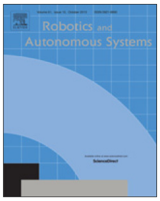
Robotics and Autonomous Systems 88 (2017) 71–78

Contents lists available at ScienceDirect

Robotics and Autonomous Systems

journal homepage: www.elsevier.com/locate/robot



Pedestrian recognition and tracking using 3D LiDAR for autonomous vehicle[☆]

Heng Wang [a,b,*], Bin Wang [b], Bingbing Liu [b], Xiaoli Meng [b], Guanghong Yang [a]

[a] *College of Information Science and Engineering, Northeastern University, Shenyang, China*
[b] *Institute for Infocomm Research, Agency for Science, Technology and Research (A*STAR), Singapore*

HIGHLIGHTS

- Curb information is used to divide pedestrian candidates into two classes, i.e., on-road and out of road.
- Tracking aided recognition strategy is used to improve the true positive rate, for example, for those on-road candidates, who have been recognized as pedestrians in former frame are classified to be pedestrians directly.
- This may increase the true positive rate for candidates who become too close or too far from moving autonomous vehicle in the following frames, where only a few laser beams are irradiated on pedestrians.
- Hash table is used for searching and comparison in the segmentation procedure to increase the efficiency of the proposed algorithm.
- The pedestrian recognition and tracking system is integrated with the autonomous vehicle platform which provides timely prediction of pedestrian motions.

ARTICLE INFO

Article history:
Received 1 December 2015
Accepted 14 November 2016
Available online 22 November 2016

Keywords:
Pedestrian recognition and tracking
Point clouds
Hash table
GUI design

ABSTRACT

This paper studies the pedestrian recognition and tracking problem for autonomous vehicles using a 3D LiDAR, a classifier trained by SVM (Support Vector Machine) is used to recognize pedestrians, the recognition performance is further improved with the aid of tracking results. By comparing positions and velocity directions of pedestrians with curb information, alarms will be generated if pedestrians are detected to be on road or close to curbs. The proposed approach has been verified on an autonomous vehicle platform.

1. Introduction

Research on autonomous vehicles such as driverless cars has received great attention in the past ten years, pedestrian recognition and tracking, as one of the most important issues for autonomous vehicles, also grows exponentially with the development of self-driving techniques. For cases where autonomous vehicles operate in close proximity with pedestrians, recognition and tracking of moving pedestrians are two basic tasks of autonomous vehicle's collision avoidance system. In addition, the timely prediction of most likely near future positions of pedestrians being tracked is essential for the real-time path planning of autonomous vehicles.

In the literature, cameras are commonly used low cost sensors for pedestrian recognition, e.g., by analyzing each image, contour cues are used to detect pedestrians in [1], while in [2], by investigating the characteristic appearance patterns of pedestrians in crowded street, a joint detector is proposed to improve the detection result for pedestrians with partial occlusions. Since humans are highly articulated, it is common to use part-based representations and multi-layer strategy for pedestrian recognition, in [3], pedestrians are modeled as flexible assemblies of parts, a coarse-to-fine cascade approach is used for part detection, and a part assembly strategy is proposed to recognize pedestrians. In [4], a bottom-up parsing of partial body masks strategy is proposed, at each level of the parsing process, partial body masks are evaluated via shape matching with exemplars. In [5], multiscale deformable part models are mixed and a margin sensitive approach for data-mining hard negative examples is combined with latent SVM to detect highly variable objects including pedestrians. In [6], leg

[☆] This work has been supported in part by the National Natural Science Foundation of China (Grant No. 61673098), and in part by the National Natural Science Foundation of China (Grant no. 61273148, 61420106016, 61621004), the IAPI Fundamental Research Funds (Grant No. 2013ZCX01-01).

* Corresponding author at: College of Information Science and Engineering, Northeastern University, Shenyang, China.

E-mail addresses: wangheng@mail.neu.edu.cn (H. Wang), wangb@i2r.a-star.edu.sg (B. Wang), bliu@i2r.a-star.edu.sg (B. Liu), mangx@i2r.a-star.edu.sg (X. Meng), yangguanghong@mail.neu.edu.cn (G. Yang).

http://dx.doi.org/10.1016/j.robot.2016.11.014
0921-8890/© 2016 Elsevier B.V. All rights reserved.

segments are considered to be 'arc-like' shapes, a circle detection technique called inscribed angle variance (IAV) is presented.

LiDARs are another kind of commonly used sensors for pedestrian recognition, compared with cameras, LiDARs can provide accurate range information and larger field of view. In [7], 3D point clouds are seen as a collection of several 2D point clouds at different heights, detectors trained by AdaBoost are used to detect pedestrians. In [8], 3D point clouds in the target are divided into parts, i.e., trunk and legs, to achieve robust *in situ* pedestrian recognition. In [9], both geometric and motion features are used to represent pedestrians, which deal with static and moving pedestrians, respectively. In [10], two new features are introduced to improve the classification performance, one is the slice feature, which represents the profile of a human body by widths at different height levels, the other is the distribution of reflection intensities of point clouds. In [11], AdaBoost is used to learn a robust leg detector from a set of statistic features, such as number of points. The same classification strategy is used in [12], where the possible distance between legs is taken into account, and a probabilistic body part-based detection framework is presented. In [13], an algorithm to combine 3D and 2D LiDARs is proposed, where 3D data is used to construct a ground elevation map while 2D data is used to detect pedestrians at a certain height above the ground. In [14], data collected by a Lidar and a monocular camera is combined, and an active pedestrian detection system is designed, through using sliding window detectors, image-based detection module is used to validate the presence of pedestrians in the regions of interest generated by Lidar module.

