MOVING OBJECT DETECTION AND TRACKING WITH DOPPLER LIDAR

by

Yuchi Ma

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 2019

THE PURDUE UNIVERSITY GRADUATE SCHOOL
STATEMENT OF COMMITTEE APPROVAL

Dr. Jie Shan, Chair

Lyles School of Civil Engineering

Dr. Melba M. Crawford

Lyles School of Civil Engineering

Dr. James S. Bethel

Lyles School of Civil Engineering

Approved by:

Dr. Dulcy M. Abraham

Head of the Graduate Program

Dedication

This thesis is dedicated to Jiashan Zhou, my dear uncle, who enlightened me and made me fall in love with reading. You will always live with us.

ACKNOWLEDGMENTS

Time passes quickly like a white pony's shadow across a crevice. I still remembered how perturbed and nervous I was when I stepped across the boarding gate and flew to the USA on August 14th, 2017. At that moment, I had no idea about my future. No fellowship, no friendship, no specific research interests. That was a sleepless flight.

Luckily, in the past two years, I have got generous help and support from so many people. First and foremost, I appreciate my advisor and the chair of my committee, Prof. Jie Shan, for his selfless support. I have no words, neither in Chinese nor in English, to describe how deeply Prof. Shan has impacted me on my mindset and personal development. Thank Prof. Shan for having faith in me and advising me to finish the thesis. I am not a smart student and so careless that tend to make the same mistakes again and again. Thank Prof. Shan for his patience and tolerance, and persistently help me correct and strengthen my weak points. He always instructs me, saying "You should magnify every tiny point when doing research and dig it as deep as you can. Try to be neat and succinct". Thank Prof. Shan for always having a strict requirement on me in research and writing. Definitely, I am far from reaching it. I will bare his words in mind and keep working on it.

Thank the Geospatial Research Lab and Blackmore Sensors and Analytics Inc. who provided the data for this research. Thank Dr. John Anderson and Stephen Crouch for helping me understand the equipment and data.

I give my sincere thanks to Prof. Melba Crawford and Prof. James Bethel, both of whom generously serve on my committee. The course Multi- and Hyper- Spectral Remote Sensing instructed by Prof. Crawford was one of the first classes I took at Purdue. In the course, I often raised my hand and asked some silly questions with my poor English, but I would get helpful feedbacks from Prof. Crawford every time. Prof. Crawford also gave a lot of constructive advice in the course project, especially helping me avoid potential and unintended plagiarism. Coincidently, the course Adjustment of Geospatial Observations instructed by Prof. Bethel was among the last courses I took at Purdue. I actually took a course about adjustment at Wuhan

University while I got a bad grade then. Thank for the clear illustration and inspiring assignments given by Prof. Bethel. I learned a lot and did better this time. I appreciate Prof. Melba Crawford and Prof. James Bethel for their help and advice.

I want to give my thanks to Prof. Wolfgang Förstner of Bonn University, Germany. I am so lucky that I could take his class when he visited Purdue. He is really a giant in the area of photogrammetry while he was always modest and ready to help students. We used to spend two or three hours discussing my research or topics from his class. Now and then, he would write code to help me understand at scene or show me how to search literatures efficiently and effectively.

How can I forget my friends? Thank Dr. Qinghua Li. I will always remember the barbeques and the field trip we had together. In cooperation with him, I published my first paper. He has set a good example for me. Thank Dr. Serkan Ural, Dr. Yong Xu and Dr. Bo Xu. I have got a lot of interesting ideas from discussion with them. Thank Yue Li and Ke Liu from who have given me a lot of advice on how to do research. Thank Zhixin Li from whom I have learned how to program in Python. Thank Zilong Yang from whom I have learned some state-of-art methods in data mining. Thank Xiangxi Tian and Ce Wang for bringing new blood to the geomatics research.

Last but not least, thank my dear parents. Thank you for letting me chase my dream. Thank you, Mom, for comforting me whenever I feel hopeless. Thank you, Dad, for assisting me retrieve courage by telling historic stories. Happy 50^{th} birthday to my knowledgeable father.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABSTRACT

Author: Ma, Yuchi. MSCE
Institution: Purdue University
Degree Received: May 2019
Title: Moving Object Detection and Tracking with Doppler LiDAR
Committee Chair: Jie Shan

Perceiving the dynamics of moving objects in complex scenarios is crucial for smart monitoring and safe navigation, thus a key enabler for intelligent supervision and autonomous driving. A variety of research has been developed to detect and track moving objects from data collected by optical sensors and/or laser scanners while most of them concentrate on certain type of objects or face the problem of lacking motion cues. In this thesis, we present a data-driven, model-free detection-based tracking approach for tracking moving objects in urban scenes from time sequential point clouds obtained via state-of-art Doppler LiDAR, which can not only collect spatial information (e.g. point clouds) but also Doppler images by using Doppler-shifted frequencies. In our approach, we first use Doppler images to detect moving points and determine the number of moving objects, which are then completely segmented via a region growing technique. The detected objects are then input to the tracking session which is based on Multiple Hypothesis Tracking (MHT) with two innovative extensions. One extension is that a new point cloud descriptor, *Oriented Ensemble of Shape Function (OESF)*, is proposed to evaluate the structure similarity when doing object-to-track association in MHT. Another extension is that speed information from Doppler images is used to predict the dynamic state of the moving objects, which is integrated into MHT to improve the estimation of dynamic state of moving objects. The proposed approach has been tested on datasets collected by a terrestrial Doppler LiDAR and a mobile Doppler LiDAR separately. The quantitative evaluation of detection and tracking results shows the unique advantages of the Doppler LiDAR and the effectiveness of the proposed detection and tracking approach.

1. INTRODUCTION

1.1 Background

Object detection deals with detecting instances of semantic objects from images or other data while object tracking is defined as the methods of following one or multiple objects over a sequence of time steps. Both have been studied extensively for their broad applications in visual surveillance, sports, traffic monitoring, robotics, and autonomous driving. For an autonomous vehicle to make decisions and traverse safely through the busy urban streets, for example, it needs to detect, track, and subsequently predict the motion states of moving objects nearby. The number of moving objects on roads (e.g. cars, pedestrians, bicyclists) varies dramatically. Their dynamic states and behavior may change as they move and interact with the environment. A moving sensor can also result in large changes to the appearance of static background due to viewpoint changes. In addition, occlusion can make objects totally disappear. All those factors make it very challenging to detect and track moving objects in urban scenes.

Current researches on object detection and tracking are mainly vision-based (images and/or videos) [1]–[6]. Multispectral information collected by optical sensors offers substantial and distinct features thus enables easy segmentation of objects in crowded scenes. In the recent past, several centralized benchmarks, such as KITTI Vision Benchmark [7], MOT16 [8], PETS 2016 [9], have been proposed from the computer vision community for performance evaluation of tasks including object detection, optical flow, pedestrian detection, and multiple target tracking. Those benchmarks and challenges have advanced the state-of-the-art in detection and tracking fields. However, with optical sensors only it is hard to derive precise 3D position information. Also, optical sensors are heavily subject to illumination conditions. Some efforts try to combine 2D/3D laser rangers and cameras by using data fusion approaches [10]. Such hybrid methods always face the precise calibration problem between the ranging device and the camera [11]. Some other researches have been carried out exclusively on the ranging/laser scanner data in the form of 3D point clouds [12]–[15]. The advantages of a laser scanner include: its high accuracy in position measurement (precision level can be $1cm$); direct measurement of 3D position; penetration of various barriers (e.g. trees, bushes) and an active sensing modality allowing it to work both during

day and night. Due to these features, researches on detection and tracking with LiDAR are gaining increased attention.

Depending on different application purposes, researches on object tracking include Single Object Tracking (SOT) and Multiple Object Tracking (MOT). Researches on SOT focus on improving appearance models and/or motion model to better address objects' variations in scale, rotation, and illumination. On the other hand, researches on MOT additionally face two challenges [16]: 1) how are any number of objects in a scene determined and 2) how is an identified object associated in continuous and discontinuous frames. As this study is interested in multiple moving objects in street scenes, we will focus on MOT.

1.2 Related Work

Based on whether detecting objects before initializing tracks, MOT methods are categorized into Detection-Based Tracking (DBT) [17] and Detection-Free Tracking (DFT) [18]. DBT methods require objects detection before tracking, while DFT methods require the number of moving objects to be assigned as input. The DBT method is more general as it doesn't require prior information about the number of objects.

To detect an object from point clouds, we can use either model-based or model-free approaches [19], [20]. Model-based approaches detect objects based on prior model information. Such approaches are preferred when the interest targets are specified and therefore can be modeled in advance. For example, some methods exploit leg signatures to determine position of pedestrians. In [21], each leg of pedestrians is extracted using the pattern of rhythmic swing legs based on successive laser frames; then, a tracker based on Kalman filter and Rao-Blackwellized Monte Carlo data association filter is used to track multiple people in crowds. [12] adopts multiple hypothesis tracking with adaptive occlusion probabilities to track 2D leg signatures in laser range data. [13] utilizes a supervised learning method with AdaBoost to train a classifier on 14 features of legs in 2D range data to detect people. The detected individual persons are tracked with multiple hypothesis tracking as well. [22] focuses on tracking vehicles, in which the vehicle's geometry is approximated with a rectangular shape of width and length. Besides, a vehicle dynamic model is proposed which assumes the velocity evolves via the addition of random bounded noise and the

pose changes by linear motion. Instead of using discriminative detectors, [11] develops a generative model with the capability to detect and track a wide range of object classes of varying sizes and shapes.

Model-free approaches don't require prior model information so that objects of arbitrary shapes and sizes can be detected. For example, [23] projects all 3D points onto 2D occupancy grid parallel to the ground with grid size as 0.1 m × 0.1 m and clusters all connected cells into one large cell; the clustered objects are then classified as human or non-human by a trained Support Vector Machine (SVM); finally, a Kalman filter is used to track the human objects. [14] proposes a Simultaneous Detection And Tracking (SDAT) method without specifically detecting the objects in each frame. This SDAT method is realized by formulating the point assignment task as an energy function. [24] constructs a system using a variant of Random Sample Consensus (RANSAC) that estimates the vehicle pose and detects moving objects via distinguishing non-stationary objects from stationary objects by the spatial consistency and differentiating moving objects from outlying objects by temporal consistency. [25] detects static points by traversing a voxel grid and finding differences in volumetric occupancy with "point shadows". In addition, some methods introduce point cloud descriptors to assist detection and object-to-track association. For example, in [20], uniformly sampled key points in consecutive scans are matched by their SHOT (Signature of Histograms of OrienTations) [26] based on what the motion models for the sensor and the moving objects are estimated. [27] organizes local convexity criterion with the normal vector and flatness value for each point, which are used to detect moving objects and manage tracks. [28] obtains high-order feature representations for 3D local patches by deep learning techniques; then automatically detects 3D vehicles using Deep Boltzmann Machine Hough Forests.

Most of current approaches, model-based or model-free, exclusively concentrate on certain objects but face limitations when applied on other objects. For example, methods proposed by [29] only detect pole-like objects in urban scenes; the motion-based method proposed by [20] can segment and track vehicles of various types but it does not work well for pedestrians because slowly-moving pedestrians do not give enough motion cues; [14] focuses on detecting and tracking pedestrians in LiDAR scans. However, in practice, all types of moving objects can be of interest and present in the same scene. Moreover, in specific application (e.g. autonomous driving),

detecting moving objects and measuring speed by calculating multiple pulse bounces with commonly used pulsed LiDAR scans is slow and error-prone.

Therefore, in this research, we intend to address those barriers and develop a model-free detection-based tracking approach for detecting and tracking moving objects in point clouds obtained by a state-of-art Doppler LiDAR [30]. The Doppler LiDAR can not only collect 3D spatial information but also the relative speed between the moving objects and the sensor on the beam direction. The developed approach is shown in the following workflow in Fig 1.1.

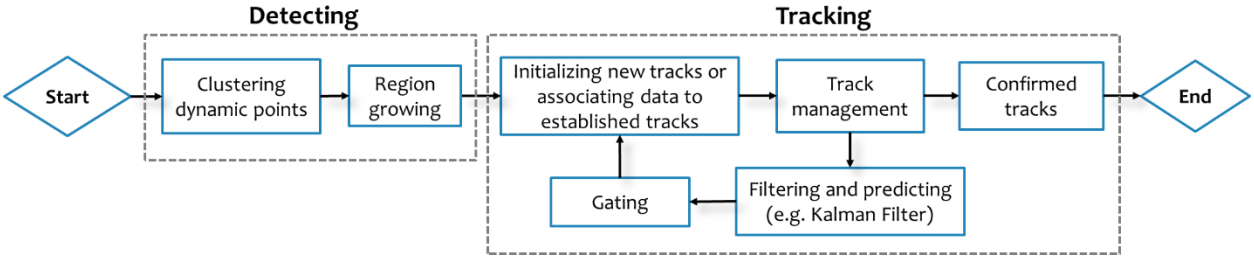


Fig 1.1 Workflow of the proposed DBT approach.

