MULTI-OBJECTS TRACKING BASED ON 3D LIDAR AND BI-DIRECTIONAL RECURRENT NEURAL NETWORKS UNDER AUTONOMOUS DRIVING SCENARIOS

By

PUJIE XIN

A thesis submitted to the

School of Graduate Studies

Rutgers, The State University of New Jersey

In Partial Fulfillment of the Requirements

For the Degree of

Master of Science

Graduate Program in Industrial and Systems Engineering

Written Under the Direction of

Zhimin Xi

And approved by

New Brunswick, New Jersey

January, 2020

ABSTRACT OF THE THESIS

Multi-objects Tracking Based on 3D LiDAR and Bi-directional Recurrent Neural Networks Under Autonomous Driving Scenarios

By PUJIE XIN

Thesis Director:

Zhimin Xi

Multi-Objects Tracking (MOT) is an important topic in navigation, where robots or vehicles should interact safely with the moving objects in the environment. The navigation system can hardly make a path plan if there is no position and velocity information of the moving objects. Generally, moving objects tracking includes three stages which are sensor measurement preprocessing, data association, and kinetic states estimation. This thesis presents a new approach to improve the matching precision in the data association stage by combining more characteristics of the targets and their kinetic states detected by sensors. In more details, different perception systems infer different characteristics of the moving objects, which will help distinguish the moving objects, and thus improve the matching precision. However, it is hard to use these specified characteristics of the targets in widely used Single Object Tracking (SOT) strategies such as the Global Nearest Neighbor (GNN) approach, the Joint Probabilistic and Data-association (JPDA) approach, and the Multi-hypothesis Tracking (MHT) approach. Generally, moving targets are viewed as point-like targets in SOT strategies, which means that many characteristics are ignored in the matching process and these SOT methods just associate the data by estimating the probability of each association and select the association with the highest probability. Only the object-to-hypothesis distance was used to compute the probability and the Hungarian algorithm provides an optimal solution to the distance matrix, which is considered as the optimal assignment in the data association. In this thesis, a new method is proposed to calculate the cost matrix considering both the distance matrix and the pose of the moving targets. To compute the new assignment matrix with both distance and pose information, bidirectional Recurrent Neural Network (Bi-RNN) is proposed to input the object-to-hypothesis distance matrix and output the optimal assignment matrix. The loss function of the Bi-RNN is simplified as the mean square error. Multiple Object Tracking Accuracy and Precision, i.e., MOTA and MOTP, are standard and widely used matrix to assess the quality of MOT. In this thesis, they are hence used to evaluate the performance of the proposed tracking method. Experimental datasets, i.e. the KITTI datasets, are used to demonstrate the effectiveness of the new algorithm.

TABLE OF CONTENTS

List of Tables

List of Illustrations

1.Introduction

1.1 Multi-Objects Tracking (MOT) in Autonomous Driving

MOT has immediate application to autonomous driving scenarios. Autonomous driving can be divided into 6 levels according to Society of Autonomous Engineer.

	Driver	Vehicle
Level 0 No automation	In charge of the driving	Respond only to the manipulation from driver, but also provide warnings in environment.
Level 1 Driver assistance	In charge of the driving but can also get basic help in some situations.	Provide basic help to drivers, containing automatic emergency braking or lane keep support.
Level 2 Partial automation	Must stay alert even the vehicle assumes some basic driving tasks	Automatically steer, accelerate and brake in limited mood
Level 3 Conditional automation	Must be always ready to take over the vehicle when the self-driving system breaks down	Can take full control steering, acceleration, and braking under certain conditions.
Level 4 High automation	Sometimes need to take full control of the vehicle when self-driving system breaks down.	Can assume all driving tasks under nearly all conditions without any without any driver attention.
Level 5 Full automation	No human driver required.	In charge of all the driving and can operate in all environments without need for human intervention.

Table 1. Levels of autonomous driving

People pursue for the Level 5 full automation in autonomous driving. In [1], level 5 autonomous vehicles don’t need drivers to monitor the vehicle constantly and the vehicle is setting the indicators, is accelerating, braking and steering automatically. In their work, they divided autonomous driving into three parts: perception, planning and control. [2]

