DETECTION AND TRACKING OF PEDESTRIANS USING DOPPLER LIDAR

by

Xiaoyi Peng

A Thesis

Submitted to the Faculty of Purdue University

In Partial Fulfillment of the Requirements for the degree of

Master of Science in Civil Engineering

Lyles School of Civil Engineering

West Lafayette, Indiana

May 2021

THE PURDUE UNIVERSITY GRADUATE SCHOOL

STATEMENT OF COMMITTEE APPROVAL

Dr. Jie Shan, Chair

Lyles School of Civil Engineering

Dr. Mohammad R. Jahanshahi

Lyles School of Civil Engineering

Dr. James L. Garrison

School of Aeronautics and Astronautics

Approved by:

Dr. Dulcy M. Abraham

ACKNOWLEDGMENTS

First and foremost, I am extremely grateful to my advisor, Prof. Jie Shan, for his continuous and enthusiastic support of my research as well as the immense knowledge he imparted to me throughout the past years. I am also thankful to the other members of my advisory committee, Prof. James Garrison and Prof. Mohammad R. Jahanshahi, for their kindness, encouragement, and insightful comments.

I am grateful as well to Dr. Wenyuan Zhang, Dr. Xianjun Gao, Dr. Qinghua Li, Zhixin Li, Xiangxi Tian, Ce Wang, and Zilong Yang, from whom I learned state-of-art methods in machine learning and data-mining. I am also thankful to Blackmore Sensors and Analytics Inc. that provided the data for this study.

Last but not least, I cannot thank my family enough for supporting me throughout my life, especially my parents for their spiritual support during my academic journey and their unceasing encouragement and concern.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

Autonomous Vehicle	AV
Advanced Driving Assisted System	ADAS
Cloth Simulation Filter	CSF
Density-Based Spatial Clustering of Applications with Noise	DBSCAN
Frequency Modulated Continuous Wave	FMCW
False Negative	FN
False Positive	FP
Identity Switch	IDSW
Inertial Measurement Unit	IMU
Intersection over Union	IoU
Kalman Filter	KF
Light Detection And Ranging	LiDAR
Multiple Objects Tracking Accuracy	MOTA
Principal Component Analysis	PCA
Random Forest	RF
Single Template Matching with Kernel Density Estimation	STM-KDE
Support Vector Machine	SVM
True Negative	TN
True Positive	TP

ABSTRACT

Pedestrian detection and tracking plays an essential role in autonomous vehicles and mobile service robots. This thesis presents a novel solution to pedestrian detection and tracking for urban scenarios based on Doppler LiDAR that records both position and velocity of the targets. The workflow consists of two stages. In the detection stage, the input point cloud is first segmented to form clusters frame by frame. A subsequent re-clustering process is introduced to further separate pedestrians close to each other. While a simple speed classifier is capable of extracting most of the moving pedestrians, a supervised machine learning-based classifier is adopted to detect pedestrians with insignificant radial velocity. In the tracking stage, pedestrian's state is estimated by a Kalman filter, which uses the speed information to measure the pedestrian's dynamics. Based on the similarity between the predicted and detected states of pedestrians, a greedy algorithm is adopted to associate the trajectories with the detection results. The presented detection and tracking methods are tested on two datasets collected in San Francisco, California by a mobile Doppler LiDAR system. The results of pedestrian detection show that the proposed two-step classifier can improve the detection performance, particularly for detecting pedestrians far from the sensor. For both datasets, the speed information increases the F1-score and the recall by 10% and 20%, respectively. Moreover, the quantitative evaluation of tracking results shows the Kalman filter with speed information is able to enhance the accuracy of the position estimation and improve the multiple object tracking accuracy (MOTA) by 5% for both datasets.

1. INTRODUCTION

1.1 Background

Almost 6,000 pedestrians were killed in 2017 in traffic crashes in the U.S., which was about one death every 1.5 hours. In addition, more than 130,000 pedestrians were treated in hospital emergency departments for nonfatal crash-related injuries in 2017 (Retting 2017). It is also known that pedestrians are more likely to be killed during a car crash than the crashed vehicle's passengers. For these reasons, pedestrian detection and tracking has become a significant and essential task for many traffic-related applications, such as autonomous vehicles (AV), advanced driving assisted systems (ADAS), and traffic management. For AV and ADAS, reliable detection and tracking of pedestrians aims to make vehicles aware of potential danger in their vicinity and thereby improve traffic safety. Such a system provides spatial-temporal information for vehicles to plan and adapt their subsequent actions. For traffic management, precise detection and tracking of pedestrians could assist in optimizing traffic control and scheduling in order to achieve high levels of safety and efficiency.

For the purpose of pedestrian detection and tracking, vision-based approaches are prevalent (Cao et al. 2008; Dehghan et al. 2014; Stewart et al. 2016). These approaches recognize and track pedestrians in images and videos by extracting the texture, color, and contour features of the targets. However, such approaches have difficulty in collecting accurate position information about humans due to their limited accuracy in depth estimation. Some researchers have tried to deal with this problem using RGB-D cameras, which combine information from images and 2D rangers to collect color information and dense point clouds simultaneously (Jafari et al. 2014; Premebida et al. 2014; Liu et al. 2015). However, RGB-D cameras usually have a narrow field of view both horizontally and vertically and limited sensor ranges (Chen et al. 2018). As such, applications that incorporate 3D LiDAR sensors in pedestrian detection and tracking have experienced dramatic development in recent years (Haselich et al. 2014; Cabanes and Senouci 2017; Wu et al. 2021). Compared to cameras or RGB-D cameras, LiDAR is a direct 3D measurement technology without the need for image matching. Another significant advantage of 3D LiDAR sensors is their ability to generate long-range and wide-angle point clouds. In addition, LiDAR point clouds are quite accurate and not affected by lighting conditions (Yan et al. 2017).

Currently, most LiDAR-based studies utilized point cloud datasets acquired by a LiDAR sensor which only collects spatial information of data points. But when pedestrians are far from the sensor, fewer points of them are collected by the scanner, which may cause missing or wrong recognition of pedestrians (Li et al. 2016; Zhang et al. 2019). Doppler LiDAR (Royo and Ballesta-Garcia 2019), which not only provides spatial information but precise radial velocity of each data point, can possibly help to address this problem. For example, as a pedestrian moves away from the sensor, its point cloud becomes sparse while its velocity would not change a lot.

This thesis aims to take advantage of the Doppler LiDAR to propose a new detection-based tracking method to detect and track pedestrians in urban scenes. The contribution of this thesis includes the following aspects. (1) A re-clustering process based on the mean shift algorithm is utilized to further segment pedestrian candidates. This process can increase the true positive rate for candidates who become too close to other objects. (2) We use speed information from the Doppler LiDAR to improve both detection and tracking performance. Specifically, for pedestrian detection, the pedestrian classifier with the speed information is robust to classify pedestrians at any distance. In the tracking step, the speed information provides a more accurate prediction of the pedestrian's location, leading to better tracking performance.

1.2 Related work

1.2.1 Pedestrian detection

Current pedestrian detection studies can be broadly classified into two approaches: model-free and model-based. Model-free methods have no restrictions on or assumptions about the shape and size of the objects to be detected. As such, they can detect pedestrians and other dynamic objects on the road, such as vehicles and bicyclists, simultaneously. Pomerleau et al. (2014) outlined a system for long-term 3D mapping in which they compared an input point cloud to a global static map and then extracted dynamic objects based on a visibility assumption. Azim and Aycard (2012) modeled the dynamic environment by an octree-based occupancy grid and segmented the dynamic objects on the basis of discrepancies between consecutive frames. They then generated 3D bounding boxes for the dynamic objects and classified them according to the geometric properties of their bounding boxes, such as their size and the ratio between their height, width, and length. Wang et al. (2015) presented a Bayesian framework for detection and tracking

pedestrians with data obtained by a 2D LiDAR sensor, in which the motion states and shape information of dynamic objects, the sensor pose, and the local static background were estimated by using a joint state representation. Dewan et al. (2015) implemented a model-free approach for pedestrian detection that relies on multiple motion cues. They first detected motions sequentially using RANSAC and then proposed a Bayesian approach to segment objects. Most of the proposed model-free methods are mainly based on motion cues, so they can fail if the motion of an object between consecutive frames is small. Therefore, the performance of model-free approaches for detecting pedestrians has never been as good as its detection of other objects, such as vehicles and bicyclists, since pedestrians always move slowly. The approach proposed by Dewan et al. (2015) did not work well for pedestrians; in fact, their approach entirely missed pedestrians if they continually walked at a low speed. Ma et al. (2019) proposed a model-free approach that achieved high performance for pedestrian detection using point clouds acquired by a Doppler LiDAR. Their approach first detected and clustered all the moving points by ST-DBSCAN to generate a set of dynamic point clusters. Then, the dynamic objects were completed from the detected dynamic clusters by region growing. The Doppler LiDAR provided speed information with a precision of $\pm 0.1\ m/s$. As such, most pedestrians could be detected successfully except those with zero radial speed.

Model-based approaches are preferred when some information about the object to be detected is known and therefore can be modeled a priori. Liu et al. (2019) proposed a pedestrian detection algorithm named single template matching with kernel density estimation (STM-KDE) clustering. Their algorithm first segments the point clouds by KDE, and the candidate clusters are projected into its main plane, which is determined by eigenvectors corresponding to the two large eigenvalues derived from the principal component analysis (PCA). Then, the minimum distance of the points to the main plane is regarded as the pixel value, and the locally adaptive regression kernel (LARK) feature of this projected candidate is adopted to estimate the contour of the pedestrians. If the cosine similarity between the LARK feature from a candidate cluster and the template is less than a pre-defined threshold, this cluster is recognized as a pedestrian.

Currently, a large number of studies on pedestrian detection from LiDAR rely on machine learning strategies. In traditional machine learning, pedestrians are represented numerically by hand-crafted features. Many studies detected people mainly by relying on the detection of the legs of people because legs are quite distinctive of the human figure. Lee et al. (2006) detected the

position of people using the geometric characteristics of human legs and proposed a human walking model based on the velocity of each leg and the frequency of steps. Cui et al. (2007) extracted each leg of persons by their pattern of rhythmic leg swing, from which people then were tracked by a final tracker composed of a dependent Kalman filter and a Rao-Blackwellized Monte Carlo data association filter. Chen et al. (2019) modeled the human leg by extracting 12 features that describe the geometric and statistical characteristics of the human leg numerically and constructed a machine learning-based human leg classifier to detect pedestrians. However, these approaches did not work well if the collected point clouds cannot provide detailed information about the legs. Thus, detection of pedestrians in many studies was done by extracting features which present the entire person. Navarro-Serment et al. (2010) proposed 11 features based on the property of clusters and PCA to describe human geometry and then classified the pedestrian candidates by a classifier composed of two independent SVMs. Based on this work, Kidono et al. (2011) improved the performance of the classifier by adding two new features: a slice feature for the cluster and a distribution pattern for the reflection intensity of the cluster. Their results showed that these two new features improved classification performance significantly even if their dimensions were relatively low. Wang and Posner (2015) presented a sliding window approach to 3D point data for pedestrian detection, wherein the point cloud was divided into a grid at a fixed resolution and each cell was represented by a vector of six features related to the scatter of points within the cell, the reflectance values of these points, and the occupancy of the cell. A 3D window detector of a given size slid down all three dimensions to stack up all the feature vectors falling in its bound into a single vector; then, a classifier was used to determine whether the current detection window contains a pedestrian. Navarro et al. (2017) applied a machine learning-based approach to detect pedestrians using high-definition 3D range data, whereby each candidate pedestrian cluster is first projected into three main planes, and a corresponding binary image for each projection is generated to extract the feature vectors. Then, k-Nearest Neighbor, Naive Bayes, and SVM classifier were used to detect pedestrians based on the above features.

Some model-based neural networks for 3D object detection have been developed in recent years in an end-to-end manner. These approaches did not rely on hand-crafted features and typically followed one of the two pipelines, i.e., either two-stage or one-stage object detection. In a two-stage detector, object candidates were first generated by region proposal networks and then input to the second stage for refinement. For example, Yang et al. (2018) generated region

proposals with multiple scale, angle and shift from predicted foreground points and then used intersection-over-union (IoU) and a new criterion named PointsIoU to reduce the redundancy and ambiguity of proposals, respectively. Shi et al. (2019) presented a two-stage PointRCNN for 3D object detection from a raw point cloud. The 3D proposals were first directly generated from the raw point clouds in stage I and then refined by fusing the semantic features and the local spatial features in stage II. The one-stage object detection pipeline predicted class probabilities and regressed the 3D bounding boxes of objects directly using a single-stage network. It could run at a high speed since region proposal and post-processing were not required. Zhou et al. (2018) proposed an approach called VoxelNet, which learns feature representation from point clouds and predicts the 3D bounding boxes of objects in an end-to-end manner. VoxelNet was composed of a feature learning network, convolutional middle layers, and a region proposal network. Lang et al. (2019) proposed a 3D object detector named PointPillars that leveraged PointNet (Qi et al. 2017) to learn the features of point clouds organized in vertical columns and encodes the learned features as a pseudo image. A 2D object detection pipeline was then applied to predict 3D bounding boxes.

Despite the fact that deep learning-based approaches provided state-of-the-art performance in many object detection tasks, this thesis did not adopt them for the following reasons. First, such methods typically required considerable fine-tuning with manual intervention, longer training time, and high-performance hardware (Yan et al. 2020). Also, pedestrian detection was essentially a straightforward binary classification, rather than a complex object detection problem. Moreover, most 3D object detection neural networks were evaluated using the KITTI benchmark (Geiger et al. 2013), while the amount of data collected by Doppler LiDAR was much less than the KITTI dataset. When training data were limited, deep learning strategies did not necessarily outperform traditional classification methods (Zhang et al. 2020).

1.2.2 Pedestrian tracking

Typical urban scenarios always contained more than one pedestrian, who may be close to, overlapping with, or obstructed by other objects, including pedestrians. Thus, a pedestrian tracking algorithm must be able to deal with multiple pedestrians, ranging from two well-separated pedestrians to small groups of pedestrians. To be specific, the objective of tracking multiple pedestrians required locating multiple pedestrians, maintaining their identities, and tracking their individual trajectories in a given point cloud sequence. Existing tracking approaches could be

grouped into two categories based on their processing mode: offline tracking and online tracking (Camara et al. 2020). Offline tracking utilized information both from past and future frames and attempts to find a globally optimal solution, which could be formulated as a network flow graph and solved by min-cost flow algorithms (Zhang et al. 2008; Brendel et al. 2011; Schulter et al. 2017). Offline tracking always had a high computational time cost since it dealt with observations from all frames and analyzed them jointly to estimate the final output (Luo et al. 2020). In contrast, for online tracking, the LiDAR sequence was handled in a step-wise manner and only considered detections at the current frame, which was usually efficient for real-time applications. Wang et al.(2017) proposed a pedestrian tracking method which was able to improve the performance of pedestrian detection. A constant velocity model was adopted to predict the pedestrians' location, and the global-nearest-neighbor algorithm is used to associate detected candidates and existed trajectories. Once a candidate is associated with an existed trajectory, it would be classified as a pedestrian. Weng and Kitani (2019) proposed an online detection-based tracking method using a Kalman filter and Hungarian algorithm. Based on this work, Chiu et al. (2020) calculated the covariance matrix in a Kalman filter using statistics results from training data and used a greedy algorithm instead of the Hungarian to associate the objects and obtained a better result.