In detection, the first step is to cluster moving points to determine the number of moving objects and their approximate positions. Clustering is the process of spatially grouping large datasets according to data similarity [31]. Several types of clustering methods are available [32]–[34]. Basically, there are four categories of clustering algorithms [31], [35]–[37]: *Partitioning*, *Hierarchical*, *Grid-based*, and *Density-based* clustering algorithms. *Partitioning* algorithms require to specify the number of clusters k as input. It normally starts with an initial partition of dataset D and iteratively relocates data among k clusters until an object function is optimized. The typical partitioning algorithms include the widely-used *k-means* algorithm [38] and its variations (i.e. *fuzzy k-means* [33], *EM for Gaussian Mixtures* [32]). The *hierarchical* algorithm creates a hierarchical decomposition of dataset D, called dendrogram [35]. There are generally two strategies for hierarchical algorithms [39]: agglomerative and divisive. The agglomerative approach (or bottom-up approach, such as CURE [40], BIRCH [41]) initially regard every single point as a unique cluster and merge "similar" clusters. The divisive approach (top-down approach) starts with one cluster and split them recursively. The *hierarchical* algorithm doesn't require the number of clusters k as input but it cannot correct erroneous merges or splits. *Grid-based* algorithms, such as STING [42], divide the object space into finite uniform cells that form a multi-

level grid structure based on which all operations for clustering are performed. Due to the grid structure, the efficiency of *Grid-based* algorithms is high.

Previously mentioned clustering methods are good at finding spherical-shaped clusters but fail to mine clusters of arbitrary shapes. To address the inadequacy, *Density-based* methods are proposed. The idea behind *Density-based* methods (i.e. *DBSCAN* [35], *OPTICS* [43], *DENCLUE* [44]) is that data, which form a dense region in terms of distance or other features, should be grouped together into one cluster. As clustering or not is dependent on the distribution of data, *Density-based* methods don't require the predetermination of the number of clusters. Moreover, clusters with arbitrary shapes can be found with *Density-based* methods. Therefore, we adopt *ST-DBSCAN* [31], a variant to *DBSCAN* [35] that considers both spatial and temporal similarity.

After detection, detected objects are input to the tracking algorithm. However, they may not all arise from the real targets but from clutter or false alarm [45]. In every frame, each detected object has basically three possible affiliations: 1) It belongs to an established track, 2) it is the starting position of a new track, or 3) it is a false alarm. Consequently, problems about how to do object-to-track association and manage tracks arise. One commonly used method to address object-to-track association and manage tracks is the Global Nearest Neighbor (GNN) method. GNN is easy to implement as it typically searches for the optimal assignment by minimizing a cost function (commonly, the total summed distance between new objects and established tracks) [46]. It attempts to find and propagate the "single most likely data association hypothesis" at each frame [44], [46]. GNN method assigns detections to tracks frame by frame. Another widely used method is the Joint Probabilistic Data Association (JPDA) algorithm [47]. With the assumption that only one target is present, and all other measurements are Poisson-distributed clutter, JPDA first computes the posterior probability of each candidate measurement as the weights [47]. After that, tracks are updated with a weighted sum of each hypothesis track estimate from every candidate measurement [48]. Similar to GNN, JPDA updates each track estimation with one set of assignment at each time step. If there are wrong assignments or missing detections, GNN and JPDA cannot recover such errors because both methods require finishing the unique object-to-track assignment at each time step and never look back.

To better handle potential wrong object-to-track association, false alarm and missing detections, Multiple Hypotheses Tracking (MHT) was put forth in 1979 [49]. When receiving new observations, MHT considers all its three possible affiliations by retaining all potential measurement-to-track assignments in the form of target trees. For example, in each frame, when an object is detected, it is used as the root node to initialize a new target tree. Meanwhile, the object is added to established target trees as a branch according to all potential object-to-track assignments decided by Gating (introduced in section 4.2). Moreover, because of the possibility that the object in an established track is missing in current frame, established target trees grow an empty branch with a null observation. While receiving new observations in each frame, target trees keep spawning and growing as a set of hypotheses is propagated with the expectation that future measurements will resolve assignment ambiguities [48]. We thus adopt MHT to manage tracks in this research. To increase the accuracy and robustness of object-to-track association in MHT, *Oriented Ensemble of Shape Functions* (*OESF*), a global Point Cloud Descriptor (PCD), is proposed to measure the structure similarity between detected point cloud objects when doing object-to-track association. We also integrate speed information into MHT to improve the statue prediction, thus increasing robustness in tracking. We have described the proposed model-free detection-based tracking method as *'MHT-PCD-Speed'*.

1.3 Overview of the Thesis

The remainder thesis is organized as follows. In section 2, the mechanism of Doppler LiDAR and datasets used in this thesis are introduced. In section 3 and 4, the detection algorithm and tracking algorithm in the proposed framework are introduced respectively. In section 5, experiments and corresponding results are shown, and the analysis is given. Finally, we offer conclusions and discussion of the thesis in section 6.

2. DOPPLER LiDAR AND DATA

2.1 Principles

Commonly used ranging systems includes RADAR (RAdio Detection And Ranging), LiDAR (Light Detection And Ranging), and LADAR (LAser Detection And Ranging). Although named differently, all of them work by transmitting and receiving electromagnetic energy while in different frequency bands [50]. A Light Detection and Ranging (LiDAR) sensor is an active surveying system that measures distance to the target by illuminating the target with pulsed laser beam and recording the reflected pulses. One of the most commonly used LiDAR is called pulsed or linear mode LiDAR, which emits very short but intense pulse of laser radiation and records its Time of Fly (TOF) from the moment when the pulse is emitted to the moment when the reflected signal is received by the sensor [51]. With the speed of the pulsed laser beam, which is known, the range can be calculated. Typical linear mode LiDAR emits Gaussian–shaped laser pulses. The waveforms of reflected pulses are used to distinguish different objects. If the width of laser pulse is too wide, multiple waveforms would be mixed as one. Therefore, the range resolution of linear mode LiDAR is limited by the width of the laser pulse.

Although linear mode LiDAR is currently the most mature light measurement approach [1], Continuous Wave (CW) LiDAR are gaining more and more attention. CW LiDAR doesn't emit a pulse but a continuous beam of laser radiation. By measuring the phase difference between transmitted and the received beam, the range can be computed. As the wavelength of a period of sinusoidal wave can be very short, CW LiDAR can achieve more precise measuring results. Based on continuous wave technology, Frequency-Modulated Continuous Wave (FMCW) LiDAR are invented. The FMCW technology is best described as "emulating a modern optical radar in its functionality", so sometimes it is also abbreviated as 'LADAR' [30]. When scanning, the system continuously illuminates a target. Once received the reflected continuous wave, the system demodulates and separates the reflected signal to the measurement signal and carrier signals. Compared to the original reference signal, the difference in phase angle between the reference signal and the reflected signal are derived, thus determining the beat note frequency [30]. To increase range resolution, the system transmits signal of a known stable frequency continuous

wave. The transmitted signal varies up and down in rapid succession after modulation. Hence the range resolution in FMCW LiDAR is determined by the highest frequency, thus the shortest wavelength [30]. The range resolution of the FMCW LiDAR is limited by the bandwidth of the modulated signal. In summary, compared to pulsed LiDAR, the FMCW LiDAR has several key benefits [30], [52]:

- Collect Doppler images with Doppler shift;
- Measure greater range with lower power;
- Higher range resolution.

The FMCW LiDAR, although outperforms the linear mode LiDAR in several aspects, still cannot replace the linear mode LiDAR because of efficiency. For example, the FMCW LiDAR requires much longer time to acquire scans [30]. Also, the processing unit for FMCW LiDAR is larger, increasing the size, weight, and power requirements of the system [30]. We need to decide which mode to use according to specific application purposes. For example, for autonomous driving, detecting moving objects and measuring speed by using the frequency changes with CW due to Doppler effect is faster and less error-prone than calculating multiple pulse bounces with pulsed LiDAR scans.

When scan a moving target, the range and Doppler effect both cause frequency shift, resulting in an ambiguity in the measurement between range and speed. The ambiguity is solved by the use of sequential chirps moving in upward and downward direction, where the Surface Acoustic Wave (SAW) device only compresses one of them [53], [54].The details are given below.

In FMCW LiDAR, there is a constant μ related to the applied voltage, laser frequency, and free spectral range. Given the speed of light C, for a target at a distance R from the scanner, the range time is:

$$t_r = 2R/C \tag{1}$$

If the target is moving with speed V_r relative to the scanner on the radial direction, for the upward chirp, there would be an additional delay caused by the Doppler shift:

$$t_d = \frac{f_0 - f_d}{\mu} \tag{2}$$

where f_0 is the center frequency. The measured TOF becomes:

$$t_{TOF\_up} = t_r + t_d \tag{3}$$

For the downward chirp, the μ has a opposite sign, thus the time delay is also opposite in sign. The measured TOF becomes:

$$t_{TOF\_down} = t_r + t_{d\_down} = t_r - t_d \tag{4}$$

With (3) and (4), the time delay $\mathrm{t_d}$ and the corrected TOF $\mathrm{t_r}$ should be:

$$t_r = \frac{1}{2}(t_{TOF_{up}} + t_{TOF_{down}}) \tag{5}$$

$$t_d = \frac{1}{2}(t_{TOF_{up}} - t_{TOF_{down}}) \tag{6}$$

With t_r the correct range to the target is recovered. With t_d the relative radial speed Δv of the target can be estimated:

$$\Delta \mathrm{v} = \frac{f_d C}{f_0} \tag{7}$$

When the target is moving away from the scanner, f_d is positive; when the target is approaching the scanner, f_d is negative. After that, with the dynamics of the scanner known, the radial speed of the moving objects can be recovered. Therefore, the sensor is named as Doppler LiDAR.

2.2 Test Areas and Data

This study uses two datasets collected by two types of Doppler LiDAR, a static terrestrial Doppler LiDAR and a mobile Doppler LiDAR which both provide point by point velocity measurements.

The static terrestrial Doppler LiDAR [30] is designed for fine scanning of certain region or targets. It is realized by a partially reflective mirror that targets the laser beam at a certain direction with a scanning angle of 4°. The scanning frequency is 5Hz. The maximum scanning range is about 200 meters. The point cloud collected by the static terrestrial Doppler LiDAR is under a local coordinate system with the sensor at the origin (0, 0, 0). We collected data with the static terrestrial Doppler LiDAR in October 2017. The testing field is located at Blackmore Sensors and Analytics

Inc. Bozeman, MT (top left of Fig 2.1). Within the region about 150 meters from the scanner, there is mainly grassland and an open parking area. In the region beyond 150 meters, there is a street (top middle of Fig 2.1).

Fig 2.1 From left to right: The static terrestrial (top) and the mobile (bottom) Doppler LiDAR system, the test site (blue triangle: scanner; red arrow: beam direction), and one frame point cloud color coded by speed (unit: m/s).

The mobile Doppler LiDAR is designed for autonomous driving or mapping. In the system, four co-registered Doppler LiDAR scanners are mounted on the top of a vehicle (bottom left of Fig 2.1). The horizontal scanning angle range for each scanner is 40°. As the scanning scope of neighboring scanners yields an overlapping area, the total horizontal scanning angle range for the system is 120°. The maximum scanning range is about 400 meters. Using the GPS/IMU built in the mobile Doppler LiDAR system, the scanned points are simultaneously transformed to the global coordinate system considering the motion of the platform. Similarly, radial speed is also adjusted to eliminate the effects of the moving platform. The test on the Mobile Doppler LiDAR scans was collected at downtown area in San Francisco in September 2018 (bottom middle of Fig 2.1).

Both types of Doppler LiDAR can do Doppler imaging, indicating the relative speed between scanned points and the scanning system (right of Fig 2.1). Positive speed suggests the target moving towards the sensor along the sensor's line of sight (radial direction), while a negative speed means the target is moving away from the sensor. For each point, we have its detected moment t

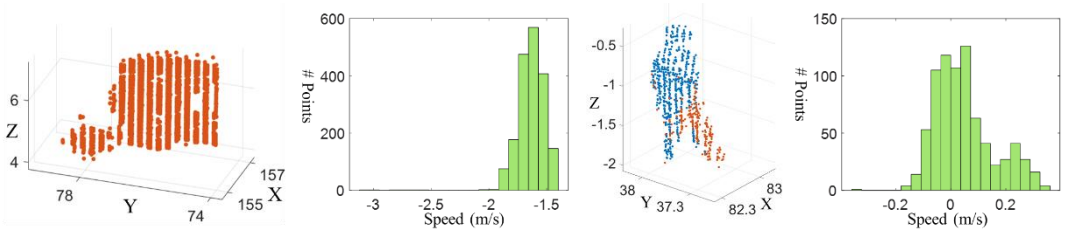
('microseconds since epoch'), its relative position (x, y, z), absolute position (x_{abs}, y_{abs}, z_{abs}), intensity I, relative radial speed S, absolute radial speed S_{abs}, the polar angle Φ measured from zenith (z-axis), the azimuthal angle θ measured from forward axis, and the range to the center of the scanner r.

3. MOVING OBJECT DETECTION

As we are interested in moving objects, the strategy is to take advantage of the speed information from Doppler images. This section first uses the Doppler image to detect and cluster moving points in the scene to determine the number of moving objects and their approximate positions in the scene. Then, a region growing technique is applied to segment the complete moving objects.

3.1 Moving Points Detection and Clustering

Doppler images can be used to easily differentiate non-stationary objects from the stationary background. However, it is difficult to directly and completely segment moving objects because not all points on the objects have the same motion state. On a rigid body (e.g. a car), every part can keep the same motion state. But for a non-rigid body (e.g. a pedestrian), different parts of the body can move in various ways (see Fig 3.1). For example, a pedestrian's two feet rhythmically interchange their duties by landing and moving. Moreover, due to noise in speed measurement, the measured speed of static background points is not exactly zero but a very small number (e.g. $10^{-4}\ m/s$). Therefore, a speed threshold ϵ_v is set to distinguish static points and dynamic points. According to the specification of the sensor, the precision of the radial speed is $\pm 0.1\ m/s$. We use it as the speed threshold, namely, points whose absolute radial speed larger than $0.1\ m/s$ are detected moving points while the rest points are static.