On the other hand, reliable tracking and prediction of pedestrian motions can make an autonomous vehicle aware of potential collision with pedestrians in its vicinity [15]. Kalman filters with nearest neighbor data association [16] and Monte Carlo Particle filters [17] are two commonly used approaches to estimate motions of pedestrians, and in [18], feature matching, ICP (Iterative Closest Point), Kalman filtering, and dynamic mapping are combined together to estimate motions. This paper deals with the pedestrian recognition and tracking problem for an autonomous vehicle which is moving on road. The contribution of this paper includes the following aspects. (1) Curb information is used to divide pedestrian candidates into two classes, i.e., on-road and out of road. Tracking aided recognition strategy is used to improve the true positive rate, for example, for those on-road candidates, who have been recognized as pedestrians in former frame are classified to be pedestrians directly. This may increase the true positive rate for candidates who become too close or too far from moving autonomous vehicle in the following frames, where only a few laser beams are irradiated on pedestrians. (2) Hash table is used for searching and comparison in the segmentation procedure to increase the efficiency of the proposed algorithm. (3) The pedestrian recognition and tracking system is integrated with the autonomous vehicle platform which provides timely prediction of pedestrian motions. The rest of this paper is organized as follows. Section 2 introduces the system architecture. Section 3 presents the pedestrian recognition method including how to choose the training examples. Section 4 discusses the pedestrian tracking approach, based on which alarms are also generated when collision dangers are predicted. Experiments are presented in Section 5, while conclusions are given in Section 6.

2. System overview

2.1. System introduction

The main sensor used in this paper is Velodyne 64 LiDAR (Light Detection And Ranging) which is mounted on the top of an autonomous vehicle as shown in Fig. 1. Besides Velodyne 64, GPS (Global Positioning System), IMU (Inertial Measurement Unit) and DMI (Distance Measurement Unit) are fused to localize the autonomous vehicle in real time. Since we mainly focus on pedestrian recognition and tracking problem in this paper, it is assumed that the autonomous vehicle is localized with enough accuracy. In addition, a Velodyne 16 Lidar is used to detect and fit curbs, which is mounted in the front of the vehicle. The other sensors mounted on the autonomous vehicle are used for other purposes which are ignored.

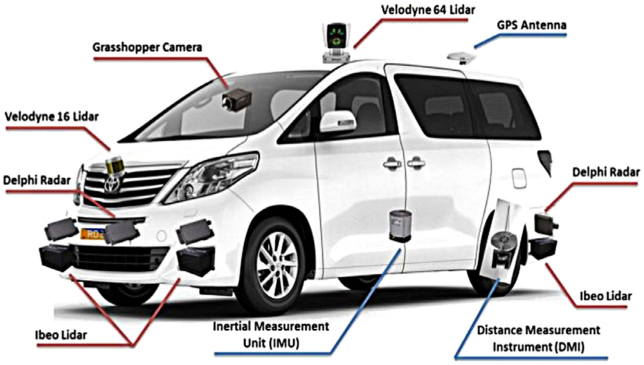


Fig. 1. Autonomous vehicle platform of Institute for Infocomm Research.

The Velodyne 64 incorporates 64 laser diodes and spins at up to 15 Hz, generates dense range data covering a 360-deg horizontal field of view and a 26.8-deg vertical field of view. A 3D point cloud is acquired from Velodyne: around 100,000 points per frame at frequency of 10 Hz, each point includes range, intensity and *xyz* coordinates. Similar to Velodyne 64, Velodyne 16 incorporates 16 laser diodes whose size is much smaller than Velodyne 64 and can be mounted more flexibly. In this paper, we use the raw data collected by Velodyne 64 and Velodyne 16 to recognize pedestrian and detect curbs, respectively. All the *xyz* coordinates of point clouds collected by both Velodyne 64 and Velodyne 16 are converted into the same global coordinate frame.

In the following sections, we will describe pedestrian recognition and tracking using the data collected by Velodyne 64 in details. 3D point clouds are firstly projected onto 2D plane and are gridded, then a nearest-distance algorithm is proposed for clustering, Kalman filter is used to track objects whose sizes are under a limit after clustering. At the same time, a trained classifier is used to recognize whether the tracked objects are pedestrians or not. In addition, pedestrian recognition performance is further improved by tracking results.

3. Pedestrian recognition

3.1. Initial processing

For further clustering and recognition, 3D point clouds acquired from Velodyne 64 are firstly processed to filter noises and uninterested areas, e.g., points whose heights are larger than some limit are removed. All the 3D points are projected onto 2D occupancy grid, Fig. 2 shows a schematic diagram of a 5 × 5 occupancy grid map, where grid cells (2,3), (3,2), (3,3) and (5,5) are occupied, and grid cells (2,3), (3,2) and (3,3) are called connected cells. The cell size of each occupancy grid is set to be 0.1 m × 0.1 m, a threshold η_1 is chosen to limit the height difference in each grid which is set to be 0.4 m in this paper, by comparing the average height in each grid with η_1, grids corresponding to ground and low objects such as curbs are filtered. Since ground and low objects are removed from occupancy grid, the occupancy grid can be represented by a sparse matrix where non-zero elements are randomly distributed.

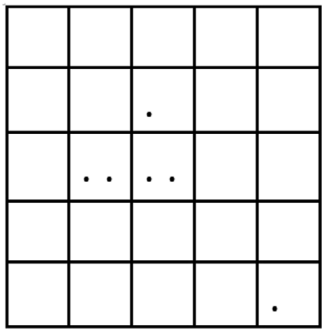


Fig. 2. Occupancy grid map.

In order to search each element efficiently, we use sparse *hash table* [19] instead of compressed row storage format. The hash-based representation is efficient for the particular operation of assembling and updating elements of sparse matrices, for which the compressed row storage format is inefficient. We represent the occupancy grid as a sparse matrix where elements are randomly distributed. The sparse matrix representation is a coordinate storage format where non-zero elements are stored in a hash table, e.g., if an element (x, y) is in a two-dimensional sparse matrix with r rows and c columns, x ∈ $[0, r-1]$, y ∈ $[0, c-1]$, the value v is then stored in a hash table with key $k = x * c + y$. The hash function $h(k)$ is simply the value k mod s, s is chosen to be a prime number such that non-zero elements can be mapped to different hash values.

3.2. Clustering

Through quick searching the hash table, all the connected cells are clustered into one blob, after which blobs are merged into one blob if the distance between two blobs are within another threshold η_2 which is set to be 0.3 m in this paper. Then, pedestrian candidates are extracted if the following conditions are satisfied:

- Height: $0.5\text{ m} \leq h \leq 2.0\text{ m}$
- Width: $w \leq 2$
- Length: $\ell \leq 2$

Note that η_2, together with η_1 are both chosen beforehand through experiences, and η_2 is used for clustering 3D points that belong to the same pedestrian.