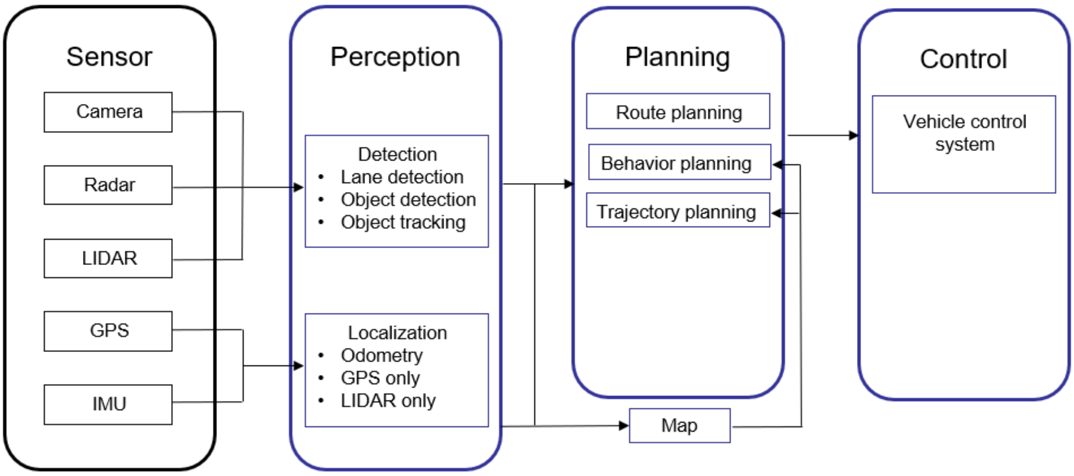


Figure 1. Perception, planning, and control in autonomous driving stack

First, data from different sensors are fed into perception function. The lane detection algorithm and MOT algorithm help build up holistic information of the environment. The autonomous vehicle must sense and perceive their surrounding environment and then change its kinetic states in real-time. There are several different sensors such as laser range finder, radar and cameras are utilized to sense the surrounding environment of the autonomous vehicle. These heterogeneous sensors simultaneously capture various physical attributes of the environment. By using the sensor fusion method, we can get a reliable and consistent perception of the world. If we can extract more features of the moving targets from the sensor and associate the new observations and the existing tracks with these features, we may get more reliable tracking results. In this thesis, LiDAR (Light Detection and Ranging) is the sensor of our tracking system. Except for the detection algorithm, the perception algorithm also contains the localization algorithm which helps to localize the vehicle. Once the surrounding environment is detected, the vehicles need to plan the route and predict the relative velocity and acceleration. Then the perception result comes to planning part, which includes route planning, behavior planning and trajectory planning. Given the assumption that the vehicle has perceived the environment nearby and it understands enough details in the environment, it remains to plan the actions for the vehicle. Route planning makes a global path plan using the given map and then behavior planning calculates the local path for overtaking and evasion maneuver.

Autonomous driving is one of the scenarios of MOT. In our circumstance, the LiDAR is the sensor in our system. And we conduct MOT based on LiDAR data. LiDAR is a remote sensing method that use light in the form of a pulsed laser to range finder, which provides distance to obstacles in environment directly. It is particularly attractive to autonomous driving applications because of their lightweight, efficient and reliable. LiDAR produces point clouds, a set of points in space, which measure many points on the external surface of objects near the LiDAR. The first step in tracking is to extract the moving targets. In the case studied in this thesis, we need to firstly extract vehicles from the point clouds. After this extraction, we will get the pose, shape, and location of each vehicle in the environment.

Once we detect the moving targets, we will initialize the tracking method using the Global Nearest Neighbor (GNN) [44] and we will get the hypothesis-to-object distance matrix. The size of this matrix is N by M where N is the number of the hypothesis and M is the number of newly detected objects. Each cell in this matrix is the distance between the predicted position of the hypothesis and the detected position of the newly detected objects. Combining the shape and pose information in objects detection, we will get a new cost matrix processed by a trained Bidirectional Recurrent Neural Network (BiRNN). The input of the BiRNN is the hypothesis-to-object distance matrix and the hypothesis-to-object pose matrix. The target is to associate the new detections to existing tracks (hypothesis). This target can be solved as an assignment problem. The output of the BiRNN is a new cost matrix and we still use Hungarians algorithm to solve the assignment problem. The assignment result is used to associate the hypothesis to new detections and to update the GNN tracking method.

This thesis builds up a complete detection and tracking framework using LiDAR data and proposes a new method in data association stage to decrease the mismatching probability and improve the successful tracking rate.

1.2 Related Works of MOT in Autonomous Driving

In Section 1.1, we briefly introduce the role of MOT in autonomous driving. In this section, we discuss some of the achievements in autonomous driving and also identify the main MOT challenges in autonomous driving. MOT serves for detecting and tracking the kinetic states of the moving objects around the vehicle. The widely used sensors include LiDAR, RADAR and cameras. In order to cope with uncertainty in sensor measurement, people use Bayes filters in MOT to predict the state of the objects. According to[3], MOT method can be divided into six classes. They are traditional, model based, stereo based, grid map based, sensor fusion-based and deep learning-based method.

