2013). Dynamic matrix and model predictive control for a semi-auto pilot car. In *Proc. 2013 IEEE Int. Conf. on Advances in Technology and Engineering*, pages 1–5.

Spinello, L., Luber, M., and Arras, K. O. (May 2011). Tracking people in 3D using a bottom-up top-down detector. In *Proc. 2011 IEEE Int. Conf. on Robotics and Automation*, pages 1304–1310.