1.3 Structure of the thesis

The remainder of this thesis is organized as follows. Chapters 2 and 3 present the proposed detection and tracking methods, respectively. Chapter 4 introduces the Doppler LiDAR datasets used. Chapter 5 presents the results of the experiments and evaluates the performance of the proposed approach. Finally, Chapter 6 discusses the conclusions and limitations of this thesis.

2. PEDESTRIAN DETECTION

This chapter describes the proposed pedestrian detection method utilized in this thesis. The processing flow from the point cloud segmentation to the human classification is first briefly introduced, and the details of the process are explained thereafter.

Figure 2.1 shows the workflow of the proposed pedestrian detection method. First, all the ground points are removed. The remaining non-ground points are then clustered by a density-based algorithm, and clusters that satisfy certain conditions are selected as pedestrian candidates. After a re-clustering process, the updated pedestrian candidates are input into a two-step pedestrian classifier.

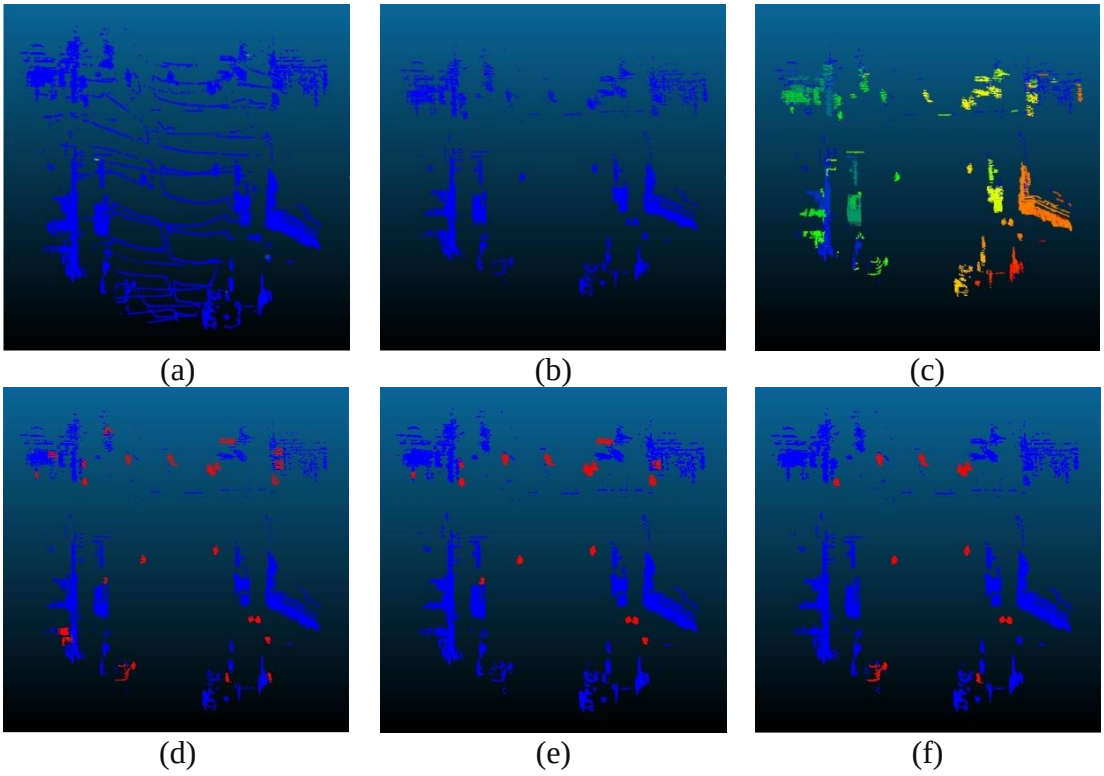


Figure 2.1. Flowchart of the proposed approach for pedestrian detection: (a) input point cloud; (b) non-ground points; (c) clustering result; (d-e) cluster selection and re-clustering (red: pedestrian candidate, blue: other objects); (f) classification result (red: pedestrians, blue: other objects).

2.1 Ground filtering

A given Doppler LiDAR scan P can be represented as a set of 3D points:

$$P = \{p_i | p_i = (x_i, y_i, z_i, v_{ri}, r_i), i = 1,2,3, \dots, M\} \quad (2\text{-}1)$$

where M is the total number of points in the scan, (x, y, z) is the 3D coordinate of a point, v_r is the measured radial velocity, and r is the range to the sensor.

As a portion of the emitted laser beam is reflected by the ground during data collection process, ground points constitute a large proportion of the raw point cloud. Such a large number of ground points not only slow down the process efficiency but affect the performance of point cloud segmentation. To reduce the computational burden and produce a better clustering result, the Cloth Simulation Filter (CSF) is applied to remove all the ground points from the raw point cloud, obtaining a subset $\bar{P} \in P$. In this step, the CSF is selected for the following reasons: (1) compared to traditional filtering algorithms, CSF requires fewer parameters, which are easy to set; and (2) CSF has shown good performance when dealing with datasets in urban areas (Zhang et al. 2016).

2.2 Detection of clusters

2.2.1 3D point clustering

Once ground points are removed, non-ground points are grouped to form clusters, some of which are possible pedestrians. In this thesis, point clusters are generated by a density-based spatial clustering of applications with noise (DBSCAN). DBSCAN requires two parameters: (1) the threshold for the number of neighbors, $MinPts$ and (2) the radius ε to form a dense region. In the clustering process, the points are classified as core points, reachable points, and outliers as follows (Ester et al. 1996):

1) point p is a core point if it has at least $MinPts$ points in its radius ε;

2) point q is a reachable point if it is in radius ε of any core points; and

3) all the points that are not reachable from any core points are considered outliers or noise.

The performance of DBSCAN is susceptible to $MinPts$ and radius ε. The purpose of $MinPts$ is to smooth the density estimate while the radius parameter ε is often harder to set. If the value of ε is too small, a single object is likely to be over-segmented to multiple objects while a

too large ε would merge multiple clusters into one cluster. Additionally, the distance between two vertical adjacent scan lines can vary a lot when the scan range changes. Therefore, for points in different scan ranges, the corresponding value of ε should be different. The strategy proposed by Zhao et al. (2019) is adopted in this thesis to determine ε. That is, the distance ε between different clusters should be at least larger than the vertical distance between two adjacent laser scans, which can be formulated by:

$$\varepsilon \geq 2r \times tan\frac{r_\theta}{2} \tag{2-2}$$

where r is the scan range and r_θ is the vertical angular resolution of the Doppler LiDAR.

Each cluster C in the scan $\bar{P}$ should satisfy the following properties:

$$C_n \in \bar{P}, n = 1,2,3, \dots, N \tag{2-3}$$

$$C_i \cap C_j = \emptyset \quad \text{for } i \neq j \tag{2-4}$$

where Equation 2-3 represents a cluster, C, which is a subset of $\bar{P}$, and Equation 2-4 indicates that there are no common points between two different clusters.

2.2.2 Selection of pedestrian candidates

After the clustering process, every cluster is transformed into a local coordinate system to identify its statistical pattern. For each cluster, the coordinates of all the points are first normalized by subtracting the centroid point coordinates. Generally, pedestrians in urban scenarios remain upright, which is the direction of z-axis. As such, $(0,0,1)^T$ is defined as the z'-axis of the local coordinate system. Then PCA is implemented to the xy plane of this cluster. Two pairs of eigenvalues and eigenvectors are calculated using PCA, which are sorted according to their decreasing eigenvalues. In order to deal with 3D point clouds, the dimension of these two eigenvectors is increased to three by adding zero as their z component. Eigenvectors corresponding to the large eigenvalue and small eigenvalue are defined as the x'-axis and the y'-axis of the local coordinate system, respectively. Then point $(0,0,0)$ is defined as the origin o' of the local coordinate system and the plane $o'x'z'$ is defined as the main plane of the cluster. Similarly, planes $o'y'z'$ and $o'x'y'$ are defined as the secondary plane and the tertiary plane. A sample of the segmented pedestrian in its local coordinate system is shown in Figure 2.2, where Figure 2.2(a) is the 3D point cloud and Figure 2.2(b) through 2.2(d) are the projection results in

the three planes. Based on these projected images, the size of the cluster can be defined by the following equations:

$$\begin{aligned} h &= z'_{max} - z'_{min} \\ l &= x'_{max} - x'_{min} \\ w &= y'_{max} - y'_{min} \end{aligned} \tag{2-5}$$

where h, l, w are height, length, width of the cluster, respectively. Then, all clusters are divided into three categories on the basis of their size: single pedestrian candidates, multiple pedestrians' candidates, and non-pedestrians. Candidates of multiple pedestrians are segmented further in the following step.

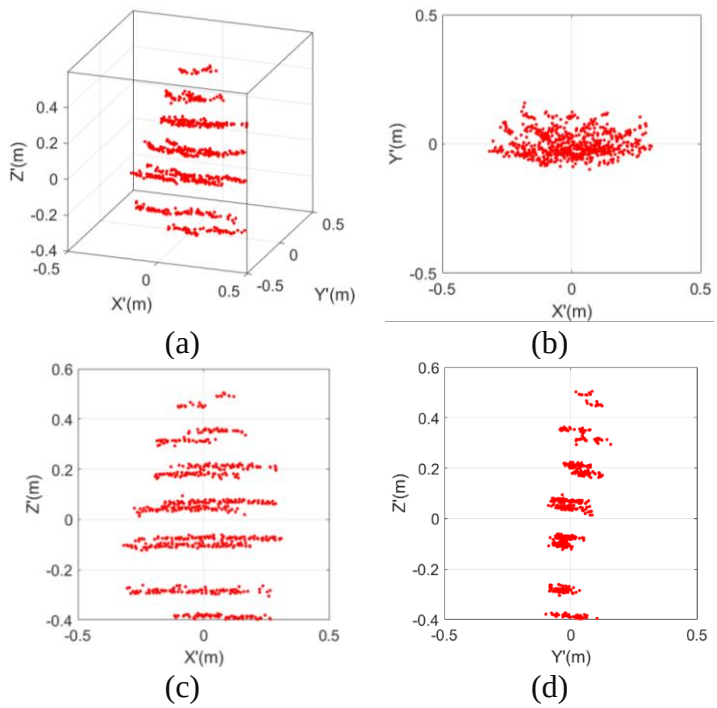
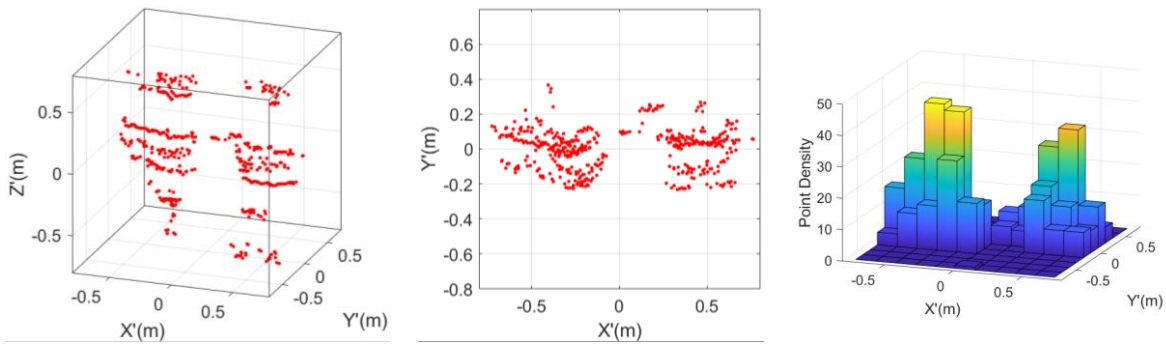


Figure 2.2. Example of a pedestrian candidate: (a) point cloud; (b) through (d) the projected images in the main plane, secondary plane and tertiary plane.

2.2.3 Multiple pedestrians re-clustering

If the distance between two pedestrians is less than the vertical height between two adjacent laser scans, it is not easy to separate them by a distance-based clustering algorithm in 3D space. For example, as can be seen in Figure 2.3(a), the distance between the two pedestrians is less than the set radius ε, so they are regarded as one cluster in the clustering process. However, as clearly

shown in the projected image in the tertiary plane (Figure 2.3(b)), there are two pedestrians that correspond to the two maxima of the density function (Figure 2.3(c)). Therefore, it is feasible to separate multiple pedestrians by recognizing the number of regions with high point densities.



(a) (b) (c)

Figure 2.3. Two pedestrians walking close-by: (a) point cloud; (b) the projected image in the tertiary plane; (c) point density distribution.

To segment such clusters, another density-based algorithm named mean shift clustering is introduced (Fukunaga and Hostetler, 1975). Mean shift is a centroid-based algorithm, which involves shifting the kernel iteratively to a higher density region until convergence. Although mean shift could not segment the whole point cloud as efficiently as DBSCAN due to its higher time complexity ($O(n^2)$), it works well for locating the density peaks corresponding to pedestrians. Given n d-dimensional points x_i, $i = 1,2,3,\dots,n$, the multivariate kernel density estimator with kernel $K(\boldsymbol{x})$ and window radius h is:

$$\hat{f}(\boldsymbol{x}) = \frac{1}{nh^d}\sum_{i=1}^{n} K\left(\frac{\boldsymbol{x}-\boldsymbol{x}_i}{h}\right) \tag{2-6}$$

Every shift is defined by a mean shift vector $\boldsymbol{m}(\boldsymbol{x})$, which is expressed as:

$$\boldsymbol{m}\,(\boldsymbol{x}) = \frac{\sum_{i=1}^{n} \boldsymbol{x}_i g\left(\|\boldsymbol{x}-\boldsymbol{x}_i\|^2/h^2\right)}{\sum_{i=1}^{n} g\left(\|\boldsymbol{x}-\boldsymbol{x}_i\|^2/h^2\right)} - \boldsymbol{x} \tag{2-7}$$

where $g(x) = -K'(\boldsymbol{x})$. The mean shift clustering algorithm is executed in the following steps (Comaniciu and Meer, 2002):

1) start by defining an arbitrary data point as the centroid of a cluster;

2) calculate the mean shift vector based on the current centroid;

3) update the location of centroids based on the mean shift vector obtained in step 2); and

4) iterate the above process so that the centroid then moves to the higher density region.