3.3. Classification

For each pedestrian candidate, a feature vector is computed and evaluated to classify whether it is a pedestrian or not. SVM [20–22] is used with radial basis function (RBF) kernel to learn the classifier. Different kinds of features have been proposed for training pedestrian classifiers in the literature [10,23]. In this paper, the following Features I–IV are used which have been verified to be efficient.

In particular, Feature I computes statistical descriptions from points located around legs and trunk area, which describes the distinctive human figure. Feature II and Feature III describe the general shape of human figure by computing the normalized 2D histograms on two planes aligned with the gravity vector. Feature IV describes the geometric sizes in different height of a human by dividing the 3D points into N blocks along the principal eigenvector, which can discriminate false positives such as trees and poles.

3.3.1. Feature extraction

After segmentation, we obtain a set of candidate objects. Let

$$S_k = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_{N_k}\}, k = 1, \ldots, n$$

be the set of points belonging to object k whose elements are represented by Cartesian coordinate (x, y, z). Denote the feature vector as f, which is

$$f = \{f_1, f_2, \ldots, f_m\}$$

where m is the dimension of f, in this paper, $m = 195$ by summing all the features. In the following, we will introduce how to calculate this feature vector.

Feature I: Feature I is formed by $f_1 \sim f_9$ which are obtained from the 2D covariance matrix in 3 zones, which are upper half, left and right lower halves.

Calculate eigenvalue $(\lambda_1, \lambda_2, \lambda_3)$ where $\lambda_1 \geq \lambda_2 \geq \lambda_3$ and corresponding eigenvector (v_1, v_2, v_3) of the covariance matrix, then we transform the original data into two representations using pair of components v_1, v_2, from which we divide the data into three parts, and calculate 3 corresponding covariances, for each covariance matrix, we only use the (1, 1), (1, 2), (2, 2), elements as feature.

Each covariance matrix can be calculated as

$$\Sigma_k = \frac{1}{N_k - 1}\sum_{i=1}^{N_k}(\mathbf{x}_i - \bar{\mathbf{x}})(\mathbf{x}_i - \bar{\mathbf{x}})^T \tag{1}$$

where $\bar{\mathbf{x}}$ is the mean.

Feature II and III: Feature II and III are formed by $f_{10} \sim f_{121}$ and $f_{122} \sim f_{175}$ respectively which are obtained by the normalized 2D histogram for x–z plane and y–z plane, where the positive direction of z is vertical up.

- 2D histogram for x–z plane: $14 \times 8, f_{10} \sim f_{121}$
- 2D histogram for y–z plane: $9 \times 6, f_{122} \sim f_{175}$.

We accumulate element (p, q) of both the 14×8 and 9×6 grids by checking each point using the following formula, take x–z plane for example:

$$p = floor\left(\frac{x_i - x_{min}}{x_{max} - x_{min}} \times 8\right) \tag{2}$$

$$q = floor\left(\frac{z_i - z_{min}}{z_{max} - z_{min}} \times 14\right) \tag{3}$$

where $floor(\xi)$ returns the largest integral value that is not greater than ξ, x_{min}, x_{max}, z_{min}, z_{max} are the minimum and maximum of x, z, respectively. In order to normalize the histogram, we divide both histograms by the total number of points.

Feature IV: Feature IV is formed by $f_{176} \sim f_{195}$ which are obtained from the slice feature for cluster S_k. 3D points in the cluster are divided into 10 blocks of the same size along z axis, 3D points in each block are projected onto a plane orthogonal to z axis, and two widths along x and y axis are calculated as features.

In this paper, we consider the real urban environment, false positives mainly include trees, light poles, traffic signs, building facades etc. The difference between human shapes and these shapes are the pattern of legs and the profile from the head to the shoulder. Features I–IV can provide better discrimination of pedestrians from non-pedestrians, the effectiveness will be verified in Section 5.

3.3.2. Training process

SVM algorithms are based on the search in parameter space of the separating hyperplane with maximal distance of the nearest learning elements, called support vectors. Moreover, to allow

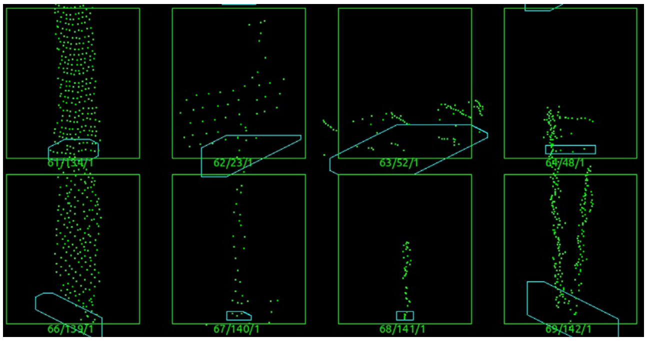


Fig. 3. Some negative training examples, including trees, traffic signs and light poles.

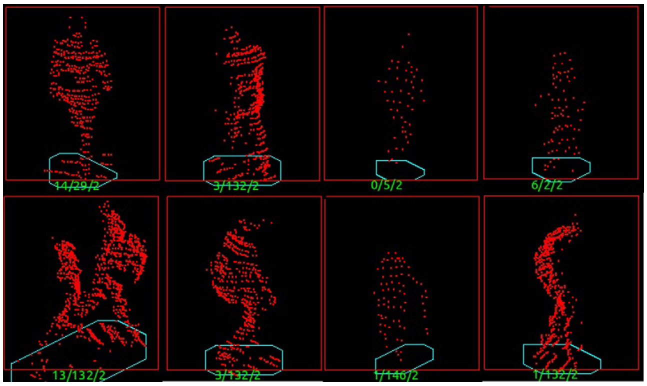


Fig. 4. Some positive training examples, including pedestrians with or without bags, walking side by side. The upper left two pedestrians are within 20 m, while the distance from the upper right two pedestrians are larger than 20 m.

separation, data are embedded in a higher dimension space by a nonlinear transform. The decision function is computed by [24]:

$$\mathcal{H}(f) = \sum_{i=1}^{l} \alpha_i^0 y_i K(f_i, f) + b \tag{4}$$

where l is the number of training examples, y_i is 1 or -1 whether the example is or is not an object of the class, and $K(\cdot, \cdot)$ is a kernel functional.