Traditional MOT methods generally include three steps, which are segmentation of data, data association and Bayesian filtering. In the segmentation step, vehicles are recognized and extracted by the recognition algorithm. Then they are associated at current time stamp with the existing tracks in previous time stamp. The assignment of the data association is used to update the existing tracks, where the Bayesian filtering serves for this process. They proposed a traditional method for MOT using 3D LiDAR sensor. They firstly segment the 3-dimensional point cloud based on Euclidean distance and then extract the vehicles. Newly observed vehicles are next assigned to existing tracks in previous scan using the nearest neighbor algorithm. The states of the vehicles are estimated using particle filter. Then they use bounding box to represent the pose and the position of the vehicles. [6], and the data association is solved by the Multiple Hypothesis Tracking (MHT) algorithm, which has better performance by maintaining the association hypothesis for several time stamps and find out the most probable hypothesis in the period.

Sensor fusion based MOT detects and tracks the vehicles using various kinds of sensors like LiDAR and RADAR to explore the reliability of perception of the environment.[4] presents the sensor fusion-based method of MOT which is adopted by the self-driving car Boss[7] (2007 DARPA Urban Challenge). The MOT system can be divided into two parts. The first part generates point model or box model to describe the detected moving objects. Then the second part associates the features to the detected model in the first part. If this association is built up, then they generate a new observation to update the previous track, otherwise, a new track is initialized. The updating process uses Kalman Filter. Later, they extended the system and installed camera to identify the category of the moving objects and further promoted the detection reliability by using LiDAR and RADAR.

Deep learning-based MOT is a method which builds up on the deep neural networks for detecting and localizing the moving objects. Moreover, the geometric information of the vehicles can also be detected at the same time. Most widely used sensor in deep learning-based MOT is based on cameras. Several works are proposed. [Dynamic path planning for autonomous driving on various roads with avoidance of static and moving obstacles] proposed the MOT method based on convolutional neural networks using monocular camera. However, several works based on LiDAR also came into our view. They use the Kalman Filter for filtering and build up a recurrent neural network (RNN) to solve the data association problem. They proposed a specified loss function to train the RNN and the trained network is used to solve the assignment problem.

Model based MOT doesn’t need segmentation of the data and data association, in adverse, it utilizes the physical models of the sensors and the geometric models of the moving objects directly, because the geometrical features in each object help match the existing tracks. [8] presents the model-based method in autonomous car in 2007 DARPA urban challenge. They combine Kalman filter and Rao-Blackwellized particle filters. In their work, moving objects, especially the vehicles, are detected by LiDAR using differences over LiDAR data. Sensor data are updated by the detected vehicles’ pose and geometry.

Stereo vision-based MOT and Grid map-based MOT are also widely used, which is out of the scope of this thesis. Interested readers can refer to the references [9][10].

1.3 Problem statement

In the preceding section, the MOT methods in autonomous driving scenarios have been introduced. Each type of MOT algorithm depends on the sensor data and its specified detection algorithms. For example, if we hope to use deep learning-based detection algorithm, the detection algorithm should detect the pose and the position of the objects. In our work, we hope to design and implement an integrated framework using 3D LiDAR to detect and the track the moving objects in autonomous driving scenarios, where the MOT method is deep learning-based MOT algorithm. The next problem is to design the deep learning-based MOT algorithm.

In deep learning-based MOT, we need to design an algorithm, which is capable to deduct the assignment matrix in data association based on not only object-to-hypothesis distance matrix, but also other features of the moving objects such as the pose and the shape, in order to efficiently and accurately detect and track the moving objects. On one hand, we hope to build up an object detection model based on LiDAR; on the other hand, we need to build up reliable data association relationships using more characteristics. To achieve the first goal, we utilize a vehicle detection model called PIXOR. Because object detection is not the main contribution of our work, we will only briefly introduce this part later. The second target is to build up a new data association algorithm, where we train a BiRNN to transform the hypothesis-to-object distance matrix to a new cost matrix, and the assignment result is calculated from the new cost matrix. At last, we need to make an integral system which combines the detection and MOT together, and to test the new framework using labeled datasets.

1.4 Contribution

The main contributions of this thesis include the following:

- Solve an assignment problem in data association stage for MOT considering shape similarity. Instead of using the well-studied Bayesian filtering, the proposed algorithm is capable of performing all multi-target tracking tasks and enable us to assign the hypothesis to existing tracks based on more features of the moving objects. Our algorithm is capable of using any features extracted by the detection algorithm in data association and this increases the compatibility to many detection algorithms which extract more detailed features of the moving objects.
- We present a framework that achieves the detection and tracking of moving objects more effectively using a new data association method.