The process stops once the centroid reaches a position from which it cannot move further; and the points located in a given boundary of this centroid are then regarded as a part of this cluster.

After the re-clustering process, all single pedestrian candidates are then input to a pedestrian classifier.

2.3 Pedestrian classification

2.3.1 Feature selection

The proposed algorithm combines both speed information and static features to recognize pedestrians. Most of the previous studies (Kidono et al. (2011); Navarro et al. (2017); Yan et al. (2017)) only use static features to represent pedestrians numerically. The static feature vector usually consists of several features because a single static feature is not sufficient for determining whether a cluster is a pedestrian. In contrast, the speed information is more straightforward for pedestrian detection as the speed of a pedestrian is unique compared to the static background and other moving objects. The velocity histograms of some typical objects in a street scene are shown in Figure 2.4. For a rigid object, such as a road sign (Figure 2.4(a)), all of its points have the same radial velocity; and due to the measurement noise, the radial velocity distribution is approximately normal. In contrast, the radial velocity distribution of a pedestrian is more complex because different parts of a pedestrian may show different motion patterns (Figure 2.4(b)). Overall, the average velocity of a cluster effectively shows whether or not a cluster is moving. However, pedestrian detection with only speed information may fail in some cases. For example, if a pedestrian is static, his average radial velocity would always be zero (Figure 2.4(c)); or another scenario may be that the moving direction of a pedestrian is almost vertical to the laser beam in which the Doppler LiDAR cannot detect the velocity (Figure 2.4(d)). However, these two categories of pedestrians can be recognized by their static features. Therefore, it is necessary to utilize both speed information and static features to achieve better pedestrian detection performance. In this thesis, a static feature vector composed of 28 features is calculated to describe a pedestrian numerically. The description and dimension of the proposed features are listed in Table 2.1.

Feature f_1 is the number of points belonging to a given cluster, and feature f_2 is the average range to the sensor for the points in a cluster. As can be seen in Figure 2.5, the number of

points belonging to a candidate cluster of pedestrians varies a lot with different ranges of clusters to sensors. Therefore, combining these two features can describe the properties of candidates better compared to only using the number of points or range thereof.

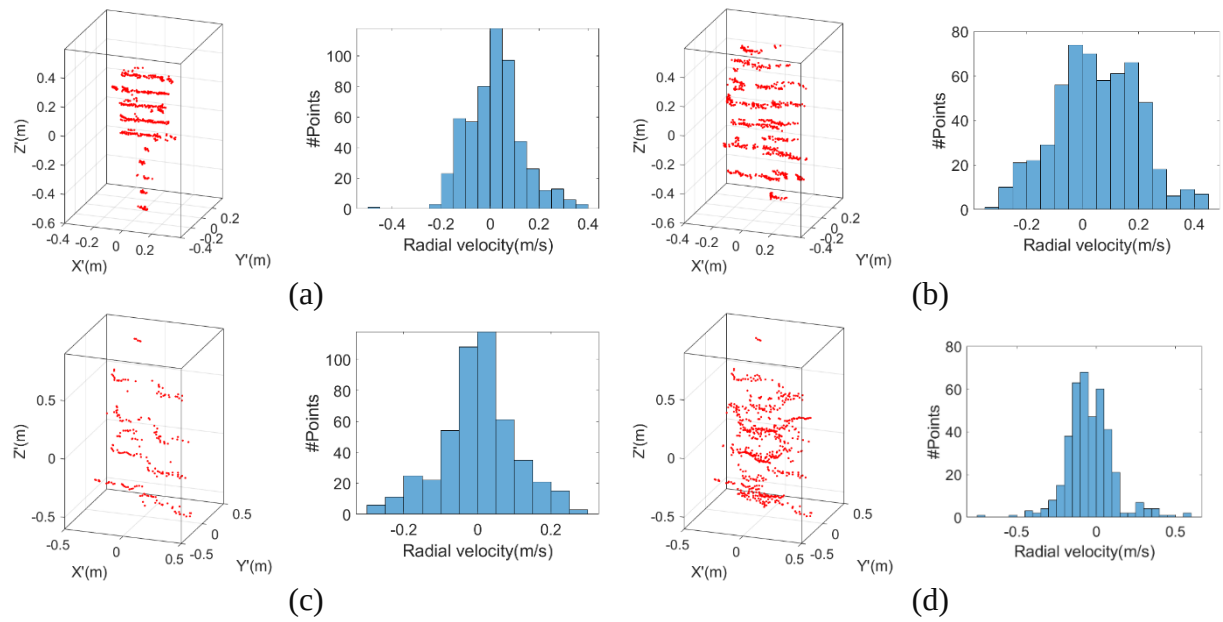


Figure 2.4. Different objects and their velocity histograms: (a) a road sign; (b) a moving pedestrian; (c) a static pedestrian; (d) a pedestrian moving perpendicular to the beam.

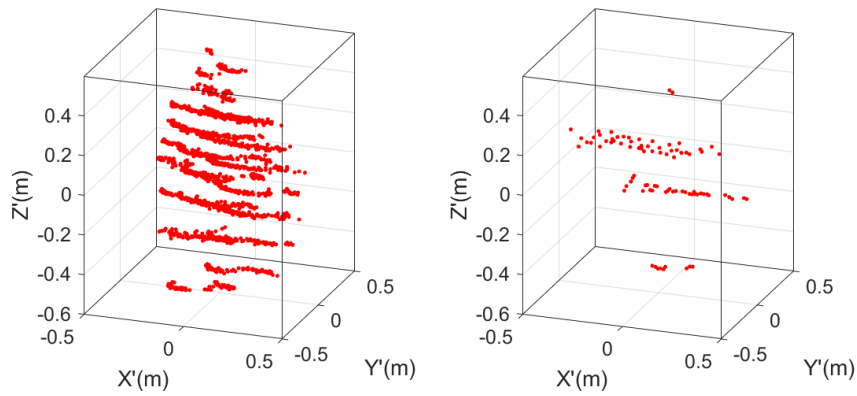
Table 2.1. Features of a cluster for pedestrian classification

Features	Description	Dimension
f_1	Number of points in the cluster	1
f_2	Average range of points to the sensor	1
$f_3 - f_8$	Covariance matrix (3D) of the points in the cluster	6
$f_9 - f_{14}$	Normalized moments of inertia tensor	6
$f_{15} - f_{26}$	Slice feature for the cluster	12
$f_{27} - f_{28}$	Mean and standard deviation of the intensities	2

Feature $f_3 - f_8$ and $f_9 - f_{14}$ are the cluster's 3D covariance matrix Σ_{XX} and normalized moment of inertia tensor M, which represent the overall distribution of all points and can be expressed as:

$$\Sigma_{XX} = \frac{1}{n}\sum_{i=1}^{n}(\boldsymbol{x}_i - \boldsymbol{x}_c)(\boldsymbol{x}_i - \boldsymbol{x}_c)^T \tag{2-8}$$

$$M = \begin{bmatrix} \sum_{i=1}^{n}(y_i^2 + z_i^2) & -\sum_{i=1}^{n} x_i y_i & -\sum_{i=1}^{n} x_i z_i \\ -\sum_{i=1}^{n} x_i y_i & \sum_{i=1}^{n}(x_i^2 + z_i^2) & -\sum_{i=1}^{n} y_i z_i \\ -\sum_{i=1}^{n} x_i z_i & -\sum_{i=1}^{n} y_i z_i & \sum_{k=1}^{n}(x_i^2 + y_i^2) \end{bmatrix} \tag{2-9}$$



(a) (b)

Figure 2.5. Pedestrians at (a) 10m (b) 37m to the sensor.

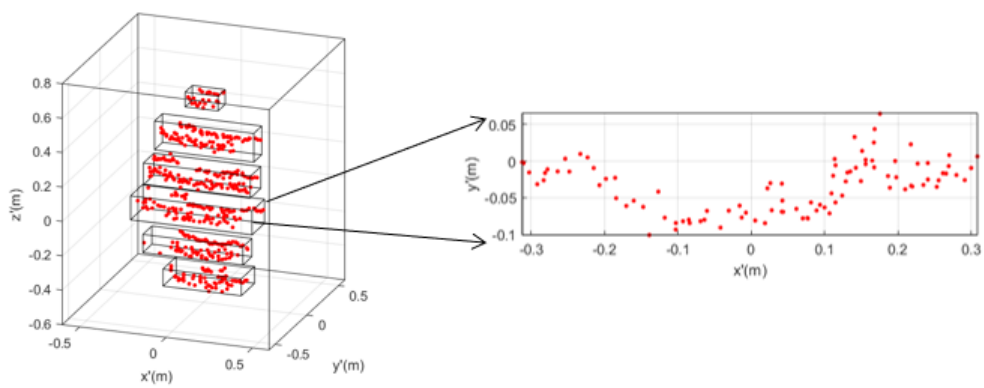


Figure 2.6. The slice feature determined through voxels.

Feature $f_{15} - f_{26}$, named the slice feature, which was first proposed by Kidono et al. (2011), is used to describe the shape of a human body. As the distance to the sensor increases, some of the features of pedestrian clusters are less detailed, but the general contour of the human body from the head to the feet does not change a lot. As shown in Figure 2.6, the target cluster is first divided into six voxels along the height direction. Then, for each voxel, a 3D bounding box is generated for the points located in this space. The length and width of the ith bounding box are l_i and w_i so this feature could be expressed as:

$$f_{15-26} = (l_1, w_1, \dots, l_6, w_6) \tag{2-10}$$

If the point cloud of a pedestrian is too sparse, some elements of this feature will be zero. To address this issue, the concave hull, which is the non-convex enclosure on a set of points, is created for such point clouds to estimate their rough shapes.

$f_{27} - f_{28}$ are the mean and standard deviation of the intensities of the candidate cluster.

2.3.2 Classification methods

Two different strategies were adopted in this thesis to utilize the speed information. The first strategy generates a simple speed classifier based on the average radial velocity v_r of the candidate clusters. The speed classifier discriminates the pedestrian candidates in the test datasets based on their speed information; and all the pedestrian candidates whose average radial velocity satisfies a pre-defined velocity threshold v_T are classified as pedestrians. Then a machine learning-based classifier trained by the static features is utilized to classify the remaining clusters. Another method is to count v_r as f_{29} and merge it with other features to form a 29-dimensional feature vector. Then, the machine learning-based classifier trained by this feature vector is utilized to classify the pedestrian candidates in the test datasets.

In this thesis, the performance for pedestrian detection of two different machine learning algorithms (MLAs) is evaluated: SVM and random forest (RF). These two MLAs were selected for the following reasons: (1) they are easy to implement and able to reproduce the results as it is found in most machine learning algorithm libraries, and (2) many previous studies related to 3D LiDAR-based object classification were implemented by them and have shown good performance (Navarro et al. (2017); Yan et al. (2017); Luo et al. (2021)).

SVM is a popular machine learning algorithm first proposed by Cortes and Vapnik (1995). The objective of SVM is to find the optimal separable hyperplane by maximizing the margin of the data to be separated. Given n training samples with feature vector $\boldsymbol{x}$ and label y, the decision surface of an SVM classifier can be expressed as:

$$\boldsymbol{w}^T\boldsymbol{x} + b = 0 \tag{2-11}$$

where $\boldsymbol{w}$ is the normal vector to the hyperplane and b is the intercept term. Thus, the objective function of a hard-margin SVM can be calculated as follows:

$$\min_{(\boldsymbol{w},b)} \frac{1}{2}\|\boldsymbol{w}\|^2 \quad s.t.\ \ y_i(\boldsymbol{w}^T\boldsymbol{x_i} + b) \geq 1, i\ = 1,2,\dots,n \tag{2-12}$$

The soft-margin SVM allows some training samples to violate the constraint in Equation 2-12 by introducing a penalty cost C and slack variables, $\xi_1, \ldots, \xi_n$, then Equation 2-12 can be modified to:

$$\min_{(\boldsymbol{w},b)} \frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{i=1}^{n}\xi_i \qquad s.t.\ \ y_i(\boldsymbol{w}^T\boldsymbol{x_i} + b) \geq 1 - \xi_n, i\ = 1,2,\ldots,n \tag{2-13}$$

The dual problem of the soft-margin SVM is:

$$\sum_{i=1}^{n}\alpha_i - \frac{1}{2}\sum_{i=1}^{n}\alpha_i\alpha_j {y_i}^T y_j {\boldsymbol{x}_i}^T\boldsymbol{x}_j \quad s.t. \quad \sum_{i=1}^{n}\alpha_i y_i = 0,\ 0 \leq \alpha_i \leq C,\ i = 1,2,\ldots,n \tag{2-14}$$

When SVM is used to perform a non-linear classification, the input data are mapped to a high-dimension feature space by a kernel function $\boldsymbol{K}_\theta(\boldsymbol{x}_i, \boldsymbol{x}_j)$. Common kernel functions include polynomial basis function, sigmoid function, and radial basis function (RBF). This thesis chose RBF as the kernel function:

$$\boldsymbol{K}_\theta(\boldsymbol{x}_i, \boldsymbol{x}_j) = \exp(-\gamma|\boldsymbol{x}_i - \boldsymbol{x}_j|^2), \qquad \gamma > 0 \tag{2-15}$$

where γ is the kernel parameter.

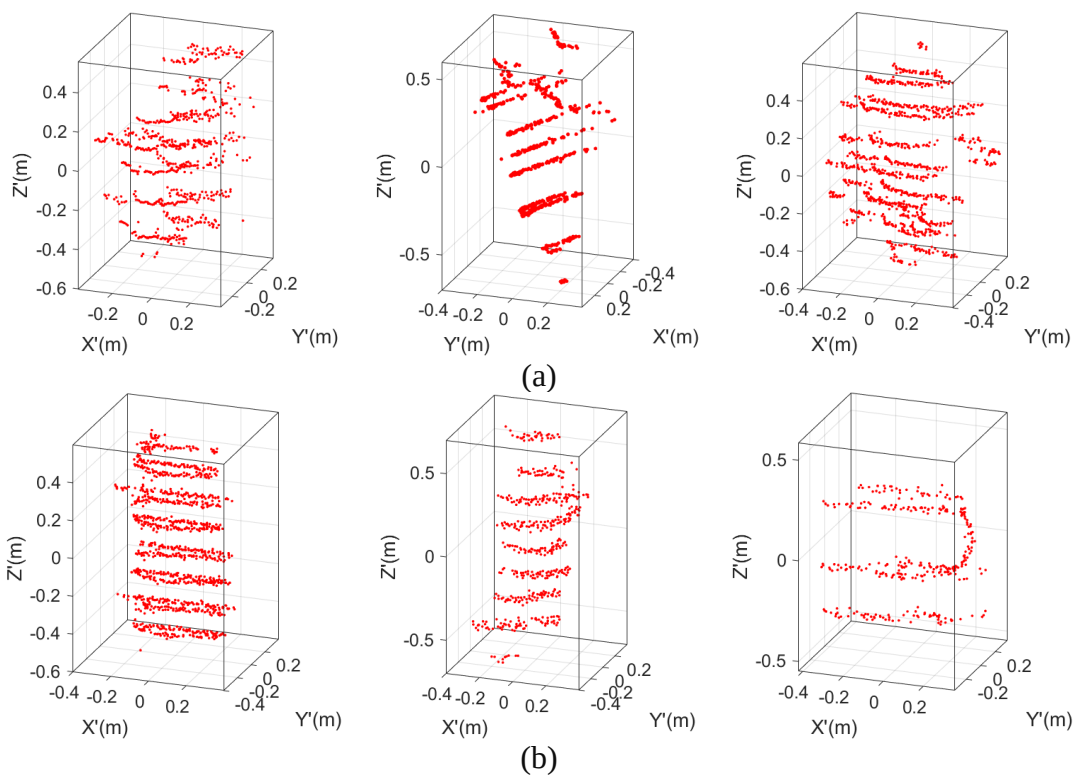


Figure 2.7. Some training samples for pedestrian classification: (a) positive samples; (b) negative samples (from left to right: a sign board, a part of a building and a part of a vehicle).