Before training a classifier, appropriate positive and negative training datasets need to be selected firstly. Since we only consider outdoor urban environment, negative examples are selected from the vegetation, light poles, traffic signs, parts of building etc., which are mostly encountered by autonomous vehicles, some samples are presented in Fig. 3. Considering that pedestrians are usually wearing hat, taking bags, walking side by side, standing or jogging etc., positive examples are selected from the real data collected outdoor, see Fig. 4 for some examples.

In order to label both positive and negative examples efficiently, a GUI is designed, see Fig. 5. By the aid of this GUI, negative and positive examples are labeled automatically, both negative and positive examples are shown in 360-degree-view, in addition, the labeled positive examples can be changed into negative easily through double clicking the rectangles. The overview of the labeling and training process is presented by a flowchart as shown in Fig. 6. Assume that the raw dataset is divided into p groups, each of which contains N_i frames of data, the total number of frames is $n = \sum_{i=1}^{p} N_i$. For each frame f_i, after segmentation, all the pedestrian candidates are labeled as positive or negative by a classifier. By using the GUI as shown in Fig. 5, inappropriate ones are dropped/exchanged from the labeled examples, then the training dataset is updated and a new classifier will be trained if the condition $f_i \geq N_i$ is satisfied. The above procedure is repeated until the performance evaluation of the new classifier satisfies a given performance index.

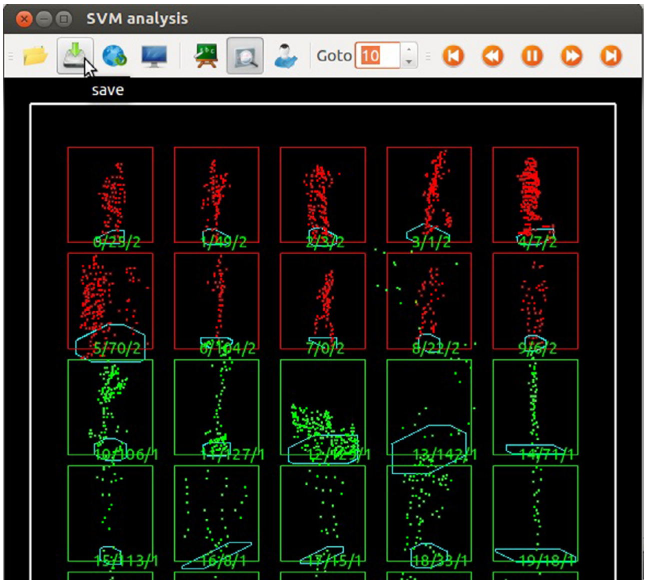


Fig. 5. GUI for labeling and training.

4. Pedestrian tracking

After recognition, the next task is to track the recognized pedestrians. Assume that pedestrian position is (x, y), velocity of pedestrian is constant, the kinematics is

$$\zeta(t) = F\zeta(t-1) + G\Delta\zeta(t-1) \tag{5}$$

where

$$F = \begin{bmatrix} 1 & \tau & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & \tau \\ 0 & 0 & 0 & 1 \end{bmatrix}, G = \begin{bmatrix} \dfrac{\tau^2}{2} & 0 \\ \tau & 0 \\ 0 & \dfrac{\tau^2}{2} \\ 0 & \tau \end{bmatrix}$$

$\zeta = \begin{bmatrix} x & \dot{x} & y & \dot{y} \end{bmatrix}^T$, and $\Delta\zeta = \begin{bmatrix} \Delta\ddot{x} & \Delta\ddot{y} \end{bmatrix}^T$ is an unknown acceleration, τ is sampling period.

The measurement model related to the pedestrian is

$$z(t) = H\zeta(t) + Lu(t) + \omega(t) \tag{6}$$

where

$$H = \begin{bmatrix} \cos\phi(t) & 0 & -\sin\phi(t) & 0 \\ \sin\phi(t) & 0 & \cos\phi(t) & 0 \end{bmatrix}, L = \begin{bmatrix} \cos\phi(t) & -\sin\phi(t) \\ \sin\phi(t) & \cos\phi(t) \end{bmatrix}$$

and $z = \begin{bmatrix} z_x & z_y \end{bmatrix}^T$ is the measurement, $\omega(t)$ is the measurement noise. In addition, $u = \begin{bmatrix} x_v & y_v \end{bmatrix}^T$ is the position of the autonomous vehicle, ϕ is the orientation, which are obtained by the fusion of GPS/IMU/DMI in real time.

The pedestrian pose $\hat{\zeta}$ and its associate error covariance P are predicted using a Kalman filter as

$$\begin{aligned} \hat{\zeta}(t|t-1) &= F\hat{\zeta}(t-1) \\ P(t|t-1) &= FP(t-1)F^T + GQ(t-1)G^T \end{aligned} \tag{7}$$

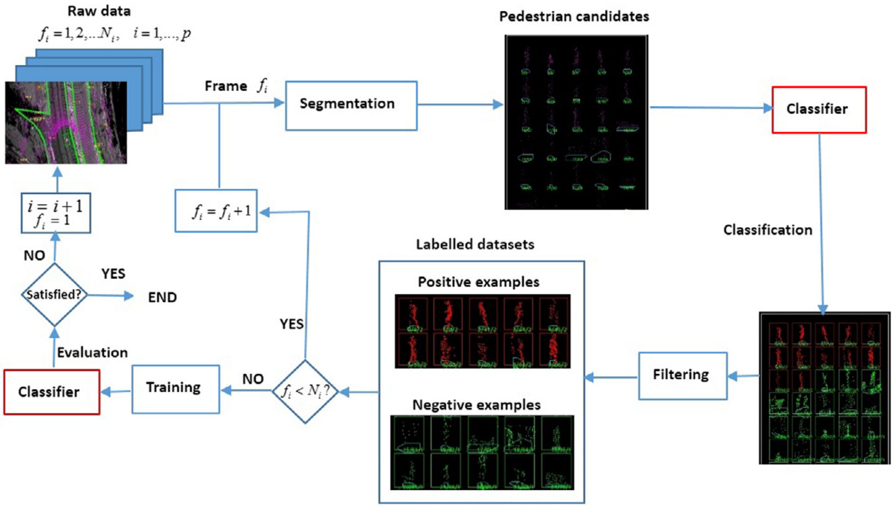


Fig. 6. Flow chart for labeling and classifier training.

where Q is the covariance of the noise $\Delta\zeta$. To achieve a reliable data association, we apply a global-nearest-neighbor (GNN) algorithm [25,26].