2. Literature review

In past decades, Multiple Object Tracking (MOT)is under active research. The objective of MOT is to precisely calculate the trajectories of the detected moving objects and hold their unique identities under all kinds of noise. Understanding the kinetic states of the existing objects in the environment is of vital importance to improve the reliability of the MOT algorithm. The output of the MOT algorithm is the input to the localization, mapping and motion planning [5]. In past decades, many MOT algorithms based on high resolution Radar [6] and traditional cameras [12] were proposed. Many sensor fusion methods, which utilized two or more kinds of sensors were also proposed [15]. Camera-based approaches need to face the inherent challenges like different illumination condition and background, fast-moving or random motion of targets and occlusion of the objects [16]. Although many algorithms were proposed to fix these inherit problems and showed promising tracking results and performance, the camera-based MOT algorithm still failed in some circumstance. Restricted to these inherit problems, researchers started working on an alternative sensor for MOT, and several MOT algorithms based on Light Detection and Ranging (LiDAR) [13] were proposed. LiDAR's advantages include higher accuracy, fast acquisition and processing, weather and light independence, higher data density. LiDAR provides a huge amount of points to describe spatial information of the surrounding environment. To model the surrounding scenarios, MOT algorithms need to continuously perceive the kinetic states of the surrounding moving objects. Generally, MOT is composed of two parts, which are detection and tracking. Tracking-by-detection is the widely used framework [30][32][33].

The detection problem can be viewed as the process to extract the target objects which are buried in the environment. Typically, detection methods can be divided into two categories. the first is 3D voxel grid [18][19]. This kind of methods transform the point cloud into a regularly spaced 3D grid. Each voxel cell contains a scalar value (occupancy information) or vector data (probabilities). Then, the 3D convolutional neural network is used to extract the objects from the voxel grid. The main drawback of this algorithm is the voxel grid is very sparse, because the point cloud itself is very sparse in the nature environment. This leads to a great waste of computation source. As a result, the computation speed is very low, which is about 1 to 2 frame per second. Contrary to the spaced 3D grid method, the second method is to project the point clouds in 3D space to a 2D plane. And then, they divided the 2D plane into grids. Since the point clouds are represented by the 2D grids, hand-crafted features can be extracted. Then the moving objects can be extracted using these features. The commonly used projection methods include range view and birds' eye view. Range view projects the point clouds on the 360-degree panoramic surface and birds' eye view is the top view of the point clouds. In 2D projection-based methods[21], the point clouds are more compact, this helps to promote the calculation efficiency and speed. However, this method also losses some information because of the compact of space from 3D to 2D. To improve the detection speed and decrease the loss of information, an algorithm called PIXOR

(Oriented 3D object detection from pixel wise neural network predictions) is proposed [17]. They utilized the birds' eye view representation for its computationally friendly, and also kept the metric space which allows the model to explore priors about the size and shape of the object categories. This algorithm performed good in both detection accuracy and detection speed. As the output of PIXOR detection algorithm, we can get accurate oriented bounding boxes of moving objects. The Bounding boxes include the position, size and pose of the moving objects.

In the tracking problem, researchers divided this topic into three areas, data association, state estimation, and track management. The most important part is data association. Data association is the process of associating uncertain measurements to known measurements. The uncertainty embedded in the data association process are generally calculated by Bayesian analysis [22]. The uncertainty mainly comes from the detection error and the unmodeled dynamic of the moving objects and usually be tackled by particle filters or Kalman filters. Two classes of association methods are used in data association, one is the probabilistic filter and another is the deterministic filter. Global Nearest Neighbor (GNN) is a typical deterministic filter, it associates object with the closest measurement based on Euclidean and Mahalanobis distance between all measurements and tracks [13][14]. Traditionally, the solution to probabilistic filter is addressed with Hungarian or Munkres algorithm. [24] The representative of probabilistic filter is Probabilistic Data Association Filter.[23] Rather than associate the detections and existing tracks by assigning one detection to one existing track, JPDA associate one detection with all existing tracks. And the tracks are updated with weighted values from all detections. The weights of detection are updated in each iteration. We use Extended Kalman Filter under the assumption that the error follows normal distribution. This method only associates the existing tracks and measurement in consecutive time step. However, if data are associated at each time step, and prone all errors like false alarms before, it is the Multiple Hypothesis Tracking (MHT) method [15]. MHT builds up a hypothesis tree which grows exponentially, then it prunes the hypothesis tree mandatorily or just get the optimal Bayesian solution without pruning. Without pruning is not practical for its exponentially increase complexity.