(a) (b)

Fig 3.1 Moving objects and their speed histograms. The figure shows the point cloud of a segmented truck (a) and a segmented pedestrian (b) (in meters) and the corresponding speed histograms (x-axis: speed at $0.1 m/s$ bin); y-axis: number of points). In this illustration, moving points are colored in *red* while static points *blue*.

After thresholding, those detected moving points are clustered to form dynamic clusters. The process of clustering is usually carried out according to the spatial-temporal similarity of data points. To process the Doppler LiDAR scans, the selected clustering method should meet three requirements:

- It should not assume the number of clusters be known as a prior since there is usually no prior information about the number of objects in each scene.
- It should be capable of discovering clusters of arbitrary shapes because the shape of clusters of generated dynamic points can be very varying.
- It should have good efficiency on a large dataset.

A bottom-up density-based clustering algorithm, *Density-Based Spatial Clustering of Applications with Noise* (*DBSCAN*), satisfies all three requirements listed above. It uses two concepts, density reachability and density connectivity, to determine whether two points are in the same cluster. *DBSCAN* requires two parameters ϵ_R and $MinPts$, where ϵ_R is the spatial distance threshold between density reachable points, and $MinPts$ is the minimum number of points in the neighborhood. With the idea that data in the same cluster should be close not only in spatial domain but also in temporal domain, we adopt *Spatial-Temporal-DBSCAN (ST-DBSCAN)* that additionally considers temporal similarity with a temporal threshold ϵ_T [31].

Moreover, in Doppler LiDAR scans, the point density varies along with range to the sensor, i.e. closer objects are more densely scanned than distant ones. There is no global optimal ϵ_R to best deal with both close and far clusters. To make the spatial metrics work on clusters in different distance range, we make the distance threshold adaptive to the range r. In summary, our *ST-DBSCAN* works in the following steps on a 3D point cloud dataset $D = \{p_1, p_2, \dots, p_n\}$ frame by frame:

1) Randomly choose an unvisited point $p_i = (x_i, y_i, z_i, t_i, r_i)$;
2) Calculate the adaptive distance threshold ϵ_{Ri} with the range r_i and the azimuth sampling resolution r_θ (see Formula 8);
3) Find all the neighboring points within spatial distance ϵ_{r_i} and within temporal distance ϵ_T to p_i. If the number of neighboring points is larger than $MinPts$, go to step 4; else, label p_i as visited noisy point and go back to step 1.

4) Cluster all points that are density reachable or density connecting [35] to point p_i and label them as visited. Go back to step 2.
5) Terminate the process until all the points are visited. The output is a set of clusters of dynamic points $C = \{c_1, \dots, c_m\}$.

The values of $MinPts$, ϵ_T, and $\epsilon(r_i)$ determine the performance of the clustering operation. $MinPts$ is set to smooth the density estimate, which is empirically set as 40 in this study. ϵ_T is set as 1% of one scanning period, equal to 0.002s The distance threshold $\epsilon(r_i)$ decides the range of density reachability [35]. The distance between two neighboring points (point density) is associated with the azimuth sampling resolution r_θ and the distance from the point to the sensor r_i. To distinguish different objects, the distance between two objects should be at least twice lager than the arc length between neighboring beams. Considering the potential non-reflection beams, we set a slightly loose threshold:

$$\epsilon_{r_i} = 3 r_i \mathrm{r}_\theta \tag{8}$$

3.2 Clustering Static Points – Completing the Object

The result of dynamic points clustering is a set of dynamic point clusters $C = \{c_1, \dots, c_m\}$. However, the detection is not finished as not all points on a non-rigid moving object are detected as dynamic points. As such, further steps need to be taken to complete the entire moving objects. The idea is to use afore detected dynamic clusters C as seeds for region growing. To make the region growing algorithm robust, a spatial threshold ϵ_{Di} and a temporal threshold ϵ_T is set as the growing criteria. Considering the point density on different objects are varying, ϵ_{Di} is uniquely set as the mean distance within the dynamic cluster c_i. The mean distance within c_i is estimated using the mean value of the mean distance of each point to its *KNN* points in the cluster.

In addition, another potential mistake in region growing algorithm would make is overgrowing, i.e., having too many points that actually do not belong to the object. To avoid overgrowing the cluster into the ground point, we filter ground points beforehand using Simple Morphological Filter [55]. In summary, the region growing algorithm works on every dynamic cluster c_i in dynamic clusters C with all other static points P_s as follows. For each frame:

1) Choose an unvisited dynamic cluster c_i from C;
2) Calculate the mean distance ϵ_{Di} among K nearest neighboring points in c_i;
3) Find all static points P_s' in P_s within ϵ_{Di} as well as ϵ_T from at least one point in c_i;
4) If there is no point left in P_s', go back to step 1; otherwise, include points in P_s' to cluster c_i and go back to step 2.
5) Terminate the region growing process when all members in C are visited.

The value of ϵ_T is set the same as in *ST-DBSCAN*; The value of K is set the same as the value of $MinPts$; After several iterations, the entire moving object can be completely segmented (see Fig 3.2). The next step is to track trajectories of detected objects between frames.

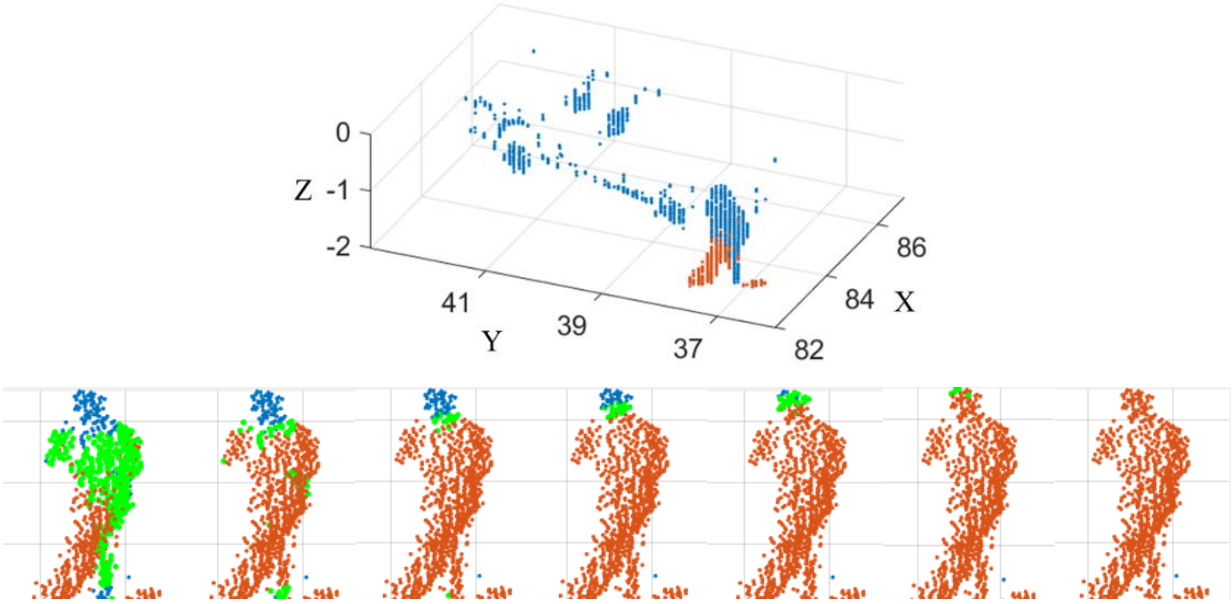


Fig 3.2 Clustered moving points and the static background in one frame (top). The growing process over seven iterations (bottom). Points on moving objects are in red while static points in blue. Points in green are growing points in each iteration.

4. MOVING OBJECT TRACKING

As introduced before, MHT outperforms other methods in terms of forming object-to-track association and handling false alarm and missing detections. Therefore, MHT framework is used to track the detected objects from the Doppler LiDAR scans. The objective of this stage is to establish the relation between objects detected in consecutive frames and determine the moving trajectories in the dataset. It consists of finding potential object-to-track associations (*Gating*), proposing track hypotheses and ranking each hypothesis according to their likelihood (*Scoring*), and determining the best set of hypotheses (*Global Hypothesis Formation*) [6], [46]–[48]. In addition, the filter used in our MHT is the Kalman filter. In the following sections, we present the implementation of the Kalman filter with Doppler speed information; then describe how the tracking algorithm works.

4.1 Kalman Filter with Doppler Images

Kalman filtering is a process of using a motion model (e.g. physical laws of motion) to filter noisy sensor observations thus giving a more accurate state estimate of a dynamic system [56]. The state variables S contain position component X and speed components V. The state vector in frame $k-1$ is:

$$S^{k-1} = [X^{k-1}, V^{k-1}]^{\mathrm{T}} \tag{9}$$

The stochastic model of the state vector is assumed to be a multivariate Gaussian model with covariance Σ_{SS}^{k-1}. The state vector $\tilde{S}^{\mathrm{k}}$ and state covariance $\tilde{\Sigma}_{SS}^{k}$ in frame k can be predicted:

$$\tilde{S}^{\mathrm{k}} = FS^{k-1} + w \tag{10}$$

$$\tilde{\Sigma}_{SS}^{k} = F\Sigma_{SS}^{k-1}(F)^{T} + Q \tag{11}$$

F is the process model, or state transition function:

$$F = \begin{bmatrix} 1 & 0 & 0 & \Delta t & 0 & 0 \\ 0 & 1 & 0 & 0 & \Delta t & 0 \\ 0 & 0 & 1 & 0 & 0 & \Delta t \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{12}$$

The process error w is approximated as a zero mean multivariate normal distribution with covariance $\mathrm{Q} = \mathrm{diag}([0, 0, 0, \sigma_{vx}^2, \sigma_{vy}^2, \sigma_{vz}^2]$. The variances of the less rapidly changing terms (position state) are set as 0 while the variances of the most rapidly changing terms (speed state) are set as 10.

After prediction, observations z^k in frame k are used to update $\tilde{S}^{\mathrm{k}}$ and $\tilde{\Sigma}_{SS}^k$. A measurement function H is used to project $\tilde{S}^{\mathrm{k}}$ from state space to measurement space and convert $\tilde{\Sigma}_{SS}^k$ to system uncertainty U^k with the measurement noise $n^k \sim N(0, R^k)$. The position terms in R^k are decided by the variance in x, y, z observations of all points on the object; the speed terms in R^k are decided by the variance of speed observation of all moving points on the object, and set the same for speed in all directions. The residual y between the real measurement z^k and estimated measurement $\tilde{z}^k$ is calculated:

$$\tilde{z}^k = H\tilde{S}^{\mathrm{k}} + n^k \tag{13}$$

$$y^k = z^k - \tilde{z}^k \tag{14}$$

$$U^k = H\tilde{\Sigma}_{SS}^k H^T + R^k \tag{15}$$

Finally, Kalman gain K is calculated and the state and state covariance are updated to frame k:

$$K^k = \tilde{\Sigma}_{SS}^k H^T (U^k)^{-1} \tag{16}$$

$$S^k = \tilde{S}^{\mathrm{k}} + K^k y^k \tag{17}$$

$$\Sigma_{SS}^k = (I - K^k H)\tilde{\Sigma}_{SS}^k \tag{18}$$

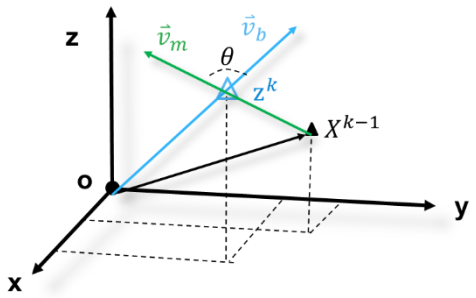


Fig 4.1 Estimate the speed in x, y, z direction with the observed speed in beam direction.

To decide values of z^k, the position components x^k is estimated using the mean value of all points p_j on the detected object. The speed component v_m^k in z^k is estimated by the measured radial (in beam direction) speed $|v_b|$. Given an observation z^k in current frame k together with the position of the scanner O, we can get the beam vector v_b^0. If the moving direction of the object v^0 is known, the angle between the beam vector and the moving direction can be estimated, thus the moving speed can be estimated by (see Fig 4.1):

$$v_m^k = \frac{|v_b|}{v_b^0 \cdot v^0}\, v^0 \tag{19}$$

When a track is initialized with the speed state vector as a zero vector, we assume the vector from X^{k-1}, the updated position statue of the track in the last frame k-1, to z^k as the moving direction v^0. In following frames, the moving direction is estimated using the updated speed state from Kalman filter. In our application, as there is no large change in elevation (z direction), we only consider tracking objects in x and y direction. Afterwards, based on the predicted and updated motion state, track hypotheses are proposed by performing gating as discussed below.