After recognizing and tracking pedestrians around the autonomous vehicle, pedestrian positions are then compared with curb positions (the calculation of curb positions will be presented in the next section), alarms will be generated if either of the following two conditions is satisfied:

- Pedestrian is on road in front of autonomous vehicle
- Pedestrian is tending to cross the road.

The second condition can be monitored by checking the positions and velocity directions of pedestrians who are close to curbs. In addition, we track all pedestrian candidates by applying nearest neighbor algorithm, the same pedestrian candidate is assigned an invariant ID, this ID exists until the candidate is lost. The recognized pedestrians are divided into two categories by comparing their positions with curbs', i.e., on road and outside of road. If a candidate is classified to be a pedestrian in frame j, it will be classified to be *pedestrian* in the following frames $j+1, j+2, \dots,$ as long as the tracking is not interrupted. The reason of this is that most false positives are caused by objects outside of road due to their similar shapes to pedestrians, while on-road false negative needs to be avoided as much as possible, however, when on-road pedestrians become too close or too far from autonomous vehicle, only a few laser beams are irradiated on pedestrian, this may cause missing recognition of pedestrians (i.e., false negative), this is very dangerous for autonomous vehicles. By the aid of tracking results and curb detection results, the true positive rate has been improved a lot, a quantitative evaluation will be presented in Section 6.

5. Experiments

5.1. Quantitative evaluation

In this section, the quantitative evaluation is carried out to verify the effectiveness of the pedestrian recognition and tracking algorithm proposed in this paper. 3D range data is collected by Velodyne 64 (mounted on top of an autonomous vehicle) in real urban environment, Fig. 7 shows the autonomous vehicle equipped with Velodyne 64, Velodyne 16, IMU and Encoder etc as stated in Section 2. Two kinds of scenarios are chosen for the experiment, one scenario is that the autonomous vehicle is parked by the side of a crowded road, pedestrians are walking around, including those who are carrying briefcases or backpacks, walking side by side etc. Another scenario is that the autonomous vehicle is moving on the road, different pedestrians are walking/waiting on the sidewalk, or crossing the road in front of the autonomous vehicle.

Fig. 7. Autonomous vehicle in real road environment.

Table 1
Condition of the scenario when autonomous vehicle is parked by the side of the road.

Description	Total	N pos.	N neg.
Training data	2770	608	2162
Evaluation data	10050	1380	8670

Table 2
Condition of the scenario when autonomous vehicle is moving on the road.

Description	Total	N pos.	N neg.
Training data	2282	328	1954
Evaluation data	6340	1020	5320

Tables 1 and 2 show conditions for training and evaluating in these two scenarios, respectively. Figs. 8 and 9 show the corresponding ROC (receiver operating characteristic) curves, where the vertical and horizontal axes represent true positive rate and false positive rate, respectively. The dashed line is recognition result of classifier trained by SVM, while solid line is the result improved by tracking results. From Figs. 8 and 9, it can be seen that true positive rate is increased approximately 0.15 and 0.1 at the point where false positive rate is 0.02 for these two scenarios, respectively.

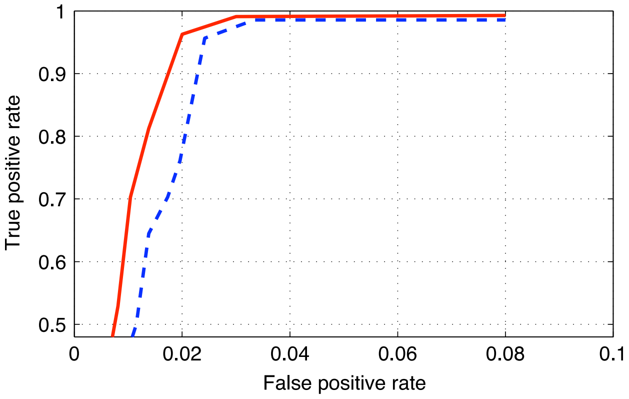


Fig. 8. Recognition performance when autonomous vehicle is parked by the side of road. Dashed line is the recognition result of classifier trained by SVM, while solid line is the result improved by tracking results.

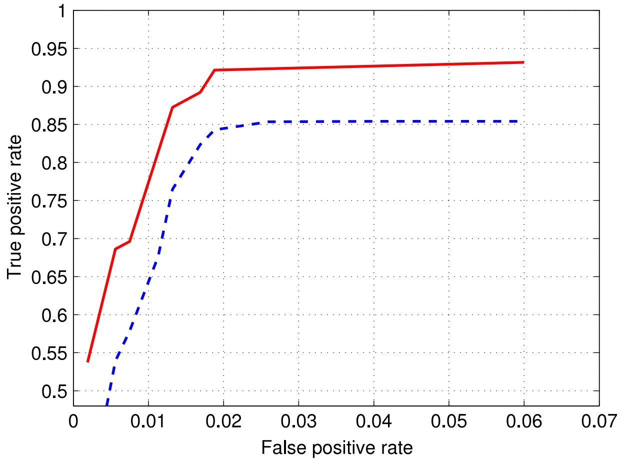


Fig. 9. Recognition performance when autonomous vehicle is moving on road. Dashed line is the recognition result of classifier trained by SVM, while solid line is the result improved by tracking results.