For kinetic states estimation of the moving objects, the first step is to build up kinetic model for all objects in the environment including the sensor. In mobile robotics applications, complications arise when moving targets are usually buried within background cluster and their measurement are influenced by their constantly changing appearance. To tackle with this difficulty, one paper [31]mentioned that there are two approaches, one is model-free approaches and the other is model-based approaches. Model-free approach builds up detection based on motion cues and there is no semantic information needed in detection. Examples of model-free dynamic obstacle detection and tracking include systems deployed in the DARPA Urban Challenge [34][35]. Contrary to the model-free approach, in model-based approaches, objects are first detected based on a parametric model of its shape and then tracked as a separate track. Because objects are detected based on their shape characteristic, the potential moving hazard will not occur. However, only those objects which have known parametric model can be recognized. Combining the data association in previous part, MHT filter can be used in model-based approach. In [36][38] which focuses on people detection, they trained the model for leg detection and use MHT to track the people.

In recent years, people [37] tried to solve GNN with Recurrent Neural Network (RNN), although it is a very classic data association method. They modeled the appearance and motion of moving objects independently. The combined information is processed with convolutional neural network. In addition, people [26] proposed Dual Matching Attention Networks (DMAN) to solve the MOT. Spatial and temporal attention mechanism are used in their work. In [27], they proposed a method solving the association based on Bidirectional Recurrent Neural Network (BiRNN). In their work, they trained BiRNN to transform the distance matrix to a new detection-to-track matrix, where the loss function is composed of multiple objects tracking accuracy and precision at the same time. In this way, they can use their network to transform any distance matrix to new cost matrix. Then the generated cost matrix is solved by Hungarian or Munkres algorithm.

3. System Overview

The experimental setup is described in this chapter. In section 3.1, the coordinate system is presented. In section 3.2, a detailed experimental scenario example is provided. In section 3.3, the trained BiRNN is introduced.

3.1 Coordinate System

Coordinate system is the footstone in autonomous navigation system. Any navigation system typically has many 3D coordinate frames that change over time, such as a world frame, base frame, sensor frame.

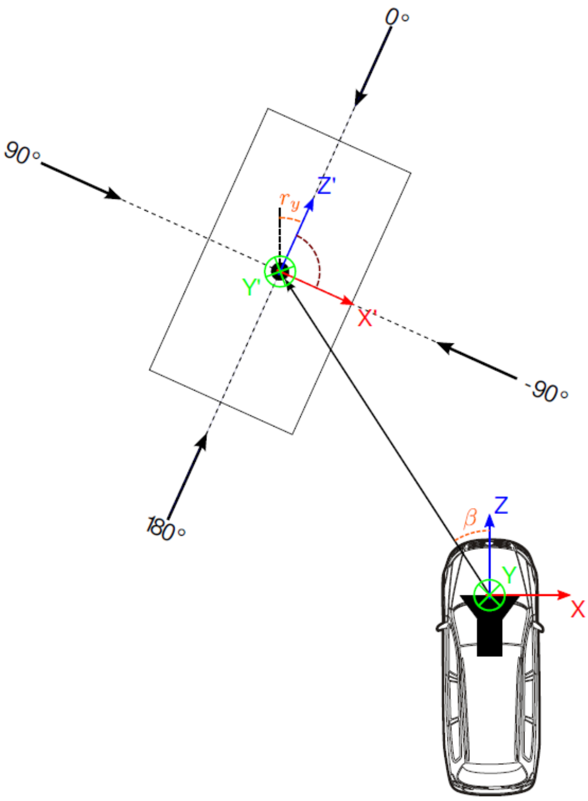


Figure 2. Coordinate of the vehicle[29]

The coordinate system is divided into two parts, where LiDAR is mounted on the moving vehicle, we call it LiDAR coordinate system$[X\ Y\ Z]^T$. The coordinate systems are defined the following way, where directions are informally given from the drivers view, when looking forward onto the road. And the moving objects coordinate system is $[X' \; Y' Z']^T$. Figure 1 shows their relationship, where β is the observation angle of object, ranging [-π, π] and rotation r_y around Y-axis in LiDAR coordinates [-π, π].

3.2 Dataset

We make use of a well-known autonomous navigation dataset, KITTI dataset [28][29] in our training and test of our algorithm. The KITTI dataset has been recorded from a moving platform while driving in and around Karlsruhe, Germany. It includes camera images, laser scans, high-precision GPS measurements and IMU accelerations from a combined GPS/IMU system. The main purpose of this dataset is to push forward the development of computer vision and robotic algorithms targeted to autonomous driving.