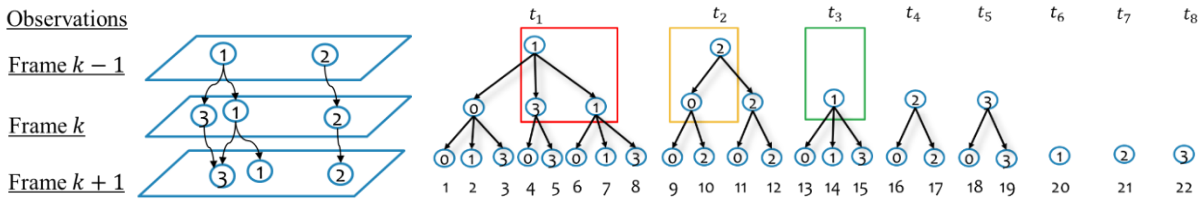
4.2 Gating and Proposing Track Hypotheses

Gating is the process of finding potential object-to-track associations [6], [46]. Intuitively, detections in consecutive frames that are close to each other are more likely to be the same object. To quantify the likelihood, a gating area for each track hypothesis is calculated based on the motion estimation in the form of (Mahalanobis) distance, which measures how many standard deviations away the jth new position observation x_j^k is from the mean of the predicted position distribution $\mathrm{N}(\tilde{X}_{l_i}^k, \tilde{\Sigma}_{XX}^k)$ of track l_i in frame k:

$$d_{\mathrm{m}}^2 = \left(\tilde{X}_{l_i}^k - x_{\mathrm{j}}^k\right)^T \left(\tilde{\Sigma}_{XX}^k\right)^{-1} \left(\tilde{X}_{l_i}^k - x_{\mathrm{j}}^k\right) \tag{20}$$

A gating threshold λ_{d_m} is set to decide candidates for object-to-track associations. To illustrate the gating process, we offer an example in Fig 4.2 (a), in which observations are given in three consecutive frames, namely, two observations in frame $k-1$ and three observations in frame k and $k+1$. Gating shows that the candidates of z_1^{k-1}, the 1st observation in frame $\mathrm{k}-1$, are z_1^k and z_3^k, while the candidate of z_2^{k-1} is z_2^k. Similar associations indicate by the arrows from frame k to $\mathrm{k}+1$ in Fig 4.2 (a).

After gating, multiple tracking hypotheses can be proposed in the form of target trees. Existing target trees are extended by including the candidate observations, with each candidate observation "spawn" a separate new branch (i.e. branches in the red rectangle in Fig 4.2 (b)). On the other hand, under the assumption that each new observation in any current frame may start a new track, each new observation is also used to start a new track as the root node (i.e. the root node is shown as the green rectangle in Fig 4.2 (b)). In addition, each existing tree spawns a new branch with a dummy observation to account for missing detections [6] (i.e. shown as the branch in the yellow rectangle in Fig 4.2 (b)).



(a) (b)

Fig 4.2 Gating and the association principle. (a): Observations in three consecutive frames. There are two existing tracks l_1 and l_2 in frame $k-1$ and 3 objects detected respectively in frame k and $k+1$. (b): Initialize and update target trees. Red rectangle: branches spawned by appending candidate observations to existing tracks; Yellow rectangle: branches spawned by appending dummy observations to account for missing detection; Green rectangle: initialize new trees to account for new tracks.

4.3 Scoring

Up to now, a set of track hypotheses are proposed while most of them are incompatible. To evaluate alternative track hypotheses, we need a probabilistic *Score Function* to rank each of them. Likelihood Ratio (*LR*) of a hypothesis H_i saying a set of observations D belongs to track l is defined as [46]:

$$LR_i = \frac{p(D|H_i)p_0(H_i)}{p(D|H_0)p_0(H_0)} = L_0 \frac{p\left(z_j^{1:K}|H_i\right)}{p\left(z_j^{1:K}|H_0\right)} = L_0 \frac{\prod_{k=1}^{K} p(z_j^k|z_j^{1:k-1}, H_i)}{\prod_{k=1}^{K} p(z_j^k|H_0)} \tag{21}$$

$p_0(H_i)$ and $p_0(H_0)$ are the prior probability of hypothesis H_i and false alarm hypothesis H_0, both of which are set as constant thus their ratio is denoted as a constant L_0. The larger $p(D|H_i)$ is and the smaller $p(D|H_0)$ is, LR is larger and hypothesis H_i is more likely to be true. We separate *LR*

into a product of Motion (MOT) term and a Point Cloud Descriptor (PCD) term. Under the assumption that measurement error from frame-to-frame is independent, *LR* is partitioned into:

$$LR_i = L_0 \prod_{k=1}^{K} LR_{MOT_i}(k) LR_{PCD_i}(k) \tag{22}$$

The MOT term LR_{MOT} accounts for the spatial closeness of an object-to-track association. The likelihood that z_j^k belongs to track l given the previous observations $z_j^{1:k-1}$ is assumed to be a Gaussian distribution. The conditional probability that z_j^k is a false alarm is set as a constant V_0^{-1}. Using d_{m}^2 from formula (20) and denoting *M* as the data dimension, LR_{MOT} in the kth frame is:

$$LR_{MOT_i}(k) = \frac{p\left(z_j^k | z_j^{1:k-1}, H_i\right)}{p\left(z_j^k | H_0\right)} = \exp\left(-\frac{d_m^2}{2}\right) V_0 \left((2\pi)^{\frac{M}{2}} \sqrt{\left|\tilde{\Sigma}_j^k\right|}\right)^{-1} \tag{23}$$

The PCD term LR_{PCD} evaluates the structural similarity of an object-to-track association. We propose a global point cloud descriptor, *Oriented Ensemble of Shape Function (OESF)*, to derive the feature vector of each observation. Details about *OESF* is given in the next section. The *OESF* feature vector of the j_{th} observation in frame k is notated as β_j^k. Assuming that associating observations based on PCD is a first-order Markov process, the PCD term at frame k is:

$$LR_{PCD_i}(k) = \frac{p(\beta_j^k | \beta_j^{1:k-1}, H_i)}{p(\beta_j^k | H_0)} = \frac{p(\beta_j^k | \beta_j^{k-1}, H_i)}{p(\beta_j^k | H_0)} = \frac{r\left(\beta_j^k,\ \beta_j^{k-1}\right)}{\gamma_0} \tag{24}$$

The likelihood of β_j^k belonging to track l given the previous PCD feature vector β_j^{k-1} is defined as $r(\beta_j^k,\ \beta_j^{k-1})$, the correlation coefficient between two *OESF* feature vectors β_j^{k-1} and β_j^k. $p(\beta_j^k | H_0)$ is set as a constant γ_0. We apply *log* on the LR to get the Log Likelihood Ratio (LLR) as the score function:

$$S_l(K) = LLR_K = ln(LR_K) = \sum_{k=1}^{K} [LLR_{MOT}(k) + LLR_{PCD}(k)] + ln(L_0) \tag{25}$$

Then, we can update the track score recursively with different weights w_{MOT} and w_{PCD} on the motion term and PCD term correspondingly [6], [46]:

$$S_i(k) = S_i(k-1) + \Delta S_i(k) \tag{26}$$

$$\Delta S_i(k) = \begin{cases} 0 & k = 0 \\ w_{MOT} LLR_{MOT_i}(k) + w_{PCD} LLR_{PCD_i}(k) & k \neq 0 \end{cases} \tag{27}$$

The w_{MOT} is set as 1. As we need to rely more on PCD term if we have more candidates in the gating area, the w_{PCD} is set as μn, determined with the number of candidates n and the PCD weight ratio μ. Then the weights for the PCD term and the MOT term are normalized:

$$w_j = \frac{w_j}{w_{MOT} + w_{PCD}}, j = MOT\ or\ PCD \tag{28}$$

According to scores of each track hypothesis, we can find a set of tracks with the highest overall score. Before that, we introduce the proposed *OESF* in next section.

4.4 Feature Description of Point Clouds of Objects

The proposed *OESF* is inspired by Ensemble of Shape Function (ESF) [57]. ESF describes 3D shape with multiple shape functions, namely, the distances between point pairs (D2), the square root of the area of the triangle formed by three randomly chosen points (D3), and the angle formed by three randomly chosen points (A3) [58]. ESF uses voxel grids as an approximation of the surface of the point cloud, based on which D2, D3, and A3 are extended to be either inside the surface, outside the surface, or a mixture of both. In addition, another feature used is the D2 ratio, which is the ratio between parts of the D2 inside the surface, and parts outside. Values of the ten shape functions are summarized separately into a 64-bin histogram and concatenated to form the ESF feature vector with a length of 640.

ESF gives a significant boost in classifying objects scanned by near-range depth sensors. However, it is not quite applicable to objects in our point clouds. One reason is that the dimension of the ESF feature vector is so large. So, it tends to overfit sparsely-scanned objects and eliminate the difference between structures of different objects in the same category. Also, ESF distinguishes shape functions as inside, outside, or a mixture of both, which doesn't applicable to comparatively sparse points on objects segmented in LiDAR point clouds. Moreover, ESF is scale-invariant but the dimension information is very important for distinguishing different moving objects in street scene (e.g. vehicles in different sizes, pedestrians in different heights).

Since shape and orientation of the same moving object, either rigid or non-rigid, in neighboring frames is unlikely to change dramatically, we propose a novel point cloud descriptor to reflect

the similarity between objects in neighboring frames, named *Oriented Ensemble Shape Functions (OESF).* To derive *OESF* feature vector, the oriented bounding box with center $B(x_B, y_B, z_B)$ and orientation $E(\vec{e}_x, \vec{e}_y, \vec{e}_z)$ is estimated based on the convex hull of a point cloud object. Five shape functions D2, D3, Ax, Ay, and Az are adopted (see Fig 4.3). D2 and D3 are the same as the original ESF. For each point p_a on the object, we have a center vector $\overrightarrow{Bp_a}$ from the center of the bounding box B pointing to the point. The angles between $\overrightarrow{Bp_a}$ and $\vec{e}_x$, $\vec{e}_y$, $\vec{e}_z$ are respectively Ax, Ay, and Az. Results of each shape function are summarized separately into a 32-bin histogram and each histogram is normalized to be summed up as 100. Furthermore, to make OESF scale-variant, we restrict the range of the D2 histogram as $0 \sim 5\ m$. In addition, we introduce a layer distribution function. After rotating the observation to make its bounding box along with axis, layer distribution function divides the observation point cloud into 32 uniform layers and counts the percentage of points in each layer to derive a 32-bin histogram. Finally, the six 32-bin histograms are concatenated to derive the *OESF* feature vector with the length of 192.

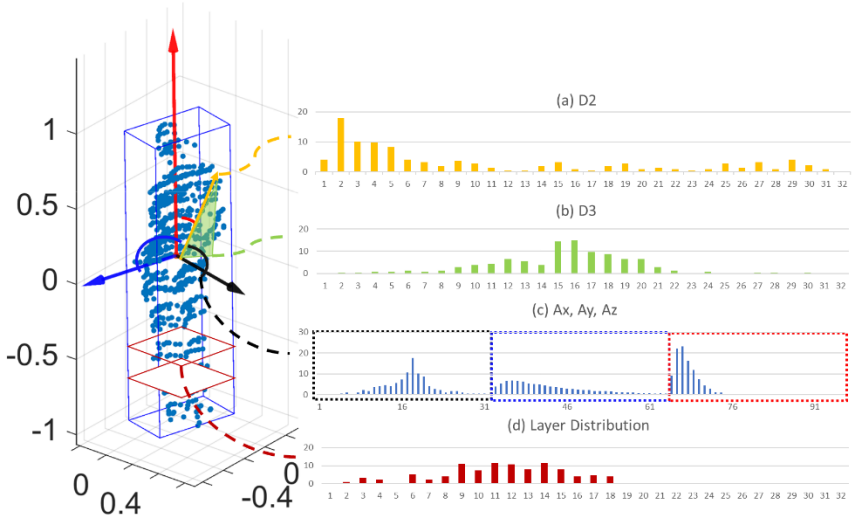


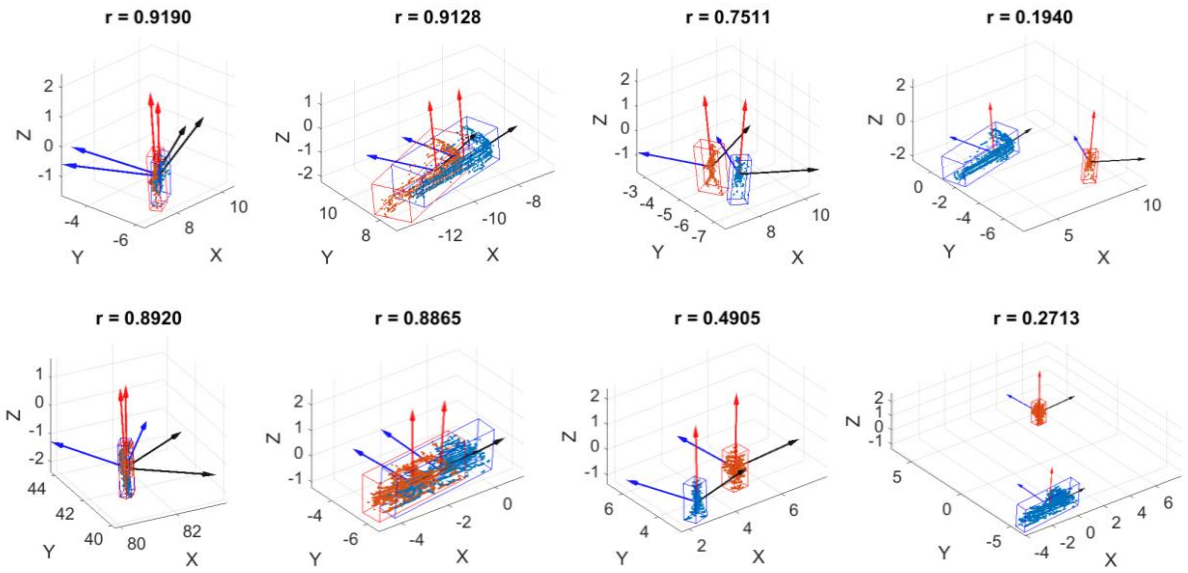
Fig 4.3 *Oriented Ensemble of Shape Functions (OESF)*: (a) D2; (b) D3; (c) Ax, Ay, and Az; (d) layer distribution function.