5.2. *Visualization of pedestrian recognition in real time*

In order to show the pedestrian recognition and tracking results in real time, a visualization interface is developed. Firstly, a 2D gray map is generated from 3D point clouds collected by Velodyne 64 by the aid of SLAM (Simultaneous Localization and Mapping) techniques [27,28]. Fig. 12 shows a 2D map generated from the data collected by Velodyne 64 at 1 Fusionopolis Way, Singapore. Secondly, curb positions are converted into the same coordinate frame of the 2D map. The precise curb positions are calculated using 3D point cloud data collected by Velodyne 16 together with vehicle poses calculated by fusing GPS/IMU/DMI. Since it is assumed that poses of autonomous vehicle are obtained with enough accuracy all the time, the curb detection task in this paper is to select the curb points from point clouds and fit them to line segments, the detailed curb detection algorithm is shown in Algorithm 1. Fig. 10 presents a sample raw data collected by Velodyne 16 in urban environment, Fig. 11 shows the curb detection result where curb points are marked by white points. Fig. 12 shows the experiment environment, i.e., 2D map with curb lines in green.

Algorithm 1 Framework of curb detection.

```
Require:
    Point clouds collected by Velodyne 16;
Ensure:
    Step 1: Given input point cloud, select interested area;
    Step 2: Calculate vector difference of adjacent points;
    Step 3: Select curb-like points and filter out noises;
    Step 4: Separate higher obstacles by comparing the height with
        a threshold;
    return Curb points.
```

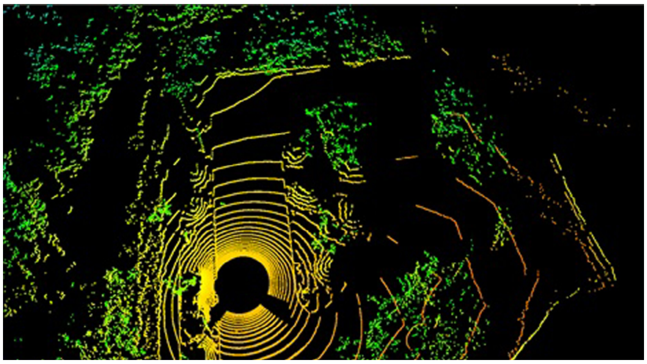


Fig. 10. A sample raw data collected by Velodyne 16.

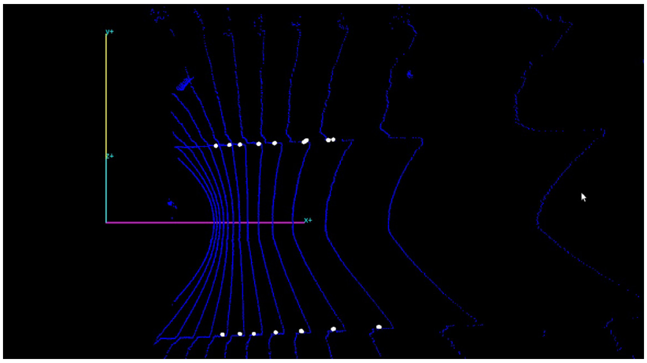


Fig. 11. A sample curb detection result, both blue and white points are from Velodyne 16, white points are detected as curb points through Algorithm 1. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Thirdly, for real time demonstration, the pedestrian recognition and tracking algorithms are implemented in C++ and running on a laptop with Intel i7 CPU @2.4 GHz and 8G RAM, the total time used to process the data collected by Velodyne 64 is less than 60 ms per scan. The pedestrian recognition and tracking results of one scan are shown in Figs. 13 and 14 for two different scenarios, respectively. For each of them, on the left half part, pedestrians recognized without the aid of tracking results are shown in red, while pedestrians recognized by the aid of tracking results are shown in orange. The other non-pedestrian objects are shown in blue, each object is labeled by an ID. The right half part of Fig. 13 shows the pedestrian recognition results which can be observed from 360-degree field of view, where the red ones are pedestrians recognized without the aid of tracking results, the white ones are obtained by the aid of tracking results, the green ones are non-pedestrian objects. Fig. 15 shows parts of Fig. 13, where a pedestrian is recognized in the middle road whose velocity is zero, an alarm will be generated. It should be pointed out that due to roof blocking and field of view of Velodyne 64, there exist blind corner areas around the autonomous vehicle, for pedestrians occurring in these areas cannot be recognized. In Fig. 16, two pedestrians are recognized who are shown in red, the velocities are 1.50 m/s

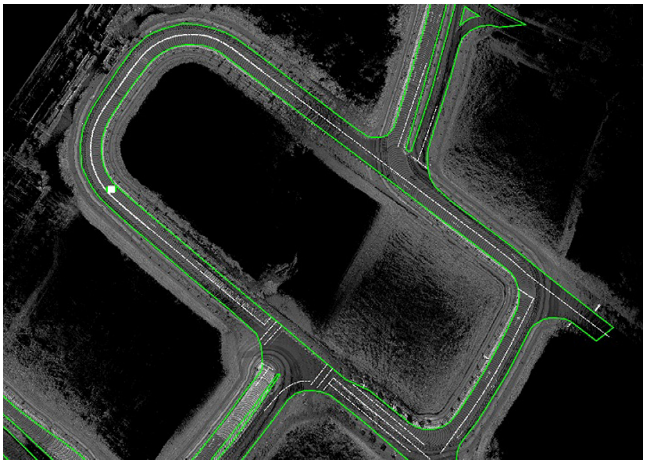


Fig. 12. 2D bird's-eye-view map of real road environment, curb lines are in green. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

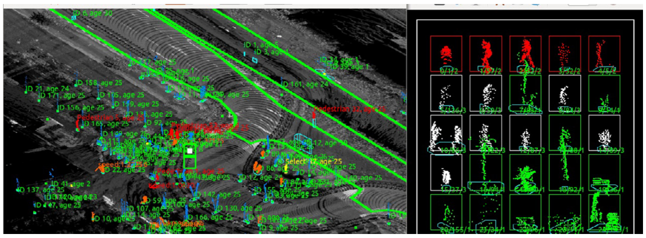


Fig. 13. Pedestrian recognition result when autonomous vehicle is parked by the side of road. Pedestrians recognized without the aid of tracking results are in red for both left and right half parts, pedestrians recognized by the aid of tracking results are in orange for the left half part and in white for the right half part, the others are classified to be non-pedestrian objects. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