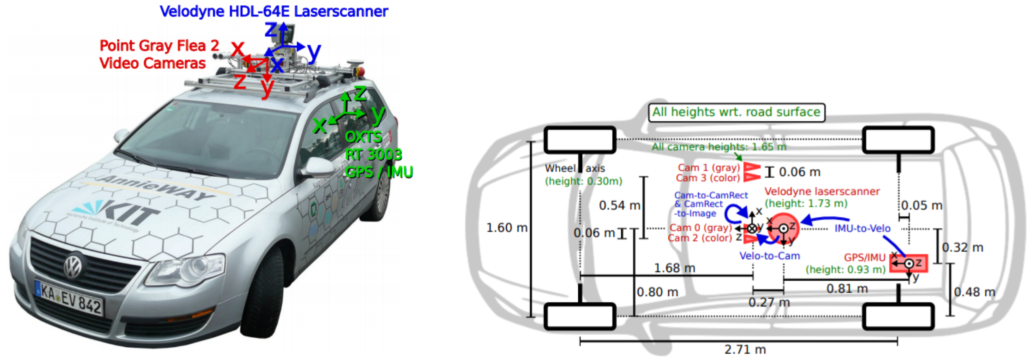


Figure 3. Sensors on vehicle[29]

We only use the data from the LiDAR, which is Velodyne HDL-64E rotating 3D laser scanner, 10 Hz, 64 beams, 0.09 degree angular resolution, 2 cm distance accuracy, collecting ∼ 1.3 million points/second, field of view: 360◦ horizontal, 26.8◦ vertical, range: 120 m. LiDAR generates point cloud and then we use a reliable vehicle detection algorithm to detect vehicles.

3.3 Vehicle Detection

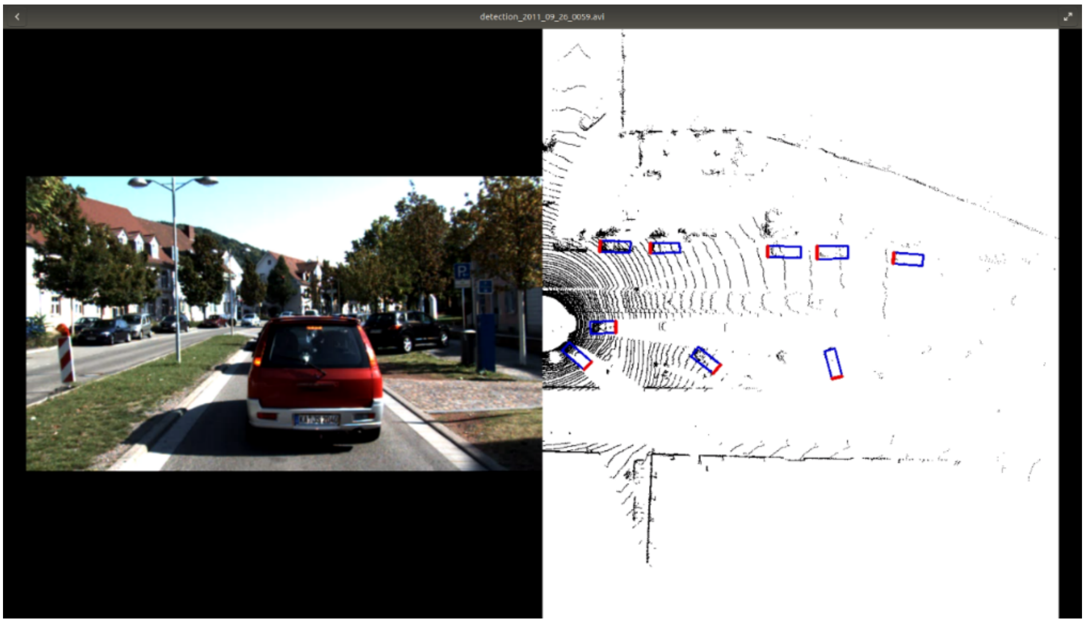


Figure 4. Birds-eye view point clouds

An efficient and reliable vehicle detection algorithm is the cornerstone of the whole tracking system. Here we use a vehicle detection algorithm called PIXOR to detect the pose, size and the position of the vehicles with respect to the LiDAR. They propose an efficient 3D object detector that is able to produce very accurate bounding boxes given the LiDAR point clouds. In figure, the left part is the detection from the camera, and the right part shows the detection results from the LiDAR simultaneously. In this picture, the blue bounding box generated by the algorithm is contour the vehicle, and the red lines on the blue bounding boxes are heading of each vehicle. The precision of this algorithm is 75% according to their test result. And this result is the input to our algorithm and benchmark algorithm.

3.4 Tracking System Modeling

In tracking part, the mean and the error covariance of the joint state vector x are updated at each iteration using the Extended Kalman Filter (EKF). The EKF is the nonlinear version of Kalman Filter (KF) and is considered the de facto standard in the theory of nonlinear state estimation in navigation system.