The RF classifier is composed of a set of decision trees (Shalev-Shwartz and Ben-David 2014). For a given sample, each decision tree would give its class prediction and the class with the most votes becomes the RF prediction. Generally, the RF classifier requires two parameters: the number of trees M and the number of features N. For each individual decision tree, it randomly selects N features from all features and makes decisions by splitting nodes into sub-nodes. The priority of the features used to split the tree is usually determined by the splitting measures like Gini Index and information gain (Zhang et al. 2020).

Before training a classifier, both positive and negative samples are needed. As can be seen from Figure 2.7, in urban road environment, the positive samples are pedestrians and the negative samples include parts of buildings, road signs, etc.

2.3.3 Evaluation metrics

There are several metrics that can be used to evaluate the performance of SVM; and these values can be calculated based on the confusion matrix.

		Actual class	
		Positive	Negative
Predicted class	Positive	TP	FP
	Negative	FN	TN

Figure 2.8. The confusion matrix

As can be seen from Figure 2.8, the confusion matrix presents the number of samples that are correctly classified by a machine learning algorithm (true positives (TP) and true negatives (TN)) against those which are not (false positives (FP) and false negatives (FN)). As pedestrian detection requires a binary classifier, several values can be calculated from the confusion matrix, such as accuracy, precision, recall and F1-score. Specifically, accuracy is the proportion of correctly classified samples and can be expressed by:

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \tag{2-16}$$

Precision is the predictive power of the algorithm to evaluate its effectiveness in detecting a single class. It can be expressed as:

$$Precision = \frac{TP}{TP+FP} \tag{2-17}$$

Recall evaluates the effectiveness of the algorithm in detecting a single class and can be expressed as:

$$Recall = \frac{TP}{TP+FN} \tag{2-18}$$

F1-score is the harmonic mean of precision and recall and can be expressed as:

$$F1\text{-}score = \frac{2 \times Precision \times Recall}{Precision + Recall} \tag{2-19}$$

3. PEDESTRIAN TRACKING

The objective of the multiple objects tracking stage is to separate pedestrians from each other and associate them with existed trajectories. Since this thesis adopted a sequential method, at every frame only the detected pedestrians at the current frame k and associated trajectories from the previous frame $k-1$ are required. At frame k, all the M detected pedestrians can be formulated as $D^k = (D_1^k, D_2^k, D_3^k, \dots, D_M^k)$ where D_i^k is the i-th pedestrian in this frame. Similarly, all the N associated trajectories from the previous frame $k-1$ can be denoted as $T^{k-1} = (T_1^{k-1}, T_2^{k-1}, T_3^{k-1}, \dots, T_N^{k-1})$ where T_i^{k-1} is the i-th associated trajectories.

The workflow of the proposed pedestrian tracking approach, which is illustrated in Figure 3.1, consists of the following steps: (1) the detection step provides D^k at the current frame k; (2) the Kalman filter predicts the state of T^{k-1} to the current frame; (3) the data association steps matches the detections with their predicted trajectories; (4) the birth and death memory step initializes new trajectories and eliminates unmatched trajectories; and (5) the Kalman filter updates the states of the pedestrians based on their predicted and measured states.

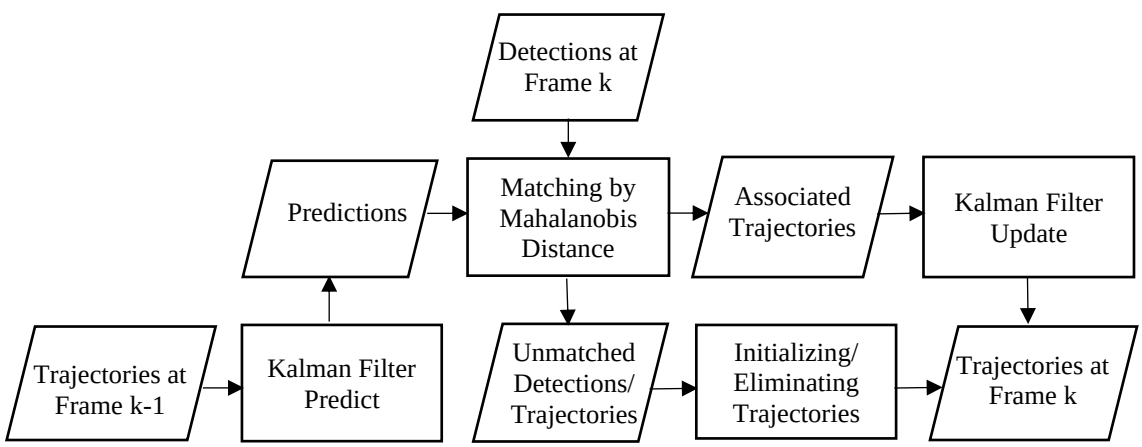


Figure 3.1. Flowchart of the proposed approach for pedestrian tracking.

3.1 Kalman filter for state prediction

The proposed pedestrian tracking algorithm was built on a Kalman filter (Yilmaz and Shah, 2006). A Kalman filter is a probabilistic inference model that estimates the state of a system from

predictions and measurements and produces values closer to the real values by analyzing the uncertainty of the predictions and measurements. These uncertainties are factored in when making final decisions on the values of the current state. In other words, the Kalman filter assumes that either the prediction or measurement is not perfectly accurate. Combining the predictions and measurements increases the accuracy of the state description. In the prediction step, a constant velocity model is adopted to describe the evolution of the state of T^{k-1} from the previous frame to the current frame to generate the predicted detections. Then, the predicted detections are associated with the detections provided by the pedestrian classifier in the current frame. Finally, the state vectors of the matched pairs of tracks and detections are combined to update the current object state estimates.

3.1.1 Pedestrian state

In the detection step, one assumption is that pedestrians in urban scenarios remain upright, so pedestrians move in the xy plane. Thus, the state vector μ of a detected pedestrian consists of eight parameters:

$$\mu = (x, y, w, l, h, \theta, v_x, v_y)^T \tag{3-1}$$

where (x, y) are the xy coordinates of a pedestrian's position; (w, l, h, θ) represent the width, length, height, and orientation of the pedestrian's 3D bounding box, respectively; (v_x, v_y) represent the velocity of the pedestrian in xy direction.

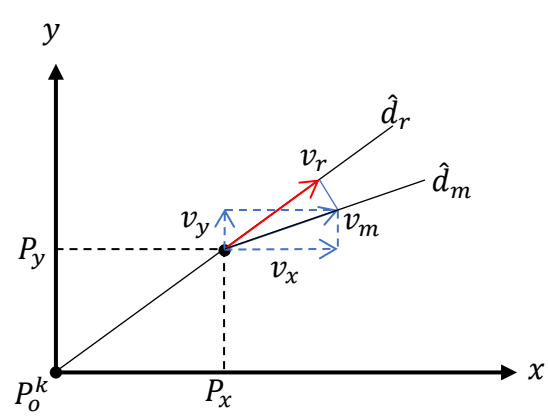


Figure 3.2. Estimate the speed in x, y direction with the measured radial velocity.

In detail, the position of a pedestrian is computed by using the mean position of all the points belonging to this detected pedestrian, and the width, length, height, and orientation can be derived from its 3D bounding box. Velocity v_m is estimated by the measured radial velocity v_r, the scanner position P_o^k, and the pedestrian's moving direction $\hat{d}_m$. As can be seen in Figure 3.2, for a given detection D_i^k with its position P_i^k, the vector of beam direction can be expressed as:

$$\hat{d}_r = P_i^k - P_o^k \tag{3-2}$$

Thus, the total speed of a pedestrian can be formulated as:

$$\|v_m\| = \frac{\|\hat{d}_m\|\|\hat{d}_r\|}{\hat{d}_m \cdot \hat{d}_r} \|v_r\| \tag{3-3}$$

3.1.2 Process model

The process model defines the propagation of the state from the previous frame to the current frame. To model the dynamics of pedestrians, this thesis adopts a constant velocity model as the process model. All the associated trajectories T^{k-1} from frame $k-1$ are propagated to frame k by the following equation:

$$\begin{cases} x^k = x^{k-1} + v_x^{k-1}\Delta t + \frac{1}{2}a_x^{k-1}\Delta t^2 \\ y^k = y^{k-1} + v_y^{k-1}\Delta t + \frac{1}{2}a_y^{k-1}\Delta t^2 \end{cases} \tag{3-4}$$

where $\left(a_x^{k-1}, a_y^{k-1}\right)$ are the linear accelerations, which are regarded as process noise in this model, Δt is the time interval between two adjacent frames (0.2 s in this thesis). Also, the dimensions and orientation of a pedestrian are assumed to be constant, so they do not change during the prediction step. Then, the Kalman filter prediction step is formulated in matrix form as:

$$\bar{\mu}^k = F\mu^{k-1} + \omega \tag{3-5}$$

$$\bar{\Sigma}^k = F\Sigma^{k-1}F^T + Q \tag{3-6}$$

where μ^{k-1} and Σ^{k-1} are the state vector and its covariance matrix at frame $k-1$; $\bar{\mu}^k$ and $\bar{\Sigma}^k$are the predicted state and the covariance matrix at frame k; F is the state transition matrix, and ω is the process noise vector that is assumed to be zero-mean Gaussian with the covariance Q. F, ω, Q can be expressed by:

$$F = \begin{bmatrix} I_{2\times2} & \Delta t I_{2\times2} & I_{2\times4} \\ 0_{2\times2} & I_{2\times2} & I_{2\times4} \\ 0_{4\times3} & 0_{4\times2} & I_{4\times4} \end{bmatrix} \tag{3-7}$$

$$\omega = \left[\frac{1}{2}a_x^t\Delta t^2, \frac{1}{2}a_y^t\Delta t^2, a_x^t\Delta t, a_y^t\Delta t, p_w, p_l, p_h, p_\theta\right]^T \tag{3-8}$$

$$Q = \begin{bmatrix} \frac{1}{4}\sigma_a^2\Delta t^4 I_{2\times2} & \frac{1}{2}\sigma_a^2\Delta t^3 I_{2\times2} & 0_{2\times4} & & & \\ \frac{1}{2}\sigma_a^2\Delta t^3 I_{2\times2} & \sigma_a^2\Delta t^2 I_{2\times2} & 0_{2\times4} & & & \\ & & \sigma_{p_w}^2 & 0 & 0 & 0 \\ & & 0 & \sigma_{p_l}^2 & 0 & 0 \\ 0_{4\times2} & 0_{4\times2} & 0 & 0 & \sigma_{p_h}^2 & 0 \\ & & 0 & 0 & 0 & \sigma_{p_\theta}^2 \end{bmatrix} \tag{3-9}$$

where σ_a^2 is the variance of acceleration, p_w, p_l, p_h, p_θ and $\sigma_{p_w}^2$, $\sigma_{p_l}^2$, $\sigma_{p_h}^2$, $\sigma_{p_\theta}^2$ are the process noise and its variance of w, l, h, θ. Since their variance could not be modeled by the constant velocity model, the strategy proposed by (Weng et al. 2019) is adopted to determine these values. That is, the variance of process noise of w, l, h and θ are same as the variance of process noise of x and y.

3.2 Data association

In frame k, M pedestrians are detected by the proposed detection algorithm and the state vector of N existed trajectories are propagated from frame $k-1$. Then, a data association strategy is designed to decide which detected pedestrian belongs to which track. For this purpose, the similarities between detections and tracks are calculated and the optimal pairing is determined by a greedy algorithm. After this process, a birth-and-death step (Kampker et al. 2018) is adopted to deal with unmatched detections and tracks by initializing new tracks and terminating unmatched tracks.

3.2.1 Measurement model

For each detected pedestrian, the sensor can provide its position and velocity so the dimension of measurement vector of a detected pedestrian would be the same as its state vector and the measurement model is:

$$\bar{z}^k = H\bar{\mu}^k + v \tag{3-10}$$

$$S^k = H\bar{\Sigma}^k H^T + R \tag{3-11}$$

where $\bar{Z}^k$ and S^k are the predicted measurement and its innovative matrix; H is the measurement matrix, and ν is the measurement noise vector that is assumed to be zero-mean Gaussian with the covariance R. H and R could be expressed as:

$$H = I_{8\times 8} \tag{3-12}$$

$$R = \begin{bmatrix} \sigma_p^2 I_{2\times 2} & 0_{2\times 2} & 0_{2\times 4} & & & \\ 0_{2\times 2} & \sigma_v^2 I_{2\times 2} & 0_{2\times 4} & & & \\ & & \sigma_{m_w}^2 & 0 & 0 & 0 \\ & & 0 & \sigma_{m_l}^2 & 0 & 0 \\ 0_{4\times 2} & 0_{4\times 2} & 0 & 0 & \sigma_{m_h}^2 & 0 \\ & & 0 & 0 & 0 & \sigma_{m_\theta}^2 \end{bmatrix} \tag{3-13}$$

where σ_p and σ_v are the position and velocity variance; m_w, m_l, m_h, m_θ and $\sigma_{m_w}^2$, $\sigma_{m_l}^2$, $\sigma_{m_h}^2$, $\sigma_{m_\theta}^2$ are the measurement noise and its variance of w, l, h and θ. Similarly, the variance of measurement noise of these four parameters are same as the variance of measurement noise of x and y.

3.2.2 Bipartite matching

To associate D^k at frame k with the tracks from T^{k-1} frame $k-1$, the Mahalanobis distance (Maesschalck et al. 2000) is adopted to measure the similarity between the predicted detections from the trajectories and measured detections. The Mahalanobis distance between the predicted measurement of a trajectory $\bar{z}_i^k$ and the real measurement of a detection z_j^k is formulated by the following equation:

$$d_m^k(i,j) = \sqrt{\left(z_j^k - \bar{z}_i^k\right)^T \left(S_i^k\right)^{-1} \left(z_j^k - \bar{z}_i^k\right)} \tag{3-14}$$

where $d_m^k(i,j)$ is the Mahalanobis distance between the predicted detection i and the measured detection j. Given the Mahalanobis distance between each pair of predictions and detections, a greedy algorithm (Algorithm 1) is used to match the detections and trajectories.