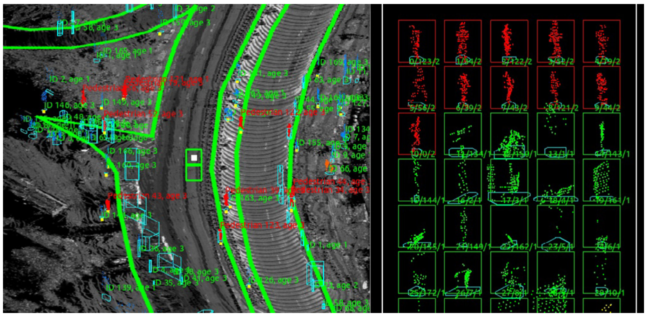


Fig. 14. Pedestrian recognition result when autonomous vehicle is moving on road. Pedestrians recognized are in red for both left and right half parts, the others are classified to be non-pedestrian objects. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

and 1.18 m/s, respectively. The arrows in Figs. 15 and 16 indicate velocity directions.

6. Conclusion

In this paper, we have studied the pedestrian recognition and tracking problem for autonomous vehicle systems, a classifier is trained through SVM to classify blobs, separated from the raw data collected by a Velodyne 64 LiDAR, into pedestrians and non-pedestrians. By combining pedestrian tracking with curb detection results, an alarms is generated if a pedestrian is recognized to be on road or close to curbs. Hash table techniques have been used in segmentation procedure, and a GUI is developed for labeling, training and real-time visualization. The future work includes applying the algorithms proposed in this paper to sparse Velodyne 16 data, recognizing other interested objects such as cars and bicycles etc. [29].

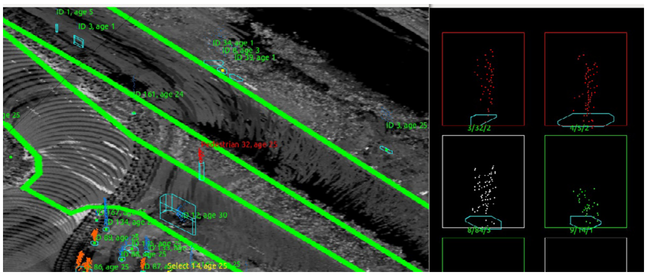


Fig. 15. On-road pedestrian recognition result. The pedestrian is in red, whose ID is 32. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

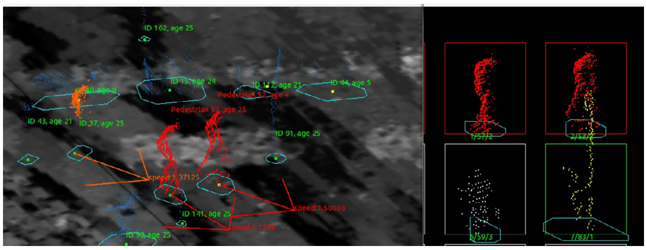


Fig. 16. Moving pedestrian recognition result, two pedestrians are walking side by side, whose IDs are 57 and 52, respectively. Both pedestrians are in red. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

References

[1] J. Wu, C.G. James, M. Rehg, Real-time human detection using contour cues, in: IEEE International Conference on Robotics and Automation, ICRA, Shanghai, China, 2011, pp. 960–967.

[2] S. Tang, M. Andriluka, B. Schiele, Detection and tracking of occluded people, Int. J. Comput. Vis. 10 (2014) 58–69.

[3] K. Mikolajczyk, C. Schmid, A. Zisserman, Human detection based on a probabilistic assembly of robust part detectors, in: 8th European Conference on Computer Vision, ECCV, Springer, Berlin, Heidelberg, 2004, pp. 69–82.

[4] P. Srinivasan, J. Shi, Bottom-up recognition and parsing of the human body, in: IEEE Conference on Computer Vision and Pattern Recognition, Minneapolis, MN, 2007, pp. 1–8.

[5] P.F. Felzenszwalb, R.B. Girshick, D. McAllester, D. Ramanan, Object detection with discriminatively trained part based models, IEEE Trans. Pattern Anal. Mach. Intell. 32 (9) (2010) 1627–1645.

[6] J. ao Xavier, M. Pacheco, D. Castro, A. Ruano, U. Nunes, Fast line, arc circle and leg detection from laser scan data in a player driver, in: IEEE International Conference on Robotics and Automation, Barcelona, Spain, 2005, pp. 3930–3935.

[7] L. Spinello, K.O. Arras, R. Triebel, R. Siegwart, A layered approach to people detection in 3d range data, in: Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence, Atlanta, Georgia, USA, 2010.

[8] N. Kirchner, A. Alempijevic, A. Virgona, Head-to-shoulder signature for person recognition, in: IEEE International Conference on Robotics and Automation, RiverCentre, Saint Paul, Minnesota, USA, 2012, pp. 1226–1231.

[9] L.E. Navarro-Serment, C. Mertz, M. Hebert, Pedestrian detection and tracking using three-dimensional ladar data, Int. J. Robot. Res. 29 (12) (2010) 1516–1528.

[10] K. Kidono, T. Miyasaka, A.W.T. Naito, J. Miura, Pedestrian recognition using high-definition lidar, in: IEEE Intelligent Vehicles Symposium (IV), 2011.

[11] K.O. Arras, Oscar Martinez Mozos, W. Burgard, Using boosted features for the detection of people in 2d range data, in: IEEE International Conference on Robotics and Automation, Roma, Italy, 2007, pp. 3402–3407.

[12] Z. Zivkovic, B. Kröse, Part based people detection using 2d range data and images, in: IEEE/RSJ International Conference on Intelligent Robots and Systems, San Diego, CA, 2007, pp. 214–219.

[13] L.E. Navarro-Serment, C. Mertz, N. Vandapel, M. Hebert, LADAR-based pedestrian detection and tracking, in: 1st IEEE Workshop on Human Detection from Mobile Platforms, Pasadena, California, May 20th 2008.