The input of the tracking step is the measurement from odometry and laser range finder. The update frequency of laser range finder is about 5 to 15 Hz and the update frequency of the odometry is about 50 Hz. Obviously, odometry has a higher update frequency. In this paper, to simplify the condition, the measurement from laser range finder and odometry are updated together when new point cloud received. Then, the latest measurement from odometry and the new point cloud will be used to update the joint states together. Here is the algorithm.

```
function
    if has new point cloud
        $(\hat{x}, P)$ ← RemoveOldTrack $(\hat{x}, P)$
        $(\hat{x}, P)$ ← MakePrediction $(\hat{x}, P)$
        $(\hat{x}, P)$ ← DataAssociation $(\hat{x}, P)$
        $(\hat{x}, P)$ ← TracksUpdate $(\hat{x}, P)$
    end
end
```

Table 2. Tracking of objects using joint kinetic states

Table 2 is the procedure carried out at each iteration when a new measurement received. $\hat{x}$ is a vector, which is the mean of the estimation of the joint kinetic states and P is the covariance of the error $(\hat{x} - x)$. First, we remove the outdated tracks if these tracks are predicted to be occluded or out of the detection range of laser range finder. Second, for the existing tracks, according to their kinetic states, we make a forward prediction for all existing tracks. Third, we use a new method in data association, which helps to associate the existing tracks to new measurement. Last, with the assignment result in data association, we update all of the existing tracks and finish the iteration.

3.4.1 Sensor Kinetic States Prediction

In this part, we will build up the kinetic model for the sensor. In the simplified motion model of the sensor, we build up the following kinetic matrix. The laser range finder rotates around the rotation axis of the moving platform, so, the kinetic states of the sensor is same to that of the moving platform. The obstacle tracking system works in a synchronous manner with a constant sampling time T. x_s represents the kinetic states of the laser range finder, and z_s represents the observed kinetic states from odometry and IMU. In x_s, $[x, y]$ is the position of the laser range finder, and $[\theta]$ is the rotation angle of the laser range finder.

$$x_s = \left[x\ \dot{x}\ y\ \dot{y}\ \theta\ \dot{\theta}\right]^T,\ \ z_s = [\dot{x}\ \dot{y}\ \dot{\theta}]^T \tag{3.1}$$

Here is the kinetic model of the sensor.

$$x_s(k+1) = F(k)x_s(k) + G(k)u(k) + \xi(k), \xi(k) \sim \aleph(0, Q(k)) \quad (3.2)$$

$$z_s(k) = H(k)x_s(k) + \varepsilon(k),\ \varepsilon(k) \sim \aleph(0, R(k)) \quad (3.3)$$

$Q(k)$ is mainly composed of the acceleration of the sensor, and $R(k)$ is mainly composed of the sensor error of odometry. Generally, they are two constant value generated from experience.

$$F(k) = \begin{bmatrix} J & 0 & 0 \\ 0 & J & 0 \\ 0 & 0 & J \end{bmatrix}, \text{where } J = \begin{bmatrix} 1 & T \\ 0 & 1 \end{bmatrix}$$

$u(k)$ is the control input of the system. Here we also use the constant velocity model for the moving platform, or for the sensor, so, $u(k)$ is in this form:

$$u(k) = [0\ 0\ 0\ 0\ 0\ 0]^T$$

$G(k)$ doesn't make sense when $u(k)$ is a zero vector.

$$H(k) = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

It is important to note that the kinetic model here is constant velocity model. The acceleration of the sensor is viewed as noise in the model, because the update frequency is high and the influence from acceleration can be ignored. What 's more, the assumption here is that the noise follows normal distribution, which is one of the prerequisites of Kalman Filter.

In this circumstance, where $u(k) = [0\ 0\ 0\ 0\ 0\ 0]^T$, the estimation of the kinetic states of the sensor is as follows.

$$x_s(k+1|k) = F(k)x_s(k|k) \tag{3.4}$$

$$P_s(k+1|k) = F(k)P_s(k|k)F^T(k) + Q(k) \tag{3.5}$$

3.4.2 Kinetic States Prediction of Objects

Similar to the kinetic model in 3.3.1, we also use a constant velocity model to track the moving objects. Different from the kinetic model of laser range finder, we can only observe the position of the boundary points of the moving objects. We can't observe the velocity of moving object directly, and we can't even get the precise position of the moving objects under the influence of self-occlusion. What's more, when object is occluded by other objects, the error in estimation of kinetic states will become even larger.