Algorithm 1: Greedy Algorithm for Data Association (Chiu et al. 2020)

```
Input:
M detections at current frame
N associated trajectories from previous frame
Similarity threshold ST
Output:
Associated detections and trackings
Initialize
MatchedD = ∅
MatchedT = ∅
Similarity = ∅
MatchedPairs = ∅
for i in 1 to M:
        for j in 1 to N:
                Similarity[i][j] = Mahalanobis Distance(Detection(i), Track(j))
        end
end
ListofPairs < −PairSortByValue(Similarity)
for k in 1 to length(ListOfPairs) :
        (i, j) = ListofPairs[k]
        if i ∉ MatchedD and j ∉ MatchedT:
                        if Similarity[i][j] < ST:
                        MatchedD.append(i)
                        MatchedT.append(j)
                        MatchedPairs.append(i, j)
                end
        end
end
return MatchedPairs
```

3.2.3 Initialization and termination

The proposed initialization and termination module is used to initialize new tracks for the unmatched detections and to eliminate unmatched tracks. To avoid false positive tracks, a new track is not created for an unmatched pedestrian until it is detected in three consecutive frames. The state vector of a new track is calculated according to its most recent detection. Similarly, to avoid terminating the true tracks which miss detections at some frames, each track is not deleted until it cannot be matched with any detections in three consecutive frames. During this process, the state vector of the unmatched pedestrians is also updated with its last known velocity estimation by the process model. The tracks with missing detections are recovered by this strategy and only

the tracks that leave the scene are terminated. An example is shown in Figure 3.2. The pedestrian is detected in the first and third frames while it is missed in the second frame. In the first frame, its coordinate and velocity in xy-plane is $(-15.80\,m, 5.88\,m)$ and $(-0.75m/s, 0.53m/s)$, respectively. In the second frame, the location of pedestrian is predicted by Equation 3-4 and changed to $(-15.95\,m, 5.98\,m)$ and this track could not be matched by any detections. However, this track is not terminated immediately but propagated to the third frame by the same estimated velocity. In the third frame, the predicted location is $(-16.10\,m, 6.08\,m)$, which can be matched with the detected pedestrian whose measured coordinate center is $(-16.03\,m, 6.04\,m)$.

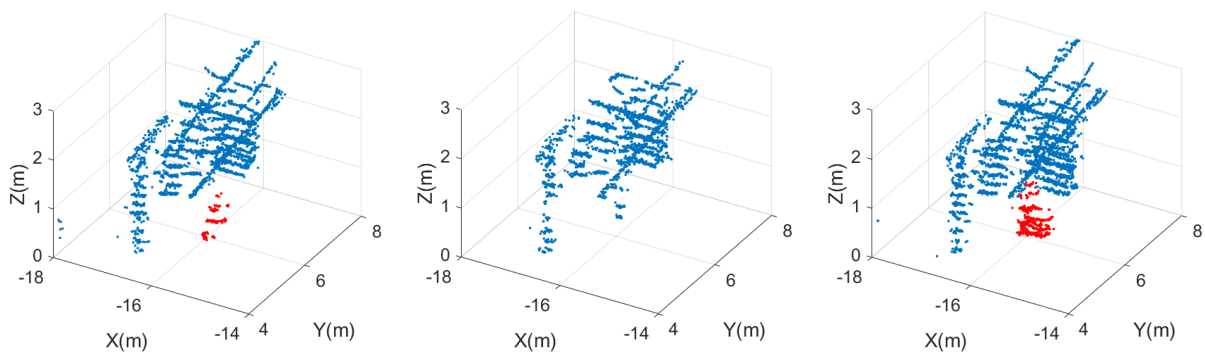


Figure 3.3. Recovery of missing detections. (This pedestrian is not scanned in the second frame, but his track is not deleted immediately and recovered when he appears again).

3.3 Kalman filter state update

Once all the tracks are associated, the predicted state mean and its covariance matrix are updated by the following equations:

$$y^k = z^k - H\bar{\mu}^k \tag{3-15}$$

$$\mu^k = \bar{\mu}^k + K^k y^k \tag{3-16}$$

$$\Sigma^k = \bar{\Sigma}^k (I - K^k H) \tag{3-17}$$

where y^k is the measurement residual, K^k is the optimal Kalman gain which can be calculated by:

$$K^k = \bar{\Sigma}^k H^T (R + H\bar{\Sigma}^k H^T)^{-1} \tag{3-18}$$

The Kalman gain is the relative weight given to the measurements and the current state estimate. With a high gain, the filter places more weight on the most recent measurements, and thus follows them more responsively. With a low gain, the filter follows the model predictions more closely (Savtchenko 2011).

3.4 Evaluation metrics

There are several metrics that can be used to evaluate the performance of the proposed tracking approach: FP, FN, identity switch (IDSW), and MOTA. A pedestrian that is missed by any hypothesis is a FN and a non-pedestrian that is wrongly assigned to a track is a FP. An IDSW is counted if a ground truth target i is matched to an incorrect track j. MOTA is used to evaluate a tracker's overall performance by combining all the above sources of errors and could be expressed as:

$$MOTA = 1 - \frac{\sum_K (FN_k + FP_k + IDSW_k)}{\sum_k GT_k} \tag{3-19}$$

where k is the frame index and GT is the number of ground truths.

4. DOPPLER LIDAR AND DATASET

4.1 Doppler LiDAR

LiDAR is a remote sensing technique that uses visible or near-infrared laser energy to measure the distance between the sensor and an object. Currently, one of the most commonly used LiDAR techniques is called pulsed LiDAR, or linear mode LiDAR, which emits short but intense pulses of laser radiation and measures the distance to a target by recording the time intervals between the transmitted and received light pulses (Royo and Ballesta-Garcia 2019). With the constant speed of light c, the distance R to the object is directly proportional to the two-way traveled time t:

$$R = \frac{1}{2}ct \tag{4-1}$$

Although pulse LiDAR is a very popular and mature light measurement approach, it has some drawbacks. Many linear mode LiDAR systems are operated at 905 nm wavelength, which is not safe for the human eye and is likely to be affected by ambient light such as bright sunlight, other sensor's light pulse and its own previous pulse. An alternative frequency modulated continuous wave (FMCW) LiDAR system has been proposed using a wavelength in the 1550 nm band, which has the advantage of lower peak power when compared to pulse technology and is safe for human eyes (Kim et al. 2020).

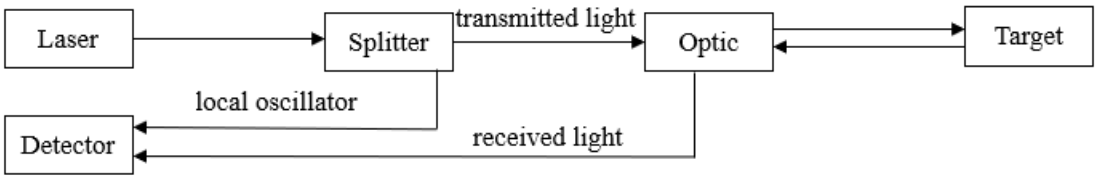


Figure 4.1. A standard interferometric circuit of Doppler LiDAR (adapted from Kadlec et al. 2019).

Instead of emitting a pulse, FMCW LiDAR emits a beam of coherent radiation to a target while keeping a reference signal, also known as a local oscillator (Kadlec et al. 2019). The reference signal plays a crucial role in the operation of the FMCW LiDAR. First, it provides a very stable reference frequency that can be used to determine velocity accurately. It also rejects

radiation from other sources, such as sunlight, to eliminate the effects of background light completely. As shown in Figure 4.1, a continuous beam of radiation illuminates the target, and a small fraction of the light is backscattered into the receiver. The motion of the target along the beam direction leads to a change of light's frequency due to the Doppler shift (Piggott, 2020); and movement towards the LiDAR brings about a compression of the wave increasing in its frequency, while movement away stretches the wave and reduce its frequency. As Figure 4.2 shows, the difference of outgoing and incoming frequency derives two beat frequencies f_1 and f_2, which could be utilized to calculate the range R and radial velocity v_r.

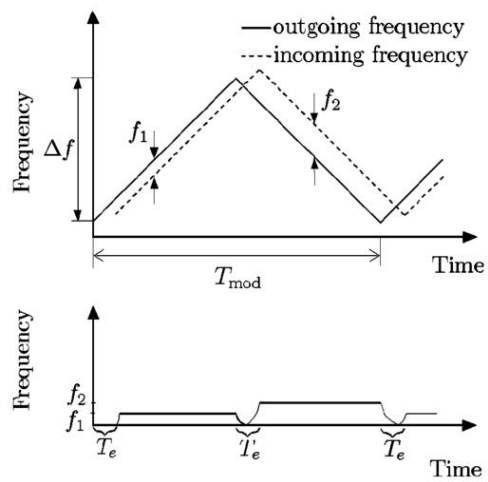


Figure 4.2. Modulation of Doppler LiDAR (adapted from Nordin and Kalevi 2002).

If the range and velocity of a given target are constant during one modulation period, f_1 and f_2 can be used to calculate the magnitude of the frequency difference f_R and the Doppler shift f_D (Nordin and Kalevi 2002). If $f_R \leq |f_D|$, f_R and f_D can be expressed by:

$$f_R = \frac{f_1 + f_2}{2} \tag{4-2}$$

$$f_D = \frac{f_2 - f_1}{2} \tag{4-3}$$

If $f_R > |f_D|$, f_R and f_D can be expressed by:

$$f_R = \left|\frac{f_2 - f_1}{2}\right| \tag{4-4}$$

$$f_D = \begin{cases} \frac{f_1+f_2}{2}, & f_1 \leq f_2 \\ -\left(\frac{f_1+f_2}{2}\right), & f_1 > f_2 \end{cases} \tag{4-5}$$

Then the range R and radial velocity v_r can be calculated by:

$$R = \frac{c \cdot f_R \cdot T_{mod}}{4\Delta f} \tag{4-6}$$

$$v_r = \frac{f_D \cdot \lambda}{2} \tag{4-7}$$

where T_{mod} is the modulation period; Δf is the bandwidth of the frequency sweep; and λ is the optical wavelength. According to Equation 4-9, a positive velocity indicates the object is moving to the sensor along the beam direction while a negative velocity suggests the object is moving away from the sensor along the sensor's line of sight.

In summary, the FMCW LiDAR indirectly measures both the distance and velocity from the Doppler effect and therefore also is called Doppler LiDAR.

4.2 Test datasets

The datasets in this thesis were collected by a mobile Doppler LiDAR system installed on top of a vehicle with GPS and an inertial measurement unit (IMU) designed for autonomous driving or mapping. The Doppler LiDAR system consists of four co-registered Doppler LiDAR scanners; and the horizontal scanning angle range for each scanner is 40º (Figure 4.3(a)). Considering the overlapping areas between adjacent sensors, the four scanners provide a 120º scanning angle in total. The scanning frequency is 5 Hz, and the maximum scanning range is about 400 m. The mobile Doppler LiDAR scans were collected in the downtown area of San Francisco, California in September 2018 (Figure 4.2(b)). A sample of one frame of the point cloud is shown in Figure 4.3(c). The color displayed in this frame is determined by the adjusted radial velocity. Yellow indicates the static points while red and blue represent objects that move to or away from the sensor, respectively.

(a) (b) (c)

Figure 4.3. (a) The Doppler LiDAR system; (b) the test site (blue dot: the sensor; red arrow: moving direction); (c) one frame point cloud color coded by adjusted radial velocity (unit: m/s).

Table 4.1. Attributes of the collected point cloud

Attribute	Unit
Range (R)	m
Azimuth Angle (φ)	rad
Polar Angle (θ)	rad
Relative Position (x, y, z)	m
World Position (x_{world}, y_{world}, z_{world})	m
Relative speed (v)	m/s
Absolute speed (v_{abs})	m/s

The attributes of collected point clouds are shown in Table 4.1. R, φ and θ represent the polar coordinates of a point in the mobile Doppler LiDAR system. The data points are transformed into a 3D Cartesian coordinate system by the following equation:

$$X = R \cdot sin\theta \cdot cos\varphi$$
$$Y = R \cdot sin\theta \cdot sin\varphi \tag{4-8}$$
$$Z = R \cdot cos\theta$$

where x, y and z represent the 3D coordinates of a point in the sensor coordinate system. With the GPS/IMU built in the mobile LiDAR system, x, y and z are transformed to the absolute x_{world}, y_{world} and z_{world}, which are the 3D coordinates of a point in a world or global coordinate system.

v is the relative radial velocity between the object and the sensor. After eliminating the effects of the moving Doppler LiDAR system, v_{abs} is derived, representing the radial velocity of the point relative to the ground.

Two datasets were collected for this thesis. Dataset I has a total of 90 frames. The vehicle equipped with Doppler LiDAR first moved fast along a street and then stopped at a crosswalk. Dataset II has a total of 107 frames and the mobile LiDAR system was constantly moving on a busy street. Approximately 1,800 pedestrians in these two datasets are labeled manually to evaluate the performance of the proposed pedestrian detection and tracking method.

5. RESULTS AND EVALUATION

This chapter presents the results of the proposed pedestrian detection and tracking with the two datasets introduced in Chapter 4. The performance of the pedestrian detection is evaluated by the recall, precision and F1-score. Additionally, this chapter shows that the performance of pedestrian classification is clearly varied by distance to the target. The results of the pedestrian tracking are presented in this chapter as well while its performance is evaluated by a set of tracking quality measures.

5.1 Pedestrian detection

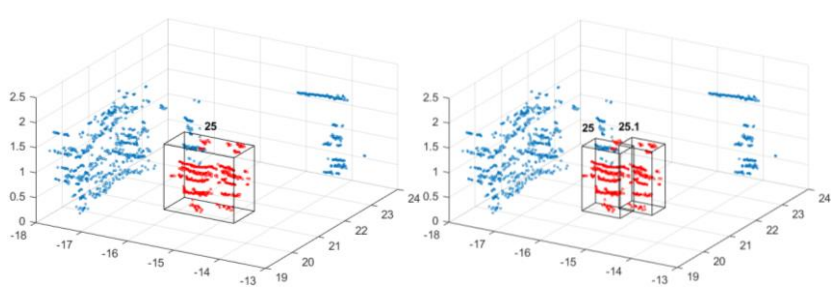
5.1.1 Cluster detection

The objective of cluster detection is to find all the clusters likely to be pedestrians. Therefore, the pedestrian candidates should contain ground truth as much as possible since only the clusters extracted from the clustering algorithm would be regarded as pedestrian candidates and then would be classified by the classier. The size thresholds for potential single pedestrian candidates are:

$$C_p = \{C_i | 0.8m < h_i < 2m, 0.2m < l_i < 1.2m, 0.2m < w_i < 0.8m\} \quad (5\text{-}1)$$

Above criteria are determined according to the height, shoulder width and step length of pedestrians. Based on these values, the size thresholds for potential multiple (usually two or three) pedestrians' candidates are:

$$C_p = \{C_i | 0.8m < h_i < 2m, 1.2m < l_i < 3m, 0.8m < w_i < 3m\} \quad (5\text{-}2)$$



(a) (b)

Figure 5.1. Segmentation of two pedestrians: (a) before re-clustering; (b) after re-clustering.