To test its effectiveness in measuring similarity between objects in LiDAR point clouds, we calculate *OESF* feature vectors of segmented moving objects from both KITTI Vision Benchmark, the scanner of which is a Velodyne HDL-64E laser scanner [7], and our Doppler LiDAR scans (see Table 4.1 and Fig 4.4). We use correlation coefficients between *OESF* feature vectors to measure the similarity of two objects. Our test proves the *OESF* feature vector can

clearly show the level of similarity between objects. For the same pedestrian or vehicle in neighboring frames, the correlation coefficients are likely to be more than 0.80. We manually segment objects in KITTI datasets while the objects in Doppler LiDAR are our detection results, therefore the correlation coefficients between the same objects in KITTI datasets are comparatively lager than those in Doppler LiDAR datasets. In Doppler LiDAR, for different objects in the same categories, their correlation coefficients decrease to below 0.70, which can decrease the risk of IDSW. Moreover, the correlation coefficients between objects in different categories (e.g. pedestrian and car) are as low as about 0.20. Therefore, with the point cloud descriptor term in scoring function, false alarm detections can hardly influence the tracking results.

Table 4.1 Test OESF by calculating correlation coefficients r between objects in point clouds

	Dataset	#Pair of Samples	Mean of r	Std. dev. of r
Same pedestrian in neighboring frame	KITTI	100	0.8848	0.0700
	Doppler LiDAR	150	0.8248	0.1097
Different pedestrians	KITTI	100	0.7752	0.0916
	Doppler LiDAR	150	0.6466	0.1442
Same car in neighboring frame	KITTI	50	0.9081	0.0649
	Doppler LiDAR	50	0.8016	0.1297
Different cars	KITTI	150	0.7051	0.1252
	Doppler LiDAR	150	0.5617	0.0967
Objects in different categories	KITTI	200	0.1348	0.0871
	Doppler LiDAR	200	0.2191	0.0593



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

Fig 4.4 Calculate correlation coefficient r between OESF feature vectors of objects in Kitti dataset (top) and Doppler LiDAR dataset (bottom): (a) the same pedestrian in neighboring frames; (b) the same vehicle in neighboring frames; (c) different pedestrians; (d) a pedestrian and a vehicle.

4.5 Managing and Confirming Track

Before confirming any track hypothesis, to manage tracks efficiently, track hypotheses are organized in several forms of data structure. Different tracks originated from the same root observation form a *target tree*, also called a *family*. Tracks from the same family are incompatible. If incompatible tracks are grouped into the same *cluster*, we can improve computational efficiency by processing each cluster independently. For that reason, *clustering* is adopted to decompose all families into smaller disjoint clusters. Clustering essentially divides a large tracking problem into several smaller sub-problems [48].

After clustering, we need to format global hypotheses. A *global (joint) hypothesis* is defined as a group of tracks that are compatible with each other. The final confirmed tracks compose the *most likely global hypothesis*, which is the combination of all compatible tracks that gives the highest overall score. The objective function of finding the *most likely global hypothesis* is:

$$max \sum_{l=1}^{n_{t_1}} s_l^{t_1} \kappa_l \dots \sum_{l=1}^{n_{t_i}} s_l^{t_i} \kappa_l \dots \sum_{l=1}^{n_{t_T}} s_l^{t_T} \kappa_l \quad (29)$$
$$\sum_{l=1}^{n_{t_i}} \kappa_l = 1, \kappa_l = 0 \; or \; 1 \;\; \text{(constraint)}$$

in which T is the number of trees; n_{t_i} is the number of tracks in tree t_i; $s_l^{t_i}$ is the score of track l in tree t_i; κ_l is the binary constraint deciding whether track l should be included in the *global hypothesis* or not. κ_l ensures one observation at most appear in one track thus that all tracks in the *global hypothesis* are compatible with each other. The multidimensional track assignment problem can be solved as a Maximum Weight Independent Set Problem (MWISP) [59]. Following [6], we use the algorithm proposed by [60] to solve the MWISP and find the most likely global hypothesis. As a result, a set of tracks giving the highest score is confirmed.

Notably, with new observations in each frame, branches of target trees are growing exponentially. Besides *clustering*, *N-Scan Pruning* is another way to keep MHT efficient, which prunes branches of target trees regularly to ease the computational burden [6], [46], [59]. The value of *N* determines how many frames a hypothesis be kept before it is confirmed or discarded (pruned). For a confirmed branch l in tree t_i at current frame k, any other subtrees in t_i who do not spawn from the same node *N* frames ago as l are pruned. Brunches in other target trees within the same cluster as l are pruned as well. The node l spawns from *N* frames ago is set as the new root node for tree t_i. For those target trees initialized less than *N* frames ago, no prune needs to be done (see Fig 4.5).

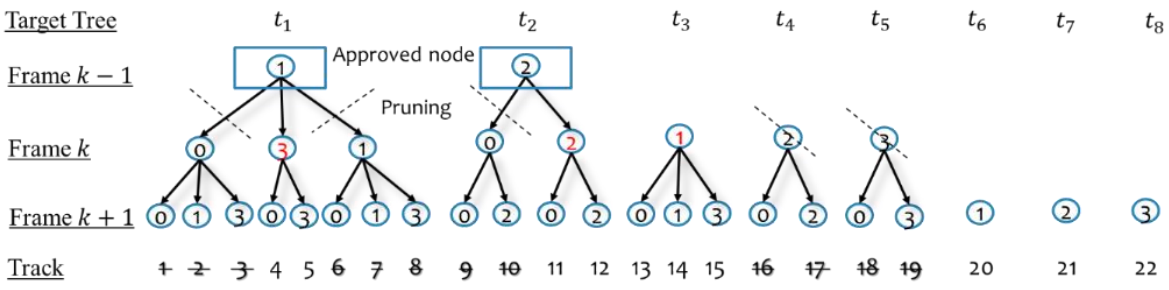


Fig 4.5 Pruning trees given *N* equals to 2. The red nodes are the new nodes after pruning.

5. EXPERIMENTS AND EVALUATION

Experiments have been conducted on the two datasets collected respectively by the static terrestrial Doppler LiDAR and mobile Doppler LiDAR. Both systems scan surroundings repetitively at a high frequency. We evaluate the performance of detection and tracking algorithms separately. Values for parameters used in the proposed method are summarized in Table 5.1. The Mahalanobis distance threshold λ_{d_m} is set as 3 according to the 3σ rule. L_0, the ratio of the prior probability of a hypothesis H_i and false alarm hypothesis H_0, is set as 10 because detected moving objects are less likely to be false alarm in Doppler LiDAR scans. V_0^{-1} is set to determine the conditional probability of null hypothesis, which should be a low value and set as 0.01. The *N* value for *N-Scan pruning* is set to determine how many frames a hypothesis should be kept if there are no consecutive new observations to update the track. A large *N* would increase the computational burden as much more hypotheses are kept, and IDSW errors are more likely to happen. If *N* is set too small, MHT would not be able to resolve assignment ambiguities with future measurements. As a trade-off, *N* is set as 5 with the idea that any moving objects should not be missed more than one second.

Table 5.1 Parameter values in the experiments

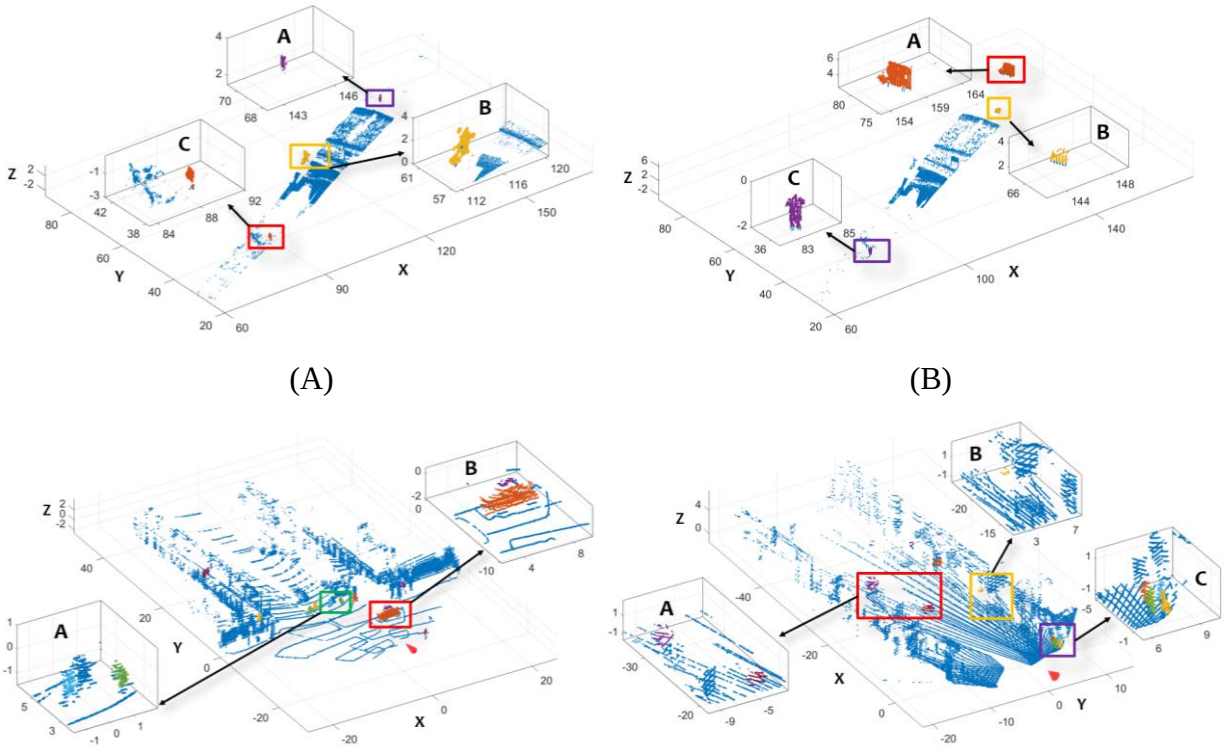
	Detection			Tracking				
Parameter	ϵ_v	$MinPts$	ϵ_T	λ_{d_m}	L_0	V_0	μ	N
Value	0.1	40	0.002s	3	10	100	0.3	5

5.1 Moving Object Detection

Due to the limited horizontal scanning angle, there are only a few moving objects in each frame (2 or 3 moving objects per frame on average) in the static terrestrial Doppler LiDAR scans. Pedestrians are the main moving objects while at the far end vehicles pass through the scanning region now and then. Two sequences of data (sequence A and B with the length of 72 frames and 81 frames respectively) are tested. The street scenarios scanned by the mobile Doppler LiDAR system are far more crowded with pedestrians, bicyclists, and vehicles (about 19 moving objects per frame on average). Two sequences of data (sequence C and D) are used for testing. In sequence C, it has a total of 90 frames. The mobile system moves fast (> 6 m/s) at first and then slows down and stops at a road cross waiting for people walking across the street. Sequence D

has a total of 107 frames, while the mobile system moves on a busy street at 8 m/s (about 30 km/h).

For quantitative evaluation, we compare the detection results with manually labeled ground truth of the moving objects. Precision (P), the ratio between the number of correct detections and all detections, and Recall (R), the ratio between the number of correct detections and ground truth, are calculated over all frames. The F1 score ($2R \times P/(R+P)$) is calculated with the average precision and recall of all frames. In addition, Object Recall (ObjRcl), the ratio between the number of frames in which a certain object is detected and frames in which the object is actually there, is also used [20]. Results are shown in Table 2.



(A) (B)

(C) (D)

Fig 5.1 Detection results on static (sequence A and B) and mobile datasets (sequence C and D)

Table 5.2 Precision, Recall, F1 Score, and Object Recall

	Seq ID	#GrdThth Obj	#Correct Obj	#Wrong Obj	Precision	Recall	F1 Score	ObjRcl
Terrestrial	A	107	102	9	0.9189	0.9533	0.9358	0.9667
	B	52	48	0	1	0.9231	0.9600	0.9757
Mobile	C	1683	1366	264	0.8380	0.8116	0.8246	0.8089
	D	1230	893	210	0.8096	0.7260	0.7655	0.7347

As sequence A and B are collected in non-crowded area with few moving objects per frame, the proposed detection algorithm performs well. In sequence C and D, the environment is more complex with ten times more objects appearing, resulting in more missing detection (false negative) and false alarm (false positive), thus the precision decreases to about 80%. Fast-moving objects (truck: site A in Fig 5.1(b)&(d); sedan: site B in Fig 5.1(b)&(c)) and slow-moving objects (pedestrian: site C in Fig 5.1(b)&(d)) in different sizes and various categories have been successfully detected and segmented. With the usage of adaptive distance threshold $\epsilon(r_i)$ in clustering, objects close to each other (less than 0.2 m) can be successfully segmented in most cases (For example, four close pedestrians at site C in Fig 5.1(d) have been well segmented).