[14] C. Premebida, U. Nunes, Fusing lidar, camera and semantic information: a context-based approach for pedestrian detection, Int. J. Robot. Res. 32 (3) (2013) 371–384.

[15] P. Morton, B. Douillard, J. Underwood, An evaluation of dynamic object tracking with 3D LIDAR, in: Proceedings of Australasian Conference on Robotics and Automation, Monash University, Melbourne, Australia, 2011.

[16] J. Shackleton, B. VanVoorst, J. Hesch, Tracking people with a 360-degree lidar, in: IEEE Conference on Advanced Video and Signal Based Surveillance, 2010, pp. 420–426.

[17] I. Miller, M. Campbell, D. Hutten-locher, Efficient unbiased tracking of multiple dynamic obstacles under large viewpoint changes, IEEE Trans. Robot. 27 (1) (2011) 29–46.

[18] F. Moosmann, T. Fraichard, Motion estimation from range images in dynamic outdoor scenes, in: IEEE International Conference on Robotics and Automation, ICRA, 2010, pp. 142–147.

[19] M.T. Goodrich, R. Tamassia, Algorithm Design. Foundations, Analysis, and Internet Examples, John Wiley & Sons, Chichester, 2001.

[20] Edgar E. Osuna, R. Freund, Federico Girosi, Support vector machines: Training and applications, AI. Memo No. 1602, 1997.

[21] N. Cristianini, J. Shawe-Taylor, An Introduction to Support Vector Machines and Other Kernel-Based Learning Methods, Cambridge University Press, 2000.

[22] J. Wang, P. Neskovic, Leon N. Cooper, Training data selection for support vector machines, in: ICNC 2005, 2005, pp. 554–564.

[23] L.E. Navarro-Serment, C. Mertz, M. Hebert, Pedestrian detection and tracking using three-dimensional LADAR data, in: Proc. Int. Conf. on Field and Service Robotics, 2009.

[24] O. Chapelle, V. Vapnik, O. Bousquet, S. Mukherjee, Choosing multiple parameters for support vector machines, Mach. Learn. 46 (1) (2002) 131–159.

[25] M. Ozaki, K. Kakimuma, M. Hashimoto, K. Takahashi, Laser-based pedestrian tracking in outdoor environments by multiple mobile robots, Sensors 12 (2012) 14489–14507.

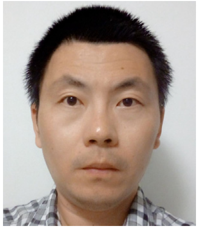
[26] P. Konstantinova, A. Udvarev, T. Semerdjiev, A study of a target tracking algorithm using global nearest neighbor approach, in: Proceedings of the 4th International Conference on Systems and Technologies, Ruse, Bulgaria, 2003, pp. 290–295.

[27] T. Bailey, J. Nieto, Scan SLAM: Recursive mapping and localisation with arbitrary shaped landmarks, in: Robotics: Science and Systems Conference, RSS, 2008.

[28] S.D. Huang, G. Dissanayake, Convergence and consistency analysis for extended Kalman filter based SLAM, IEEE Trans. Robot. 23 (5) (2007) 1036–1049.

[29] T. Chen, B. Dai, R. Wang, D. Liu, Gaussian-Process-based real-time ground segmentation for autonomous land vehicles, J. Intell. Robot. Syst. 76 (3) (2014) 563–582.

Heng Wang received the Bachelor's degree and the Ph.D. degree from the College of Information Science and Engineering at Northeastern University, PR China in 2003 and 2008, respectively. He is currently an Associate Professor at Northeastern University, China. He was a Post-Doc Research Fellow at the Centre for Autonomous Systems, University of Technology, Sydney from July 2010 to July 2011, and a Lecturer at College of Electronic Information and Control Engineering, Beijing University of Technology, China from July 2008 to March 2014. From April 2014 to November 2015, he worked as a Research Scientist at Institute for Infocomm Research(I2R), A*STAR, Singapore. His current research interests include fault detection, robust control, and mobile robots simultaneous localization and mapping (SLAM) etc.



Bin Wang is a Research Engineer with Institute for Infocomm Research, A*STAR, Singapore. He received his Bachelor and Master degrees in 2000 and 2002, respectively, both from Harbin Institute of Technology, China. His research interests include sensor fusion for mapping and localization for autonomous vehicles.

Bingbing Liu is a Research Scientist with Institute for Infocomm Research, A*STAR, Singapore and he also leads the group of Localization and Mapping of the Autonomous Vehicle Department. He received his Bachelor degree from Harbin Institute of Technology, China in 2000 and the Ph.D degree from Nanyang Technological University, Singapore in 2007. His research interests include SLAM, inertial navigation, sensor fusion for mapping and localization for autonomous vehicles etc.

Xiaoli Meng received the B.E. degree in communication engineering from Tianjin University, Tianjin, China, in 2007, and the Ph.D. degree from the Sensor Network and Application Research Center, Graduate University, Chinese Academy of Sciences, Beijing, China, in 2012. She was a Research Fellow with the Department of Bioengineering, Nation University of Singapore, Singapore, from 2012 to 2013. She is currently a Research Scientist with the Institute for Infocomm Research, Singapore. Her current research interests include autonomous vehicle localization, object detection and tracking, and human motion tracking.

Guanghong Yang received the B.S. and M.S. degrees from Northeast University of Technology, Liaoning, China, in 1983 and 1986, respectively, and the Ph.D. degree in Control Engineering from Northeastern University, China (formerly, Northeast University of Technology), in 1994. He was a Lecturer/Associate Professor with Northeastern University from 1986 to 1995. He joined the Nanyang Technological University in 1996 as a Postdoctoral Fellow. From 2001 to 2005, he was a Research Scientist/Senior Research Scientist with the National University of Singapore. He is currently a professor at the College of Information Science and Engineering, Northeastern University. His current research interests include fault tolerant control, fault detection and isolation, non-fragile control systems design, and robust control. Dr. Yang is an Associate Editor for the International Journal of Control, Automation, and Systems (IJCAS), the International Journal of Systems Science (IJSS), the IET Control Theory & Applications, and the IEEE Transactions on Fuzzy Systems.