In this thesis, close pedestrians are segmented into single pedestrians successfully in the re-clustering process. As Figure 5.1(a) shows, two pedestrians very close to each other, are far from the scanner. Therefore, a relatively large ε value is assigned to them during the clustering process, so they are clustered as one object. After the re-clustering process (Figure 5.1(b)), however, they are separated successfully.

Recall, which is the ratio between the number of pedestrians extracted by the clustering process and the ground truth, is adopted to quantitatively evaluate the performance of the proposed cluster detection algorithm. As shown in Table 5.1, the recall for both datasets is larger than 92%, which indicates that less than 8% of the pedestrians are not extracted during the process of cluster detection.

Table 5.1. Performance of pedestrian candidate detection (Count of pedestrians)

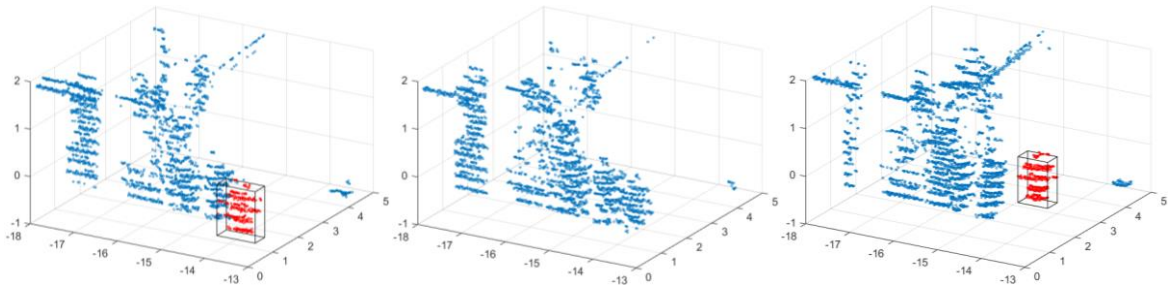
Dataset	#Frames	#Ground truth of pedestrians	#Pedestrians in candidates	#Missing pedestrians	Recall
I	90	1201	1147	54	0.9550
II	107	644	593	51	0.9208

Table 5.2. Summary of missing detections (Count of pedestrians)

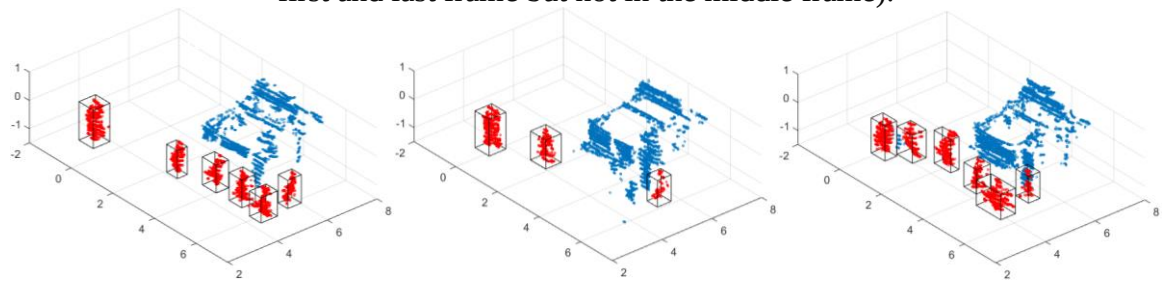
Dataset	# Partially scanned pedestrians	#Mismatched pedestrians	#Unsegmetned pedestrians
I	18	14	22
II	17	16	18

Missing detections, or FNs, are produced by various causes (Table 5.2). To be specific, missing detections can occur in either the clustering or cluster selection processes. In the clustering process, if pedestrians are too close to large objects (e.g. buildings), they may not be segmented as a single pedestrian. Instead, they can be regarded as a part of the large objects. For example, as can be seen from Figure 5.2(a), a pedestrian is passing by a building in three consecutive frames. In the first and last frame, the distance between the pedestrian and the building is larger than ε so this pedestrian is segmented to a single cluster and selected as a pedestrian candidate in the following step. But in the second frame, this distance becomes less than ε so the pedestrian is

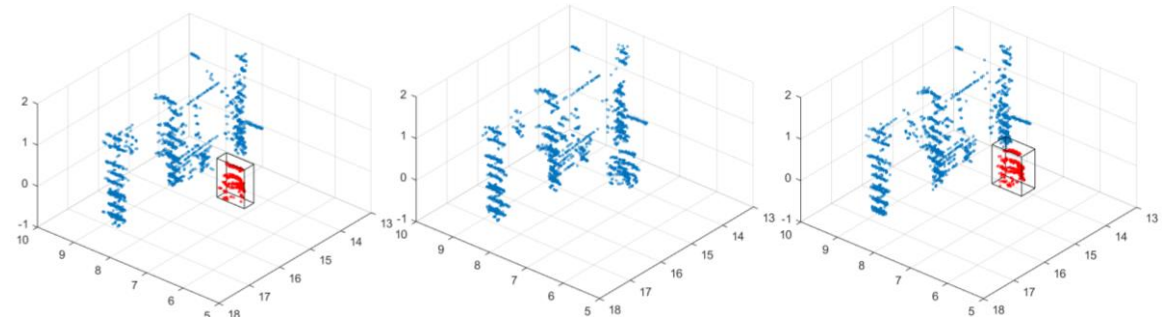
merged into the cluster of the building. In addition, unlike clusters composed of multiple pedestrians, the size of such cluster is much larger than the threshold set for multiple pedestrians' candidates; as a result, this pedestrian is not successfully extracted by the re-clustering process.



(a) Example of an unsegmented pedestrian (This pedestrian is segmented successfully in the first and last frame but not in the middle frame).



(b) Example of partially scanned pedestrians.



(c) Example of a mismatched pedestrian.

Figure 5.2. Examples of missing detections of pedestrians.

In the cluster selection process, both the partially scanned point clouds and the unmatched sizes lead to missing detections. An example of partial scanned point clouds is shown in Figure 5.2(b), where there are six detected pedestrians in the first and last frames. However, in the second frame, the point clouds of three pedestrians are only partially collected by the sensor so they are not selected in the cluster selection process because their sizes are less than the thresholds set

previously. A more extreme case is a pedestrian who totally disappeared in one or more frames. Missing detections also happen when one or more dimensions (w, l, h) of a cluster are not in the range of the predefined threshold. As shown in Figure 5.2(c), a pedestrian is detected successfully in the first and last frames but is missed in the second frame because its height is smaller than the threshold.

5.1.2 Pedestrian classifier

As mentioned previously, dataset I and dataset II have 90 frames and 107 frames of point clouds, respectively. According to the manually-labeled ground truths, there are 1,147 and 593 pedestrians in dataset I and dataset II, respectively. For each dataset, 60 percent of the samples are used as training data and 40 percent of the samples are used as testing data. The descriptions of the two datasets are shown in Table 5.3.

Table 5.3. Summary of training and testing data

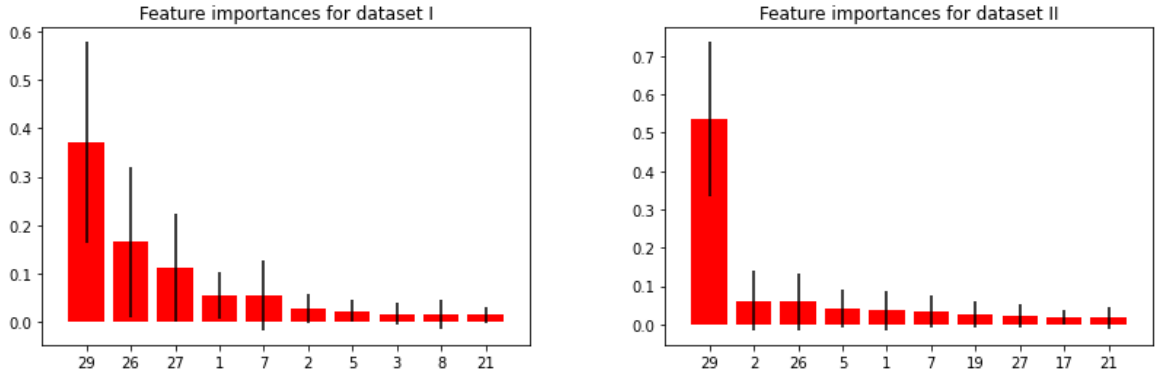
	Description	#Positive Samples	#Negative Samples	Total
Dataset I	Training data	745	1,208	1,953
	Testing data	402	704	1,106
	Description	#Positive Samples	#Negative Samples	Total
Dataset II	Training data	348	1,022	1,370
	Testing data	245	601	846

The proposed two-step classifier is composed of a speed classifier and a machine learning-based classifier. According to the specifications of the Doppler LiDAR (Ma et al. 2019), the precision of the radial velocity is $0.1\ m/s$, so this value is adopted as the lower bound of the v_T of the speed classifier. The upper bound of the v_T is set as $3\ m/s$ because generally the walking speed of a pedestrian is not likely to be more than it. The quantitative results for each step are shown in Table 5.3. After the speed classifier (stage I), the FP rate is less than 0.06 because most of the static objects' average radial velocities are not likely to be larger than the v_T. Most of the FPs are over-segmented vehicles while a few FPs are static objects with large average velocities resulting from measurement error. As previously mentioned, the FNs are pedestrians with zero

measured radial velocity. The remaining clusters are then input to the machine learning-based classifier (stage II). The results show that the FNs decrease substantially because the pedestrians misclassified in the stage I are classified correctly in stage II by the SVM or RF classifier. In contrast, the FPs increase because some of the non-pedestrians are classified as pedestrians incorrectly during this process.

Table 5.4. Quantitative results of each stage of the two-step classifier (Count of pedestrians)

	MLA	Stage	TP	FP	FN	TN
Dataset I	SVM	I	375	4	27	700
		II	389	25	13	679
	RF	I	375	4	27	700
		II	395	20	7	684
Dataset II	MLA	Stage	TP	FP	FN	TN
	SVM	I	219	12	26	589
		II	235	55	10	546
	RF	I	219	12	26	589
		II	230	55	15	546



(a) (b)

Figure 5.3. Significance of features of RF; the feature ID is referred to Table 2.1.

Unlike SVM, it is unnecessary to project the features in RF to higher dimension space by the kernel trick to find the optimal separable hyperplane. Therefore, the feature importance can be ranked according to the GINI index. The importance of features for the RF classifier are shown in

Figure 5.3. The average radial velocity of a cluster (f_{29}) is the most important feature among all the features. Therefore, it is reasonable to use f_{29} as the only feature to form the speed classifier.

To evaluate the effect of the speed classifier as well as the effect of the speed information, we compare the results from two two-step classifiers (named Two-Step SVM and Two-Step RF), two one-step classifiers with the speed information (named SVM-Speed and RF-Speed) and two one-step classifiers without the speed information (named SVM and RF). As can be seen from Table 5.5, all six classifiers perform better on the dataset I because when the vehicle equipped with mobile Doppler Lidar stopped at a traffic light, most of the pedestrians were crossing the street; so they were fully scanned and had a complete shape. In contrast, for dataset II, most of the pedestrians were constantly walking along the street; so they were more likely to be obstructed by other pedestrians or objects such as vehicles and poles, resulting in partially scanned point clouds and incomplete shapes.

Table 5.5. Quantitative results of different pedestrian classifiers

Classifier	Dataset	Accuracy	Precision	Recall	F1-score
Two-Step-SVM	I	0.9497	0.9023	0.9652	0.9327
	II	0.9173	0.8070	0.9388	0.8679
SVM-Speed	I	0.9647	0.9612	0.9403	0.9509
	II	0.9113	0.8295	0.8735	0.8509
SVM	I	0.8843	0.8914	0.7761	0.8298
	II	0.8664	0.8208	0.7143	0.7559
Two-Step RF	I	0.9737	0.9388	0.9825	0.9649
	II	0.9385	0.8561	0.9470	0.8992
RF-Speed	I	0.9792	0.9588	0.9851	0.9718
	II	0.9444	0.8779	0.9388	0.9073
RF	I	0.9141	0.9449	0.8109	0.8728
	II	0.8759	0.8431	0.7020	0.7661

Overall, the four classifiers with speed information outperform the two classifiers without speed information on all the evaluation metrics, especially the recall. The F1-scores among four classifiers with speed information are not significantly different. For SVM-based classifiers, the

Two-Step SVM has higher recall and the SVM-speed has higher precision. This is because the SVM-speed recognizes pedestrians based on both the speed information and the static features so some of the pedestrians with very blurred contours are misclassified even if they are moving. In contrast, the speed classifier is able to recognize moving pedestrians only by their average radial velocity and therefore is able to detect all of the moving pedestrians. For RF-based classifiers, the two-step RF and the RF-speed show similar performance. A possible explanation is that once the subset of features selected by an individual tree including the average radial velocity, the role of this feature is similar to a simple speed classifier. For the classifiers based on different MLAs, the average F1-score of SVM-based and RF-based classifiers are 0.8647 and 0.8970, respectively. Thus, it is concluded that the overall performance of the RF-based classifiers outperforms the SVM-based classifiers. Therefore, the detection results from Two-Step RF can be used to track pedestrians.

The detection results of the classifiers in varied ranges are listed in Tables 5.6 and 5.7, respectively. As shown in Tables 5.6 and 5.7, as the distance between the objects and the sensors increase, the overall performance of most classifiers decreases, indicating a decreasing trend for pedestrian detection efficiency. However, the four classifiers with speed information generally perform better than the classifiers having only static features, especially for pedestrian detection at ranges larger than 30 m. To be specific, for close pedestrian detection, static features are sufficient to detect most pedestrians. The additional speed information increases all the evaluation metrics slightly. Based on these outcomes, it is concluded that both static features and speed information are important to detect pedestrians at the range of 30 m. However, most static features are sensitive to spatial resolution. When the distance increases, the number of points returned as pedestrians decreases so the contour of the candidate pedestrians gradually blurs or is even lost (Figure 5.4), which makes the shapes less different between pedestrians and other objects. Unlike static features, the effect of speed information is not distance-dependent; therefore, as the distance increases, the performance of the two-step classifiers and one-step classifier with speed only decreases slightly. In dataset I, the additional speed information increases all the evaluation metrics by 10%. In dataset II, the two classifiers with only static features thoroughly fail to detect pedestrians in this region because most of the pedestrians that are at the range of 30 m to 50 m do not have a clear shape while the two-step classifier and the one-step classifier with speed

information still perform robustly. Therefore, it is concluded in this thesis that speed information plays a major role in detecting pedestrians, especially when they were far away from the sensor.