When comes to errors, in sequence A and B, missing detection rarely happens but false alarms are found in several frames, whereas both missing detection and false alarms happen more frequently in sequence C and D. There are mainly two causes to false alarms. One cause is that part of a stationary object is moving due to external force. For example, the object at site A in Fig 5.1(a) is a stationary tree but it is detected as a moving object. That is because part of its branches is detected as moving. After growing, the entire tree is segmented as a moving object. Strictly speaking, it is not a false alarm because its branches are actually moving forced by the wind. However, in the evaluation we still regard such detection as wrong detection. Another cause to false alarm is that one object is segmented to two or more objects. It happens when parts of the object have no reflected points thus divide the object to several isolated parts. For example, the car at site B in Fig 5.1(c) is segmented into two objects in red and purple respectively. The purple part is the roof rack on the top of the car. The connection part between

the car and the roof rack is not fully scanned. As the car is very close to the scanner, a small distance threshold $\epsilon(r_i)$ (less than 5cm) is used in clustering. Therefore, the car has been segmented into two objects.

In some cases, missing detection happens when multiple objects close to each other are grouped as one object. For example, in Figure 11 (a), there are two people moving side by side on the left side of the street (see red rectangle). These two people appear in the scene from frame 3 to 81 and has been constantly segmented from frame 7 to 79. After frame 79, there are very few points reflected from these two people as they are moving out of the field of view.

Missing detection also happens if the object is right in front of the scanner and moving perpendicular to the beam. In this situation, the radial speed of points on that object is almost zero. For example, there are three pedestrians at site A in Fig 5.1(c). The pedestrians in sky blue and green are successfully detected and segmented while the one between them are missed due to the same reason we addressed above. Moreover, when severe occlusion happen, objects would be partially scanned or totally missed. For instance, at site B in Fig 5.1(d), the segmented object in yellow is a pedestrian while only the upper part of its body has been scanned.

In summary, for moving object detection, the method can detect arbitrary number of objects with high precision. Instead of deriving motion cues by processing a sequence of point clouds, moving objects can be efficiently detected and segmented frame by frame. False alarm can be effectively avoided instead of the condition that a large object is very close to the scanner. Missing detection would happen when multiple close-by objects are detected as one object or objects moving perpendicular to the beam have very few moving points reflected.

5.2 Moving Object Tracking

Due to the narrow horizontal scanning angle of static terrestrial Doppler LiDAR, most of fast-moving objects only appear once over all frames in sequence A and B. We need observe the same object in at least two frames to track it. Therefore, we do not evaluate the tracking algorithm on sequence A and B. Some tracking results are shown in Fig 5.2-4, in which the red arrow indicates the position and direction of the mobile Doppler LiDAR platform; the dark blue

points indicate static points; clustered point clouds in different colors represent individual detected moving objects; blue cubes with unique track ID are the tracking results.

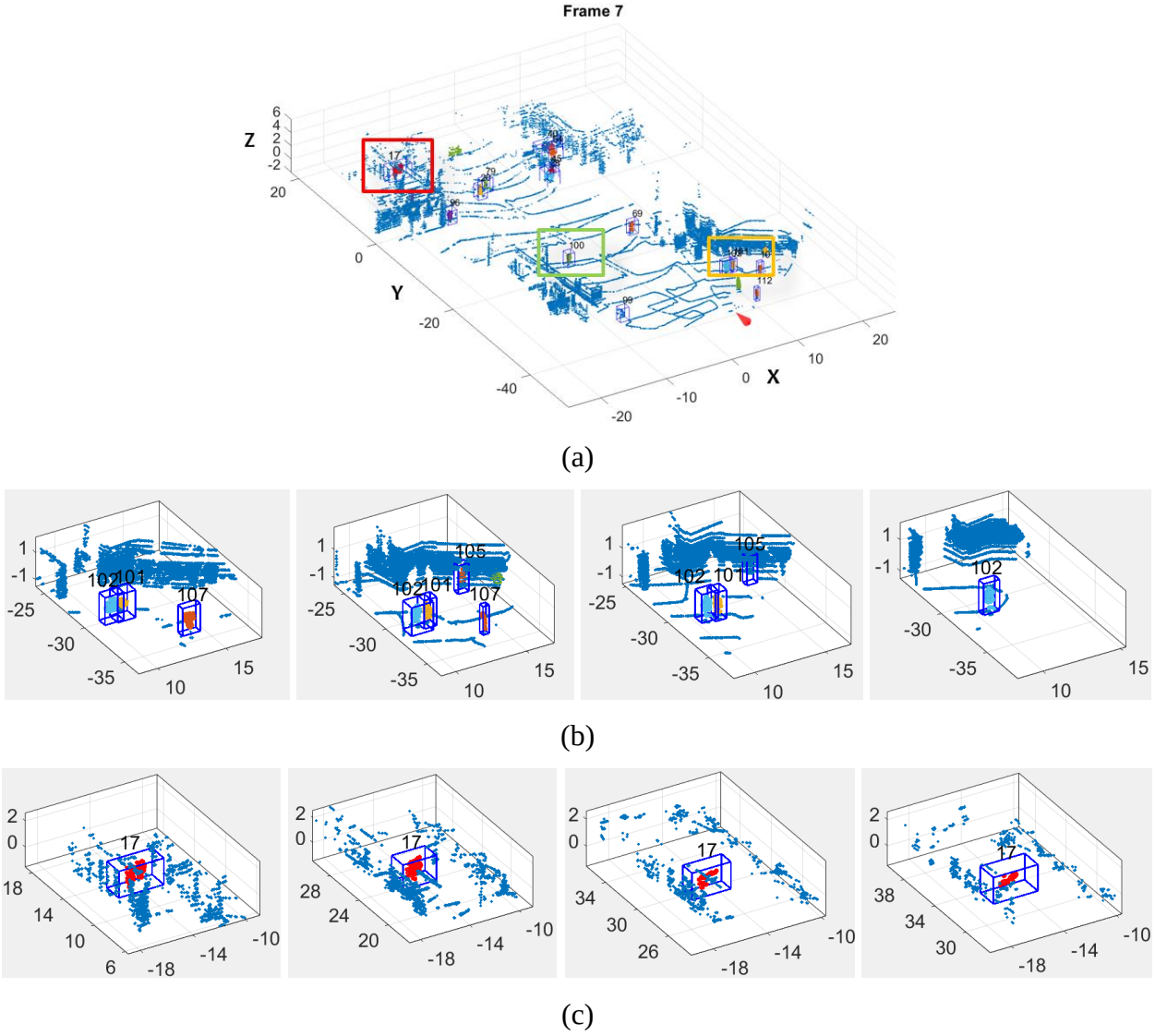


Fig 5.2 (a) Tracking results of frame 7 in sequence C; (b) Track #101, #105, #107 in frame 2, 10, 11 and 14 (Track #107 leaves from the scene, so do Track #101 and Track #105 in frame 14);; (c) Track #17 in frame 7, 49, 67 and 79.

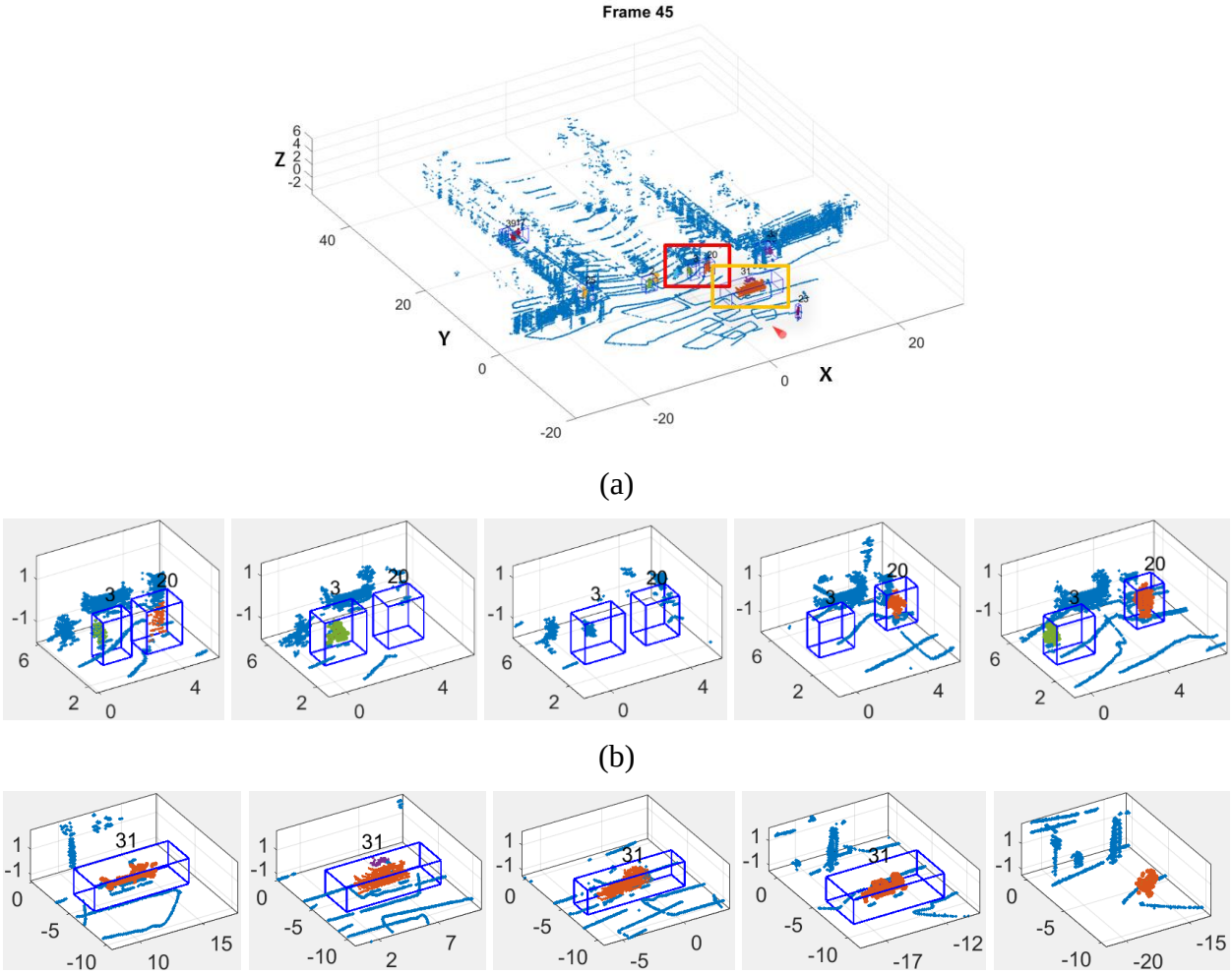
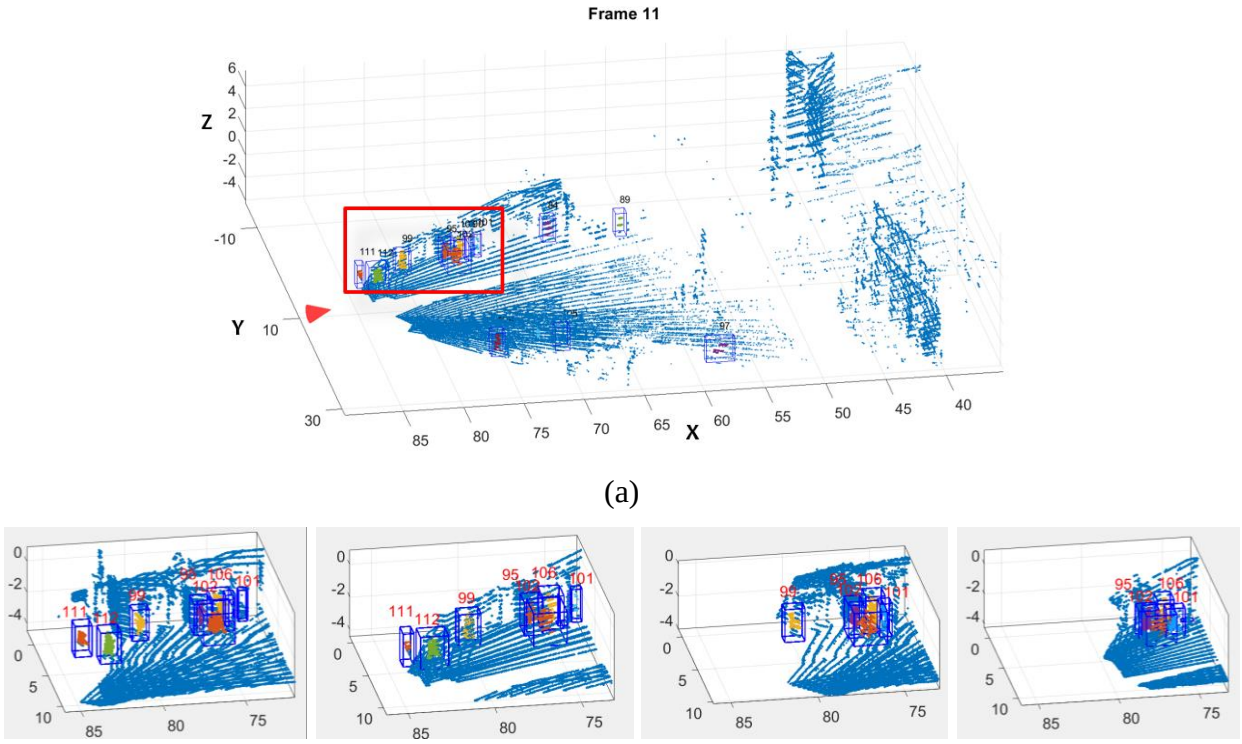


Fig 5.3 (a) Tracking results of frame 45 in sequence C; (b) Track #3 and #20 in frame 45 – 49 (Due to occlusion, the pedestrian in Track 3 isn't scanned in frame 47 and 48, and the pedestrian in Track 3 isn't scanned in frame 46 and 47. However, both tracks are kept and recovered when missing objects appear again); (c) Track # 31 in frame 41, 45, 49, 56, 57.



(a)

(b)

Fig 5.4 (a) Tracking results of frame 11 in sequence D; (b) a group of people (seven) in the red rectangle has been fully tracked until they leave the scanning region.

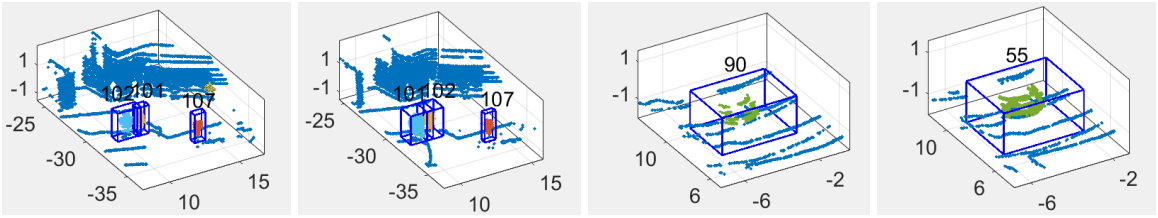
To evaluate the tracking results, we use metrics proposed by MOT16, a benchmark for Multi-Object Tracking in videos [8]. We count the number of True Positive (TP) and False Alarm (or False Positive, FP). A moving object that is missed tracking is a False Negative (FN). The mismatch error (IDentity SWitch, IDSW) is counted if a ground truth target has been labeled with different track numbers. To combine multiple sources of errors and evaluate multiple object tracking accuracy, Multiple Object Tracking Accuracy (MOTA) [8], [61] is widely used to evaluate a tracker's performance:

$$MOTA = 1 - \frac{\sum_k (FN_k + FP_k + IDSW_k)}{\sum_k (GT_k)} \tag{30}$$

Furthermore, to measure the track quality, we classify each ground truth trajectory as Mostly Tracked (MT, with at least 80% tracked), Mostly Lost (ML, with at most 20% tracked), and Partially Tracked (PT) [8]. Notably, a tracked moving object can be wrongly labeled with a new track ID in three scenarios. One scenario is that it leaves the field-of-view for several frames

(within N frames) and then reappears; another is that the structure of the moving point cloud object changes a lot due to its deformation or its relative position to the scanner; the third scenario is that its label exchanges with the label of other track when occlusion happens. All three scenarios are counted as IDSW.

To evaluate the effects of including the point cloud descriptor term and speed measurement in our MHT-PCD-Speed, we compare its results with the results of the original MHT and MHT only with PCD term (named MHT-PCD) (See Table 3).



(a) (b)

Fig 5.5 (a) The IDSW example in frame 7 and 8 of sequence C by the original MHT; (b) The IDSW example in frame 20 and 21 of sequence C by the original MHT and MHT-PCD.

Table 5.3 Track Quality Evaluation on Sequence C and D

Method	Seq.	Avg FN	Avg FP	Total IDSW	MOTA (%)	MT (%)	ML (%)
MHT-PCD-Speed	C D	3.66 2.25	0.40 0.75	31 19	76.45 72.42	66.13 51.65	9.68 27.47
MHT-PCD	C D	3.89 2.67	0.59 0.85	35 27	73.96 67.26	56.45 43.96	12.90 27.47
MHT	C D	4.83 3.56	0. 64 0.98	37 41	68.55 57.28	53.23 41.76	16.13 32.97

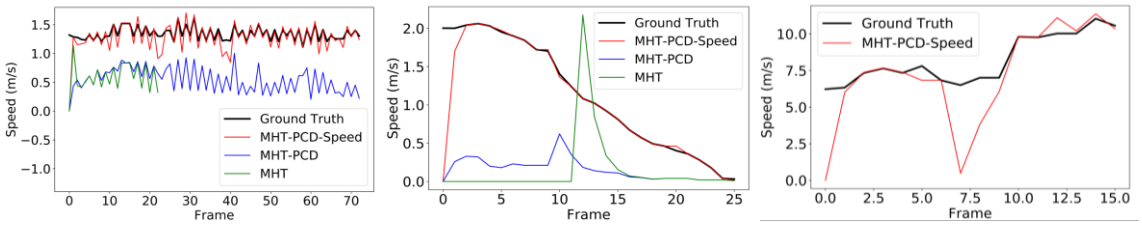
It shows that the proposed tracking method can track objects of various sizes in highly crowded street. For example, as shown in Fig 5.2 (b), three pedestrians have been tracked over 14 frames until they leave the field of view; in Fig 5.4 (b), seven pedestrians walking towards the mobile system have been completely tracked. In most cases, false alarms in detection would not influence tracking because they don't appear constantly in the scene and their structures are very

different from the real moving objects. For example, the roof rack on top of the car shown in Fig 5.1 (c) is a false alarm detection but it is not tracked at all (see Fig 5.3 (c)).

MHT keeps multiple possible tracks over N frames with the expectation that future inputs can resolve assignment ambiguities [48]. Therefore, tracks can be maintained even when missing detection happens. For example, as shown in Fig 5.3 (b), two pedestrians walking in opposite direction are tracked as Track # 3 and #20 respectively. Occluded by a moving vehicle (Track #31), the pedestrian in Track #3 is missed in frame 47 and 48. Similarly, the pedestrian in Track #20 is missed in frame 46 and 47. Both tracks retain and successfully retrack corresponding pedestrians when they appear again in later frames.

To evaluate the effects of including the point cloud descriptor term and speed measurement in our MHT-PCD-Speed, we compare its results with the ones of the original MHT, and the ones of MHT with PCD term (named MHT-PCD) (See Table 5.3). Compared to MHT-PCD-Speed, the original MHT is more likely to make IDSW error in a crowded scene. For example, Figure 13 (a) shows the same three people as shown in Fig 5.2 (b). While well tracked with MHT-PCD-Speed method, in the tracking result by the original MHT Track #101 and #102 switch track IDs with each other. That is because the original MHT forms tracks only based on spatial relation between track's predicted positions and detected objects' positions. MHT-PCD can kind of solve such problem as the structure of the point cloud is considered, but both the original MHT and MHT-PCD are subject to IDSW errors when objects rapidly change their dynamic states. For example, as shown in Fig 5.5 (b), there is a car coming to stop at the crossroad. It is originally tracked as #90 but switched to #55 in the next frame. The Kalman filter predicts the position based on previous position and moving speed. Without speed measurement, Kalman filter doesn't know the car is slowing down and predict position of the car with previously estimated speed, thus the predicted position is far away from the measurement position, out of the gating area. Therefore, both the original MHT and MHT-PCD start a new Track #55. This error can be avoided by setting a larger gating threshold λ_{d_m} or increasing the scanning frequency. However, both solutions are not optimal as IDSW are more likely to happen with larger λ_{d_m}, and increasing scanning frequency will definitely impose computational burdens. Moreover, if the object moves very fast (e.g. the car shown in Fig 5.3 (c)), the original MHT and MHT-PCD may totally lose its

track because under poor speed estimation the right candidate may not always be in the gating area. With the speed measurement estimated from Doppler images, those errors can be mostly avoided, resulting in less IDSW and more MT. As shown in Table 5.3, our extension to MHT by including the PCD term and Doppler speed measurement is proven to be effective as MOTA of MHT-PCD-Speed is the highest in both tests. For the overall performance in terms of MOTA the proposed MHT-PCD-Speed outperforms the original MHT by about 8% and 15% on sequence C and sequence D, respectively.



(a) (b) (c)

Fig 5.6 (a) Speed comparison of a pedestrian; (b) Speed comparison of a slow-down vehicle; (c) Speed comparison of the passing vehicle in Track #31.

We also compare the speed estimation to the ground truth data. Fig 5.6 (a) shows the speed of a walking person appearing in the scene for 73 frames with a speed about 1.3 m/s. The MHT-PCD-Speed not only completely tracks the person but also keeps a low RMSE (Root Means Square Error), which is 0.12 m/s. In contrary, the original MHT only tracks the person for 22 frames and then loses it. The MHT-PCD can completely track the person but with much worse speed estimation, whose RMSE is 0.81 m/s. Similarly, Fig 5.6 (b) shows the speed of a slow-down vehicle appearing in the scene for 26 frames. The MHT-PCD-Speed can well estimate the speed of the moving objects while the original MHT and MHT-PCD give bad estimation. Fig 5.6 (c) shows the speed of the passing vehicle in Track #31 given in Fig 5.6 (c). As explained before, the original MHT and the MHT-PCD totally lost that fast passing vehicle. The MHT-PCD-Speed well estimates its speed except in frame 7 and 8, where the vehicle is right in front the scanner and moving perpendicular to the beam. At that moment, the front part of the vehicle is moving away (reflected points with negative radial speed) while the rear part is moving close to the scanner (reflected points with positive radial speed), resulting a poor overall speed estimation, but such poor estimation can be recovered very soon. As a summary, the original MHT and

MHT-PCD requires a large amount of consecutive position observations to get a precise and accurate speed estimation, which is difficult to realize in complex street scenes, while our MHT-PCD-Speed can derive a precise speed estimation within three to five frames. It is also more robust to observation errors and can recover the state estimation very soon.

In summary, in the proposed MHT-PCD-Speed, the use of MHT makes the method robust to missing detection. It also makes it possible to fix wrong tracking with both previous and later observations. By introducing PCD term in scoring and considering structural similarity with *OESF*, the tracking method can process highly crowded scenes with few IDSW mistakes, and false alarm detections are unlikely to influence tracking results. Moreover, it has been demonstrated that the use of speed based on Doppler images enhance the tracking reliability and increase the precision of dynamic state estimation on moving objects, especially those whose speed is changing fast.

6. DISCUSSION AND CONCLUSION

In this study, we propose a novel data-driven model-free detection-based tracking approach, MHT-PCD-Speed, to detect and track moving objects in Doppler LiDAR scans. Our tests demonstrate the effectiveness and robustness of the proposed detection and tracking approach. Our major contributions include:

- The study reveals that the use of Doppler images can enhance the tracking reliability and increases the precision of dynamic state estimation.
- In detection, moving objects are clustered and segmented based on speed information with an adaptive ST-DBSCAN and a region growing technique. The detection method doesn't require multiple sequential frames of point clouds as input.
- In tracking, the dynamic state of moving objects is estimated with position observation and speed observation, increasing the precision of dynamic state estimation on moving objects, especially those whose speed is changing fast.
- A point cloud descriptor, *OESF*, is proposed and added in the scoring process of MHT, which allows managing tracks not only according to spatial closeness but also similarity in structure of detections in neighboring frames.

The performance of the detection approach and tracking approach has been discussed in detail. False positives occur when objects are moving perpendicular to the beam direction. Also, if the moving object has a large dimension, parts of its body may move close to the scanner while the other parts move away, resulting a wrong speed estimation. One solution to these two limitations is to mount several scanners and detect moving objects in point clouds collected by each scanner independently before merging those point clouds to the same coordinate system. With several scanners, it can be guaranteed an object doesn't move perpendicular to all beams simultaneously. This solution requires the information of precise position and orientation of each scanner, which is stored in the scanning system but not included in the output files. After we get access to the necessary information, we will further process point clouds from each scanner independently to improve the results.

Moreover, a more representative and robust point cloud descriptor is needed. Current OESF relies heavily on the point distribution of the objects, which is subject to objects' gestures as well as the scanning angle. One solution is to introduce vision-based sensors so that the current point cloud descriptor can be extended by including RGB features. With that additional information, close objects (such as the multiple hand-in-hand people whom we track as one object) can be separated, making the framework more resistant to IDSW errors. Another solution is to mine highly arbitrary features with deep learning methods, which requires massive data and high computational power.

We also attempt to reveal the applicability of our methods by summarizing the tracking results according to the category, dimension, and speed of moving objects. In terms of dimension, the object is regarded as small size if both the ranges of x and y of its point clouds (dx and dy) are less than 0.5 m; otherwise, it is regarded as of large size. In terms of speed, we set two thresholds ($0.5\ m/s$ and $1.0\ m/s$) to determine whether the object is moving slow, middle, or fast. Considering that some objects are rapidly changing their motion state, we label those objects who rapidly change their speed from slow to fast or from fast to slow within one second as "change speed rapidly" (see Table 6.1).

Table 6.1 Track Quality Evaluation on Objects of Different Types

Object Type	Classification	#Obj	MT (%)	ML (%)
Category	Pedestrian	142	56.34	20.42
	Vehicle	10	51.65	20.00
	Bicyclist	1	1	0
Dimension	Small (dx<0.5m & dy<0.5m)	140	56.42	20.00
	Large (dx>0.5m || dy>0.5m)	13	61.54	23.08
Speed	Slow ($\|v\| < 0.5\mathrm{m/s}$)	60	53.23	16.13
	Middle ($0.5 < \|v\| < 1\mathrm{m/s}$)	70	41.76	32.97
	Fast ($\|v\| > 1\mathrm{m/s}$)	18	33.33	5.56
	Changing rapidly	5	40.00	20.00

Currently, we do not have a conclusion about the applicability of our detection-based tracking method on Doppler LiDAR for several reasons. One reason is that we don't have a balanced dataset. Although we totally have about 3000 objects appearing over all frames, the number of unique objects is 153, with less than 10% of them are vehicles and only one bicyclist. Therefore, we cannot conclude whether our method works better on tracking pedestrians or vehicles.