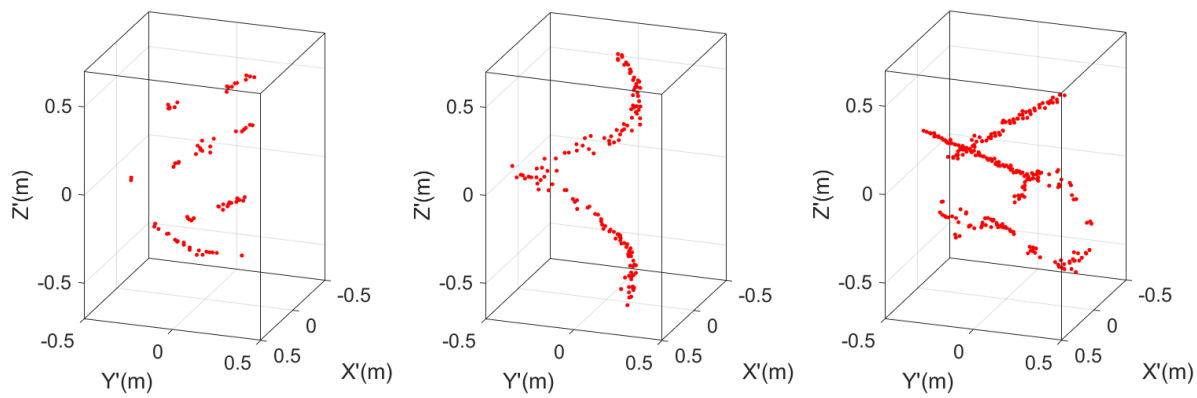


Figure 5.4. Some failure cases in pedestrian detection without the speed (contours of these pedestrians are blurred so they could not be classified correctly by classifiers without speed information).

Table 5.6. Quality of pedestrian detection at the range of 30 m from the sensor

Classifier	Datasets	Total Number	Truly Detected	False Alarms	Precision	Recall	F1-score
Two-Step SVM	I	196	186	15	0.9254	0.9490	0.9370
	II	195	188	37	0.8356	0.9641	0.8952
SVM-Speed	I	196	182	7	0.9630	0.9286	0.9455
	II	195	177	34	0.8389	0.9077	0.8719
SVM	I	196	172	15	0.9198	0.8776	0.8982
	II	195	160	35	0.8205	0.8205	0.8205
Two-Step RF	I	196	186	14	0.9300	0.9490	0.9394
	II	195	189	35	0.8438	0.9692	0.9021
RF-Speed	I	196	191	7	0.9646	0.9744	0.9695
	II	195	85	30	0.8605	0.9487	0.9024
RF	I	196	188	13	0.9353	0.9592	0.9471
	II	195	157	33	0.8263	0.8051	0.8156

In summary, during the clustering process, the proposed method is able to extract most of the pedestrian candidates (over 92%) and separate single pedestrian well. In the pedestrian classification, the average radial velocity improves the classification performance significantly, especially for pedestrians at the range of 30 m to 50 m to the sensor. The performance of RF is slightly better than the SVM. Overall, the highest recall (0.9788 and 0.9450 for dataset I and II respectively) is obtained by using the proposed two-step classifiers while the highest precision (0.9010 and 0.9125 for dataset I and II respectively) is obtained by adopting the one-step classifiers with speed information. For the purpose of pedestrian tracking, a higher recall is more important than precision. As a non-pedestrian may not be misclassified as a pedestrian in all consecutive frames, its influence could be eliminated by an appropriate tracking management strategy. In contrast, pedestrians with only a sparse or partially scanned point cloud are likely to be missed in the pedestrian classification step, which will result in increasing tracking errors.

Table 5.7. Quality of pedestrian detection at the range of 50 m from the sensor

Classifier	Datasets	Total Number	Truly Detected	False Alarms	Precision	Recall	F1-score
Two-Step SVM	I	206	203	10	0.9531	0.9854	0.9690
	II	50	42	18	0.7	0.8400	0.7636
SVM-Speed	I	206	196	8	0.9608	0.9514	0.9561
	II	50	37	10	0.7872	0.74	0.7629
SVM	I	206	158	23	0.8729	0.7667	0.8165
	II	50	15	8	0.6522	0.3000	0.4110
Two-Step RF	I	206	205	7	0.9670	0.9951	0.9809
	II	50	41	14	0.7454	0.8200	0.7810
RF-Speed	I	206	203	4	0.9807	0.9854	0.9831
	II	50	44	10	0.8148	0.8800	0.8462
RF	I	206	147	1	0.9932	0.7136	0.8305
	II	50	15	4	0.7895	0.3000	0.4348

5.2 Pedestrian tracking

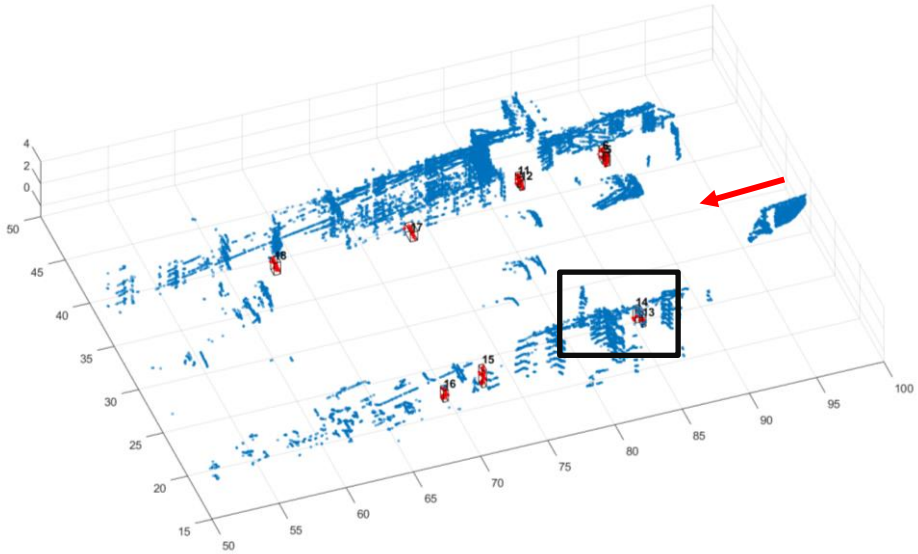
The results of the evaluation of the effects of including speed measurement in the proposed tracking algorithm are shown in Table 5.8. Compared to the original KF, KF-Speed is less likely to have IDSW errors, especially in crowded scenes.

Table 5.8. Tracking quality evaluation

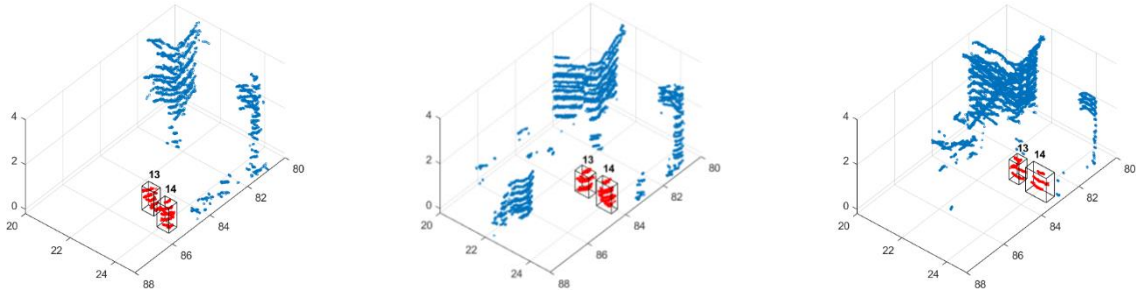
Dataset	Method	FN	FP	IDSW	MOTA
I	KF-Speed	38	15	16	0.7862
	KF	47	21	28	0.7215
II	KF-Speed	28	13	10	0.7353
	KF	35	16	25	0.6774

Some sample results of pedestrian tracking are shown in Figure 5.5 (a). The red clusters represent pedestrians, and the blue points are the static background and other objects. The black boxes represent the 3D bounding box of each tracked pedestrian and the corresponding number is the unique tracking ID of each pedestrian.

The results show that the proposed tracking method can track pedestrians on crowded streets. For example, in Figure 5.5 (b), two pedestrians have been tracked completely over ten frames until they leave the scene. Also, the proposed strategy that manages the initialization and termination of the trajectories is able to deal with the FPs and FNs generated during the detection step. For example, in Figure 5.6, two close pedestrians are shown walking along the street (tracked as #0 and #1). In frame 67 and frame 68(Figure 5.6 (b)-(c)), these two pedestrians are only partially scanned by the sensor and therefore are recognized as one pedestrian (Figure 5.6 (b)-(c)). Thus, in the data association step, Track #0 is not associated with any detected pedestrians, but Track #0 is not terminated immediately according to the strategy for initializing and terminating tracks. Instead, its state vector is updated and propagated to frame 68 and frame 69 by using its current velocity. In frame 69, these two pedestrians are separated well again so Track #0 retracks the corresponding pedestrians successfully (Figure 5.6(d)). As shown in Figure 5.7, five pedestrians are fully tracked in frames 66 through 75 (Figure 5.7(a)). However, in frame 69 and 72, Track #3 and Track #5 are too close to a high pole so that those pedestrians could not be detected successfully (Figure5.7(b)) and are recovered once their corresponding pedestrians are detected again.

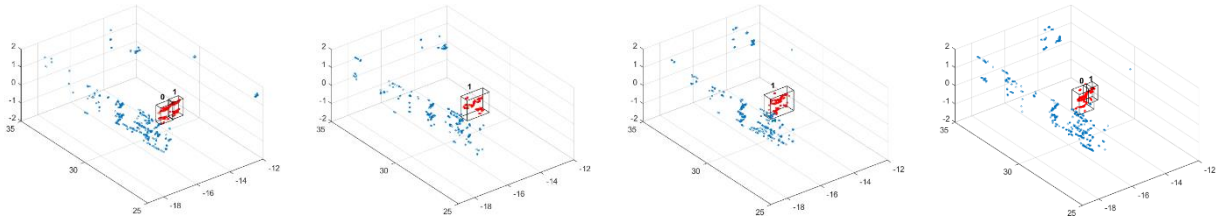


(a)



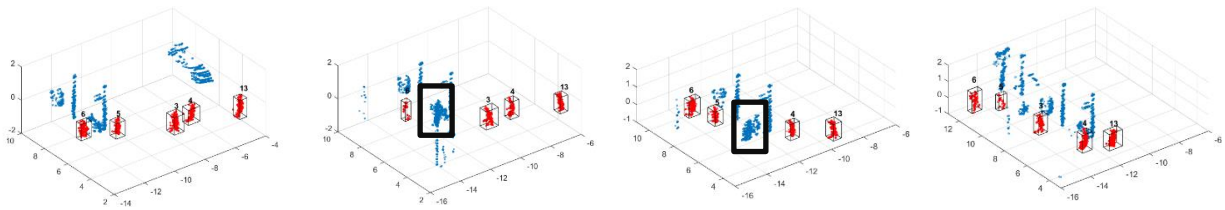
(b)

Figure 5.5. Sample results of pedestrian tracking: (a) tracking results of frame 74 in dataset II (Unit: m) (red arrow: direction of the mobile Doppler LiDAR), (b) Track#13 and #14 in frame 69,74 and 79.

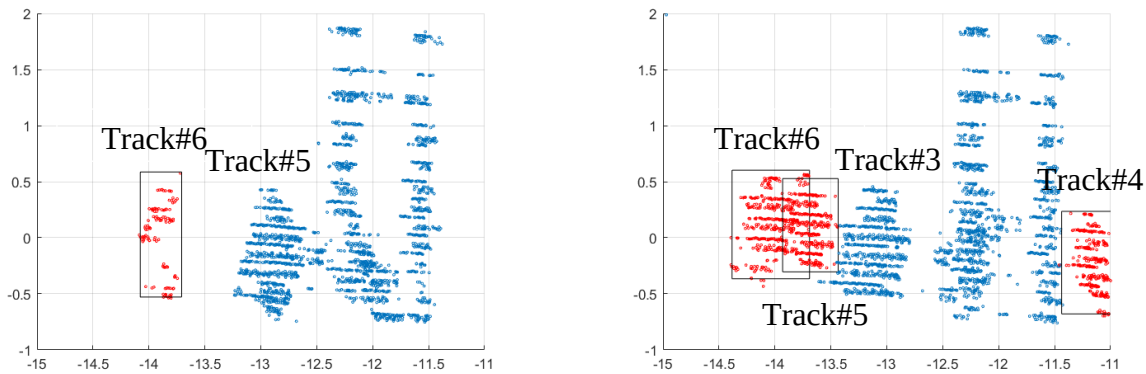


(a) (b) (c) (d)

Figure 5.6. Tracks #0 and #1 in frames 66-69 in dataset I (Unit: m).



(a)

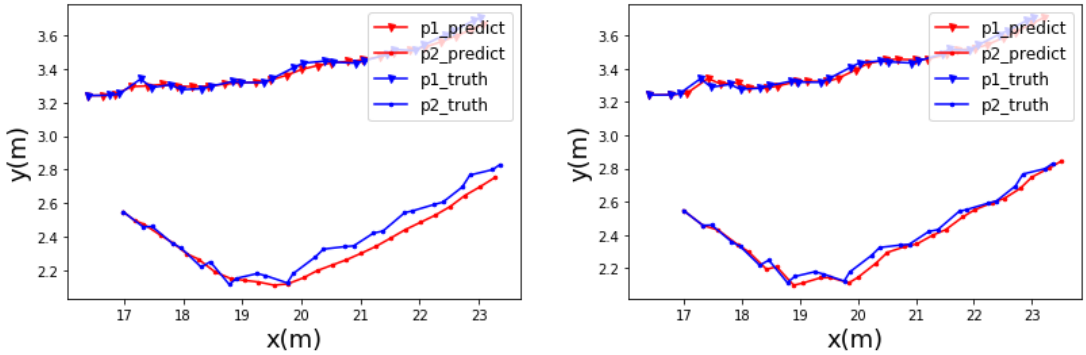


(b)

Figure 5.7. (a)Tracks #3, Track #4, Tracks #5, Tracks #6 and Track #13 in frames 66, 69, 72, 75 in dataset I (Unit: m); (b) the detail views of the frame 69 and 72.