Another reason is that the way to decide the size of objects is not very reliable as the size of point clouds of objects would change due to the movement of non-rigid body objects and the variation of scanning angle. Namely, a pedestrian who has a large stride may be labelled as a small object and large object alternately. Moreover, moving objects constantly interact with the environment, which can affect the tracking results. For example, the tracking results on the same pedestrian would be totally different if he is moving alone in an open field, moving together with a crowd of people, or moving close to static objects (e.g. walls). This study does reveal the advantages of the state-of-art Doppler LiDAR and prove the effectiveness of our detection-based tracking approach. However, a benchmark Doppler LiDAR dataset which considers and controls the environmental variables will need to be acquired before we can conclude the applicability of the proposed method and compare our method with other tracking methods.

REFERENCES

[1] A. A. Pourezzat, A. A. Sadabadi, A. Khalouei, G. T. Attar, and R. Kalhorian, "Object Tracking: A Survey," *Adv. Environ. Biol.*, vol. 7, no. 2, pp. 253–259, 2013.

[2] D. Prajapati and H. J. Galiyawala, "A Review on Moving Object Detection and Tracking," *Int. J. Comput. Appl.*, vol. 5, no. 3, pp. 168–175, 2015.

[3] K. Schindler, A. Ess, B. Leibe, and L. Van Gool, "Automatic detection and tracking of pedestrians from a moving stereo rig," *ISPRS J. Photogramm. Remote Sens.*, vol. 65, no. 6, pp. 523–537, 2010.

[4] Y. Cao, D. Guan, Y. Wu, J. Yang, Y. Cao, and M. Y. Yang, "Box-level segmentation supervised deep neural networks for accurate and real-time multispectral pedestrian detection," *ISPRS J. Photogramm. Remote Sens.*, vol. 150, pp. 70–79, 2019.

[5] H. Luo *et al.*, "Vehicle Detection in High-Resolution Aerial Images via Sparse Representation and Superpixels," *IEEE Trans. Geosci. Remote Sens.*, vol. 54, no. 1, pp. 103–116, 2015.

[6] J. M. Kim, Chanho; Li, Fuxin; Ciptadi, Arridhana; Rehg, "Multiple Hypothesis Tracking Revisisted," *Int. Conf. Comput. Vis.*, 2015.

[7] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the kitti vision benchmark suite," in *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, 2012, pp. 3354–3361.

[8] A. Milan, L. Leal-Taixé, I. Reid, S. Roth, and K. Schindler, "MOT16: A benchmark for multi-object tracking," *arXiv Prepr. arXiv1603.00831*, 2016.

[9] L. Patino, T. Cane, A. Vallee, and J. Ferryman, "Pets 2016: Dataset and challenge," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2016, pp. 1–8.

[10] L. Spinello, R. Triebel, and R. Siegwart, "Multimodal People Detection and Tracking in Crowded Scenes.," in *AAAI*, 2008, pp. 1409–1414.

[11] R. Kaestner, J. Maye, Y. Pilat, and R. Siegwart, "Generative Object Detection and Tracking in 3D Range Data - Kaestner et al. - Unknown.pdf," 2012.

[12] K. O. Arras, S. Grzonka, M. Luber, and W. Burgard, "Efficient people tracking in laser range data using a multi-hypothesis leg-tracker with adaptive occlusion probabilities,"

Proc. - IEEE Int. Conf. Robot. Autom., pp. 1710–1715, 2008.

[13] K. O. Arras, Ó. M. Mozos, and W. Burgard, "Using boosted features for the detection of people in 2D range data," *Proc. - IEEE Int. Conf. Robot. Autom.*, no. April, pp. 3402–3407, 2007.

[14] W. Xiao, B. Vallet, K. Schindler, and N. Paparoditis, "Simultaneous detection and tracking of pedestrian from panoramic laser scanning data," *ISPRS Ann. Photogramm. Remote Sens. Spat. Inf. Sci.*, vol. 3, no. July, pp. 295–302, 2016.

[15] Z. Yan, T. Duckett, and N. Bellotto, "Online learning for human classification in 3D LiDAR-based tracking," *IEEE Int. Conf. Intell. Robot. Syst.*, vol. 2017–Septe, pp. 864–871, 2017.

[16] W. Luo *et al.*, "Multiple Object Tracking: A Literature Review," pp. 1–18, 2014.

[17] B. Song, T.-Y. Jeng, E. Staudt, and A. K. Roy-Chowdhury, "A stochastic graph evolution framework for robust multi-target tracking," in *European Conference on Computer Vision*, 2010, pp. 605–619.

[18] M. Yang, T. Yu, and Y. Wu, "Game-theoretic multiple target tracking," in *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*, 2007, pp. 1–8.

[19] Z. Wang, "Laser-based detection and tracking of dynamic objects." University of Oxford, 2014.

[20] A. Dewan, T. Caselitz, G. D. Tipaldi, and W. Burgard, "Motion-based detection and tracking in 3D LiDAR scans," *Proc. - IEEE Int. Conf. Robot. Autom.*, vol. 2016–June, pp. 4508–4513, 2016.

[21] J. Cui, H. Zha, H. Zhao, and R. Shibasaki, "Laser-based detection and tracking of multiple people in crowds," *Comput. Vis. Image Underst.*, vol. 106, no. 2–3, pp. 300–312, 2007.

[22] O. Brock and F. Trinkle, J. Ramos, "Model Based Vehicle Tracking for Autonomous Driving in Urban Environments," *MIT Press*, vol. 1, pp. 175–182, 2009.

[23] H. Wang, B. Wang, B. Liu, X. Meng, and G. Yang, "Pedestrian recognition and tracking using 3D LiDAR for autonomous vehicle," *Rob. Auton. Syst.*, vol. 88, pp. 71–78, 2017.

[24] S. Yang and C. Wang, "Simultaneous egomotion estimation, segmentation, and moving object detection," *J. F. Robot.*, vol. 28, no. 4, pp. 565–588, 2011.

[25] J. Schauer and A. Nüchter, "Removing non-static objects from 3D laser scan data," *ISPRS J. Photogramm. Remote Sens.*, vol. 143, no. October 2017, pp. 15–38, 2018.

[26] S. Salti, F. Tombari, and L. Di Stefano, “SHOT: Unique signatures of histograms for surface and texture description,” *Comput. Vis. Image Underst.*, vol. 125, pp. 251–264, 2014.

[27] F. Moosmann and C. Stiller, “Joint self-localization and tracking of generic objects in 3D range data,” *Proc. - IEEE Int. Conf. Robot. Autom.*, pp. 1146–1152, 2013.

[28] Y. Yu, J. Li, H. Guan, and C. Wang, “Automated Detection of Three-Dimensional Cars in Mobile Laser Scanning Point Clouds Using DBM-Hough-Forests,” *IEEE Trans. Geosci. Remote Sens.*, vol. 54, no. 7, pp. 4130–4142, 2016.

[29] H. Yokoyama, H. Date, S. Kanai, and H. Takeda, “Detection and classification of pole-like objects from mobile laser scanning data of urban environments,” *Int. J. Cad/Cam*, vol. 13, no. 2, pp. 31–40, 2013.

[30] J. Anderson, R. Massaro, J. Curry, R. Reibel, J. Nelson, and J. Edwards, “LADAR: Frequency-Modulated, Continuous Wave LAser Detection And Ranging,” *Photogramm. Eng. Remote Sens.*, vol. 83, no. 11, pp. 721–727, 2017.

[31] D. Birant and A. Kut, “ST-DBSCAN: An algorithm for clustering spatial–temporal data,” *Data Knowl. Eng.*, vol. 60, no. 1, pp. 208–221, 2007.

[32] N. M. Nasrabadi, “Pattern recognition and machine learning,” *J. Electron. Imaging*, vol. 16, no. 4, p. 49901, 2007.

[33] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern classification*. John Wiley & Sons, 2012.

[34] J. MacQueen, “Some methods for classification and analysis of multivariate observations,” in *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, 1967, vol. 1, no. 14, pp. 281–297.

[35] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, “A density-based algorithm for discovering clusters in large spatial databases with noise.,” in *Kdd*, 1996, vol. 96, no. 34, pp. 226–231.

[36] A. Sampath and J. Shan, “Segmentation and reconstruction of polyhedral building roofs from aerial lidar point clouds,” *IEEE Trans. Geosci. Remote Sens.*, vol. 48, no. 3, pp. 1554–1567, 2010.

[37] J. Han, J. Pei, and M. Kamber, *Data mining: concepts and techniques*. Elsevier, 2011.

[38] A. K. Jain, “Data clustering: 50 years beyond K-means,” *Pattern Recognit. Lett.*, vol. 31, no. 8, pp. 651–666, 2010.

[39] L. Rokach and O. Maimon, “Clustering methods,” in *Data mining and knowledge*

discovery handbook, Springer, 2005, pp. 321–352.

[40] S. Guha, R. Rastogi, and K. Shim, “CURE: an efficient clustering algorithm for large databases,” in *ACM Sigmod Record*, 1998, vol. 27, no. 2, pp. 73–84.

[41] T. Zhang, R. Ramakrishnan, and M. Livny, “BIRCH: an efficient data clustering method for very large databases,” in *ACM Sigmod Record*, 1996, vol. 25, no. 2, pp. 103–114.

[42] W. Wang, J. Yang, and R. Muntz, “STING: A statistical information grid approach to spatial data mining,” in *VLDB*, 1997, vol. 97, pp. 186–195.

[43] M. Ankerst, M. M. Breunig, H.-P. Kriegel, and J. Sander, “OPTICS: ordering points to identify the clustering structure,” in *ACM Sigmod record*, 1999, vol. 28, no. 2, pp. 49–60.

[44] A. Hinneburg and D. A. Keim, “An efficient approach to clustering in large multimedia databases with noise,” in *KDD*, 1998, vol. 98, pp. 58–65.

[45] P. Konstantinova, A. Udvarev, and T. Semerdjiev, “A study of a target tracking algorithm using global nearest neighbor approach,” in *Proceedings of the International Conference on Computer Systems and Technologies (CompSysTech’03)*, 2003, pp. 290–295.

[46] S. Blackman and R. Popoli, “Design and Analysis of Modern Tracking Systems (Artech House Radar Library),” *Artech house*, 1999.

[47] T. Fortmann, Y. Bar-Shalom, and M. Scheffe, “Sonar tracking of multiple targets using joint probabilistic data association,” *IEEE J. Ocean. Eng.*, vol. 8, no. 3, pp. 173–184, 1983.

[48] A. Amditis, G. Thomaidis, P. Maroudis, P. Lytrivis, and G. Karaseitanidis, “Multiple Hypothesis Tracking Implementation,” *Laser Scanner Technol.*, 2012.

[49] D. B. Reid, “An Algorithm for Tracking Multiple Targets,” *IEEE Trans. Automat. Contr.*, vol. 24, no. 6, pp. 843–854, 1979.

[50] R. Richmond and S. Cain, “Direct-detection LADAR systems,” 2010.

[51] J. Shan and C. K. Toth, *Topographic laser ranging and scanning: principles and processing*. CRC press, 2018.

[52] Z. W. Barber, W. R. Babbitt, B. Kaylor, R. R. Reibel, and P. A. Roos, “Accuracy of active chirp linearization for broadband frequency modulated continuous wave ladar,” *Appl. Opt.*, vol. 49, no. 2, pp. 213–219, 2010.

[53] M. J. Halmos and D. M. Henderson, “Pulse compression of an FM chirped CO 2 laser,” *Appl. Opt.*, vol. 28, no. 17, pp. 3595–3602, 1989.

[54] A. R. Slotwinski, F. E. Goodwin, and D. L. Simonson, “Utilizing GaalAs Laser Diodes As A Source For Frequency Modulated Continuous Wave (FMCW) Coherent Laser Radars,” no. June 1989, p. 245, 1989.

[55] T. J. Pingel, K. C. Clarke, and W. A. McBride, “An improved simple morphological filter for the terrain classification of airborne LIDAR data,” *ISPRS J. Photogramm. Remote Sens.*, vol. 77, pp. 21–30, 2013.

[56] A. Yilmaz, O. Javed, and M. Shah, “Object tracking: A survey,” *Acm Comput. Surv.*, vol. 38, no. 4, p. 13, 2006.

[57] W. Wohlkinger and M. Vincze, “Ensemble of shape functions for 3D object classification,” *2011 IEEE Int. Conf. Robot. Biomimetics, ROBIO 2011*, pp. 2987–2992, 2011.

[58] R. Osada, T. Funkhouser, B. Chazelle, and D. Dobkin, “Matching 3D models with shape distributions,” *Proc. - Int. Conf. Shape Model. Appl. SMI 2001*, pp. 154–166, 2001.

[59] D. J. Papageorgiou and M. R. Salpukas, “The maximum weight independent set problem for data association in multiple hypothesis tracking,” in *Optimization and Cooperative Control Strategies*, Springer, 2009, pp. 235–255.

[60] P. R. J. Östergård, “A new algorithm for the maximum-weight clique problem,” *Nord. J. Comput.*, vol. 8, no. 4, pp. 424–436, 2001.

[61] R. Stiefelhagen, K. Bernardin, R. Bowers, J. Garofolo, D. Mostefa, and P. Soundararajan, “The CLEAR 2006 evaluation,” in *International evaluation workshop on classification of events, activities and relationships*, 2006, pp. 1–44.