Table 5.9. RMSE between predicted locations and true locations

Pedestrian	Method	RMSE (m)
I	KF	0.145
	KF-Speed	0.113
II	KF	0.180
	KF-Speed	0.137



(a) (b)

Figure 5.8. Birds-eye view tracking visualization of two pedestrians: (a) predicted location without measured speed; (b) predicted location with measured speed.

Figure 5.8 shows the effect of radial velocity by visualizing the trajectories of the two close pedestrians. The red and blue lines represent their predicted and true movement trajectory, respectively. Root Means Square Error (RMSE) is adopted to quantitatively evaluate the accuracy of the predicted location of two KF models and results are shown in Table 5.9. It is obvious that compared to the KF (Figure 5.8(a)), KF-Speed (Figure 5.8(b)) is able to provide a more reliable prediction of the locations of the two pedestrians, leading to better tracking performance.

6. CONCLUSION

Reliable pedestrian detection and tracking are crucial for numerous utility areas. This thesis presented a new tracking-by-detection approach for detecting and tracking pedestrians from point clouds acquired by a mobile Doppler LiDAR.

The performance of the proposed approach was discussed in terms of both pedestrian detection and tracking. Pedestrian detection consisted of cluster selection and pedestrian classification. The mean shift-based re-clustering process, which was introduced between the clustering process and classification, was shown to be capable of extracting single pedestrian candidates from multiple pedestrians' candidates effectively (Figure 5.1). To evaluate the effect of speed information in classification, we compared the performance of the classifiers with the speed information to the performance of those without the speed information. In general, the classifiers including the speed information outperformed those without the speed information, especially for detecting pedestrians that were far from the scanner. With the speed information, missing detections only occurred when pedestrians fulfilled both the following conditions: (1) pedestrians were occluded or far from the sensor, and (2) their average radial velocity was nearly zero. In addition, by utilizing the speed information, the average recall for the two datasets is 0.9620. Such a high recall was capable of improving the performance of pedestrian tracking. We also compared the performance of SVM-based classifiers and RF-based classifiers. Overall, the performance of the RF-based classifiers was better than the SVM-based classifiers no matter whether the speed information was included so the detection results from Two-Step RF were selected as input for pedestrian tracking. In the tracking step, the state vector of the pedestrians was estimated by not only position observations but speed observations as well, which increased the precision of the predicted movements of the pedestrians, leading to more robust and reliable tracking performance.

There are several areas where the proposed method could be improved. First, the detection method could only segment pedestrians that are close to other pedestrians or small objects such as road signs. When pedestrians are moving to large objects such as buildings or poles, they are often regarded as a part of them. Future research could investigate a method of segmenting a pedestrian from a larger cluster. Second, there are some pedestrians missed because one or more of their dimensions are out of the range of the pre-defined thresholds. Therefore, more adaptive criteria should be proposed to include these pedestrians. Third, the proposed approach detects pedestrians

frame by frame but does not consider the relationship between consecutive frames. The detection results from previous frames may provide useful information for pedestrian detection in the current frames. Lastly, even though the constant velocity model achieves high performance in pedestrian tracking, there are some more complex and advanced models, such as Constant Turn Rate and Velocity (Schubert et al. 2011), could be investigated to improve the tracking performance.

REFERENCES

Azim, A., & Aycard, O. (2012). Detection, classification and tracking of moving objects in a 3D environment. *2012 IEEE Intelligent Vehicles Symposium*, 802–807.

Brendel, W., Amer, M., & Todorovic, S. (2011). Multiobject tracking as maximum weight independent set. *CVPR 2011*, 1273–1280.

Cabanes, Q., & Senouci, B. (2017). Objects detection and recognition in smart vehicle applications: Point cloud based approach. *2017 Ninth International Conference on Ubiquitous and Future Networks (ICUFN)*, 287–289.

Cao, X. B., Qiao, H., & Keane, J. (2008). A low-cost pedestrian-detection system with a single optical camera. *IEEE Transactions on Intelligent Transportation Systems*, *9*(1), 58-67.

Camara, F., Bellotto, N., Cosar, S., Nathanael, D., Althoff, M., Wu, J., Ruenz, J., Dietrich, A., & Fox, C. (2020). Pedestrian models for autonomous driving part i: Low-level models, from sensing to tracking. *IEEE Transactions on Intelligent Transportation Systems*, 1–21.

Chen, C., Yang, B., Song, S., Tian, M., Li, J., Dai, W., & Fang, L. (2018). Calibrate Multiple Consumer RGB-D Cameras for Low-cost and Efficient 3D Indoor Mapping. *Remote Sensing*, *10*(2), 328.

Chen, J., Ye, P., & Sun, Z. (2019). Pedestrian Detection and Tracking Based on 2D Lidar. *2019 6th International Conference on Systems and Informatics (ICSAI)*, 421–426.

Chiu, H. K., Prioletti, A., Li, J., & Bohg, J. (2020). Probabilistic 3D Multi-Object Tracking for Autonomous Driving. *arXiv preprint arXiv:2001.05673*.

Comaniciu, D., & Meer, P. (2002). Mean shift: A robust approach toward feature space analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *24*(5), 603–619.

Cortes, C., & Vapnik, V. (1995). Support-Vector Networks. *Machine Learning*, *20*(3), 273–297.

Cui, J., Zha, H., Zhao, H., & Shibasaki, R. (2007). Laser-based detection and tracking of multiple people in crowds. *Computer Vision and Image Understanding*, *106*(2–3), 300–312.

De Maesschalck, R., Jouan-Rimbaud, D., & Massart, D. L. (2000). The Mahalanobis distance. *Chemometrics and Intelligent Laboratory Systems*, *50*(1), 1–18.

Dehghan, A., Idrees, H., Zamir, A. R., & Shah, M. (2014). Automatic Detection and Tracking of Pedestrians in Videos with Various Crowd Densities. In U. Weidmann, U. Kirsch, & M. Schreckenberg (Eds.), *Pedestrian and Evacuation Dynamics 2012* (pp. 3–19). Springer International Publishing.

Dewan, A., Caselitz, T., Tipaldi, G. D., & Burgard, W. (2016). Motion-based detection and tracking in 3D LiDAR scans. *2016 IEEE International Conference on Robotics and Automation (ICRA)*, 4508–4513.

Ester, M., Kriegel, H. P., Sander, J., & Xu, X. (1996, August). A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. In *Kdd* (Vol. 96, No. 34, pp. 226-231).

Fukunaga, K., & Hostetler, L. (1975). The estimation of the gradient of a density function, with applications in pattern recognition. *IEEE Transactions on information theory*, *21*(1), 32-40.

Geiger, A., Lenz, P., Stiller, C., & Urtasun, R. (2013). Vision meets robotics: The KITTI dataset. *The International Journal of Robotics Research*, *32*(11), 1231–1237.

Haselich, M., Jobgen, B., Wojke, N., Hedrich, J., & Paulus, D. (2014). Confidence-based pedestrian tracking in unstructured environments using 3D laser distance measurements. *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 4118–4123.

Jafari, O. H., Mitzel, D., & Leibe, B. (2014). Real-time RGB-D based people detection and tracking for mobile robots and head-worn cameras. *2014 IEEE International Conference on Robotics and Automation (ICRA)*, 5636–5643.

Kadlec, E. A., Barber, Z. W., Rupavatharam, K., Angus, E., Galloway, R., Rogers, E. M., Thornton, J., & Crouch, S. (2019). Coherent Lidar for Autonomous Vehicle Applications. *2019 24th OptoElectronics and Communications Conference (OECC) and 2019 International Conference on Photonics in Switching and Computing (PSC)*, 1–3.

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

Kim, C., Jung, Y., & Lee, S. (2020). FMCW LiDAR System to Reduce Hardware Complexity and Post-Processing Techniques to Improve Distance Resolution. *Sensors*, *20*(22), 6676.

Lang, A. H., Vora, S., Caesar, H., Zhou, L., Yang, J., & Beijbom, O. (2019). PointPillars: Fast Encoders for Object Detection from Point Clouds. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 12689–12697.

Lee, J. H., Tsubouchi, T., Yamamoto, K., & Egawa, S. (2006). People Tracking Using a Robot in Motion with Laser Range Finder. *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2936–2942.

Li, K., Wang, X., Xu, Y., & Wang, J. (2016). Density enhancement-based long-range pedestrian detection using 3-d range data. *IEEE Transactions on Intelligent Transportation Systems*, *17*(5), 1368–1380.

Liu, J., Liu, Y., Zhang, G., Zhu, P., & Chen, Y. Q. (2015). Detecting and tracking people in real time with RGB-D camera. *Pattern Recognition Letters*, *53*, 16–23.

Liu, K., Wang, W., & Wang, J. (2019). Pedestrian Detection with Lidar Point Clouds Based on Single Template Matching. *Electronics*, *8*(7), 780.

Luo, W., Xing, J., Milan, A., Zhang, X., Liu, W., & Kim, T.-K. (2021). Multiple Object Tracking: A Literature Review. *Artificial Intelligence*, *293*, 103448.

Ma, Y., Anderson, J., Crouch, S., & Shan, J. (2019). Moving Object Detection and Tracking with Doppler LiDAR. *Remote Sensing*, *11*(10), 1154.

Navarro-Serment, L. E., Mertz, C., & Hebert, M. (2010). Pedestrian Detection and Tracking Using Three-dimensional LADAR Data. *The International Journal of Robotics Research*, *29*(12), 1516–1528.

Navarro, P., Fernández, C., Borraz, R., & Alonso, D. (2017). A Machine Learning Approach to Pedestrian Detection for Autonomous Vehicles Using High-Definition 3D Range Data. *Sensors*, *17*(12), 18.

Nordin, D., & Hyyppae, K. (2002). Advantages of a new modulation scheme in an optical self mixing frequency-modulated continuous-wave system. *Optical Engineering*, *41*(5), 1128-1133.

Piggott, A. Y. (2020). Understanding the physics of coherent LiDAR. *arXiv preprint arXiv:2011.05313*. 41.

Pomerleau, F., Krusi, P., Colas, F., Furgale, P., & Siegwart, R. (2014). Long-term 3D map maintenance in dynamic environments. *2014 IEEE International Conference on Robotics and Automation (ICRA)*, 3712–3719.

Premebida, C., Carreira, J., Batista, J., & Nunes, U. (2014). Pedestrian Detection Combining RGB and Dense LIDAR Data. *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 4112–4117.

Qi, C. R., Su, H., Mo, K., & Guibas, L. J. (2017). PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 77–85.

Retting, R. (2017). Pedestrian Traffic Fatalities by State. *Governors Highway Safety Association: Washington, DC, USA*.

Royo, S., & Ballesta-Garcia, M. (2019). An Overview of Lidar Imaging Systems for Autonomous Vehicles. *Applied Sciences*, *9*(19), 4093.

Savtchenko, C., & Spletzer, J. R. (2011). Sidewalk-level People Tracking with a Low-cost 3D LIDAR System. *Lehigh University Technical Report LU-CSE-11-003*.

Schubert, R., Adam, C., Obst, M., Mattern, N., Leonhardt, V., & Wanielik, G. (2011). Empirical evaluation of vehicular models for ego motion estimation. *2011 IEEE Intelligent Vehicles Symposium (IV)*, 534–539.

Schulter, S., Vernaza, P., Choi, W., & Chandraker, M. (2017). Deep network flow for multi-object tracking. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 6951–6960.

Shalev-Shwartz, S., & Ben-David, S. (2014). *Understanding machine learning: From theory to algorithms*.

Shi, S., Wang, X., & Li, H. (2019). PointRCNN: 3D Object Proposal Generation and Detection from Point Cloud. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 770–779.

Stewart, R., Andriluka, M., & Ng, A. Y. (2016). End-to-end people detection in crowded scenes. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2325–2333.

Wang, D. Z., & Posner, I. (2015, July). Voting for Voting in Online Point Cloud Object Detection. In *Robotics: Science and Systems* (Vol. 1, No. 3, pp. 10-15607).

Wang, D. Z., Posner, I., & Newman, P. (2015). Model-Free Detection and Tracking of Dynamic Objects with 2D Lidar. *The International Journal of Robotics Research*, *34*(7), 1039–1063.

Weng, X., & Kitani, K. (2019). A Baseline for 3D Multi-Object Tracking. *arXiv preprint arXiv:1907.03961*.

Wu, T., Hu, J., Ye, L., & Ding, K. (2021). A pedestrian detection algorithm based on score fusion for multi-lidar systems. Sensors, 21(4), 1159.

Yan, Z., Duckett, T., & Bellotto, N. (2020). Online learning for 3D LiDAR-based human detection: Experimental analysis of point cloud clustering and classification methods. *Autonomous Robots*, *44*(2), 147–164.

Yan, Z., Duckett, T., & Bellotto, N. (2017). Online learning for human classification in 3D LiDAR-based tracking. *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 864–871.

Yang, Z., Sun, Y., Liu, S., Shen, X., & Jia, J. (2018). Ipod: Intensive point-based object detector for point cloud. *arXiv preprint arXiv:1812.05276*.

Yilmaz, A., Javed, O., & Shah, M. (2006). Object Tracking: A Survey. *Acm computing surveys (CSUR)*, *38*(4), 13-es.

Zhang, L., Li, Y., & Nevatia, R. (2008). Global data association for multi-object tracking using network flows. *2008 IEEE Conference on Computer Vision and Pattern Recognition*, 1–8.

Zhang, J., Xiao, W., Coifman, B., & Mills, J. P. (2020). Vehicle Tracking and Speed Estimation From Roadside Lidar. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, *13*, 5597-5608.

Zhang, M., Fu, R., Cheng, W., Wang, L., & Ma, Y. (2019). An Approach to Segment and Track-Based Pedestrian Detection from Four-Layer Laser Scanner Data. *Sensors*, *19*(24), 5450.

Zhang, M., Fu, R., Guo, Y., & Wang, L. (2020). Moving Object Classification Using 3D Point Cloud in Urban Traffic Environment. Journal of Advanced Transportation, 2020.

Zhang, W., Qi, J., Wan, P., Wang, H., Xie, D., Wang, X., & Yan, G. (2016). An Easy-to-Use Airborne LiDAR Data Filtering Method Based on Cloth Simulation. *Remote Sensing*, *8*(6), 501.

Zhao, J., Xu, H., Liu, H., Wu, J., Zheng, Y., & Wu, D. (2019). Detection and tracking of pedestrians and vehicles using roadside LiDAR sensors. *Transportation research part C: emerging technologies*, *100*, 68-87.

Zhou, Y., & Tuzel, O. (2018). VoxelNet: End-to-End Learning for Point Cloud Based 3D Object Detection. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 4490–4499.

Zhu, B., Jiang, Z., Zhou, X., Li, Z., & Yu, G. (2019). Class-balanced Grouping and Sampling for Point Cloud 3D Object Detection. *arXiv preprint arXiv:1908.09492*.