$$x_t = [x\ \dot{x}\ y\ \dot{y}]^T, \quad z_t = [x\ y]^T \tag{3.6}$$

And the observation system is in this form,

$$x_t(k+1) = F(k)x_t(k) + \xi(k),\ \xi(k) \sim \aleph(0, Q(k)) \tag{3.7}$$

$$z_t(k) = H(k)x_t(k) + \varepsilon(k), \varepsilon(k) \sim \aleph(0, R(k)) \tag{3.8}$$

$$\text{where } F(k) = \begin{bmatrix} J & 0 \\ 0 & J \end{bmatrix}, \text{and } J = \begin{bmatrix} 1 & T \\ 0 & 1 \end{bmatrix},$$

$$and\ H(k) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}.$$

In this part, we also assume that the error follows normal distribution. However, different from the kinetic states of sensor model, the variance of the noise in this model is a key point to data association. It is hard to find an appropriate value for variance. Large variance may lead to false matching and small variance will lead to rejection in matching. Many works find the proper variance by setting the maximum velocity of the moving objects in the environment. Obviously, this is not a good solution to general case, where the velocity of the moving objects has a large variation range. The prediction of the kinetic states of the vehicles is:

$$x_t(k+1|k) = F(k)x_t(k|k) \tag{3.9}$$

The prediction of the covariance of $(x_t - \widehat{x_t})$ is

$$P_t(k+1|k) = F(k)P_t(k|k)F^T(k) + Q(k) \tag{3.10}$$

4. Traditional MOT Method

As is introduced in 1.2, MOT can be divided into six classes. In this chapter, we will introduce the traditional MOT method which is based on global nearest neighbor search in data association.[44]

4.1 Data Association Based on Global Nearest Neighbor (GNN) Search

One important problem in moving object tracking is data association. Most data association method require the measure of probability in order to evaluate alternative hypotheses. The basic GNN approach attempts to find and to propagate the single most likely hypothesis at each scan.

The track updating process generally begins with a procedure that is used to choose the best observation to track association. This problem is known as data correlation and is composed of two stages which are coarse level data association and precise level data association.

Traditional MOT based on GNN can be divided into two stages. The first is coarse level data association and the second part is the precise level data association.

4.1.1 Coarse Level Data Association

Gating is a coarse test for eliminating unlikely observation-to-track pairing and help to reduce the computation. One gate is generated when we predict the position of one track. All measurements which fall in the gating area are considered as the candidates of the pairing object. And the gating process is achieved by using the Kalman Filter.

Using the kinetic states represented in 3.4.2 of the tracks, at laser scan k-1, the Kalman Filter evaluates the prediction $x_t(k+1|k)$ of each track with the observation $z_t(k) = H(k)x_t(k) + \varepsilon(k)$, $\varepsilon(k) \sim \aleph(0, R(k))$. The measurement pre-fit residual between the observed states and the prediction states is

$$\widetilde{y}_t(k+1) = z_t(k+1) - H(k)x_t(k+1|k) \tag{4.1}$$

Next, by using the residual covariance matrix,

$$S(k+1) = H(k+1)P(k+1|k)H(k+1)^T + Q(k+1) \tag{4.2}$$

Then the normalized residual matrix d^2 is,

$$d(k+1)^2 = \widetilde{y}_t(k+1)^T S(k+1)^{-1} \widetilde{y}_t(k+1) \tag{4.3}$$

Recall that the Mahalanobis distance of a vector $\vec{x} = (x_1, x_2, x_3, \dots, x_N)^T$ from a set of observations with mean $\vec{\mu} = (\mu_1, \mu_2, \mu_3, \dots, \mu_N)^T$ and use S as covariance matrix. Mahalanobis distance is

$$D_M(\vec{x}) = \sqrt{(\vec{x} - \vec{\mu})^T S^{-1} (\vec{x} - \vec{\mu})} \tag{4.4}$$

Return to the gating process, we define a threshold constant for gate G such that correlation is allowed if the following relationship is satisfied by the norm d^2

of the residual vector

$$d(k+1)^2 = \tilde{y}_t(k+1)^T S(k+1)^{-1} \tilde{y}_t(k+1) < G \qquad (4.5)$$

The value of d^2 is the sum of N independent Gaussian random variables with zeros mean and unit standard deviation and d^2 follows χ_N^2 distribution with N degree of freedom and naturally, we can use cumulative distribution function to represent the probability that an observation fall into the prediction gate. On two dimensional spaces, the validation gate is an ellipse, it is usually called as validation gate. G is a predefined value to determine the probability that a new observation falls into the validation gate of the previous prediction.

There is a trade off in setting the value of G. If G is set too large, we will fail to reject false matching in gating process, however, if G is set too small, we may reject the true matching pairs.

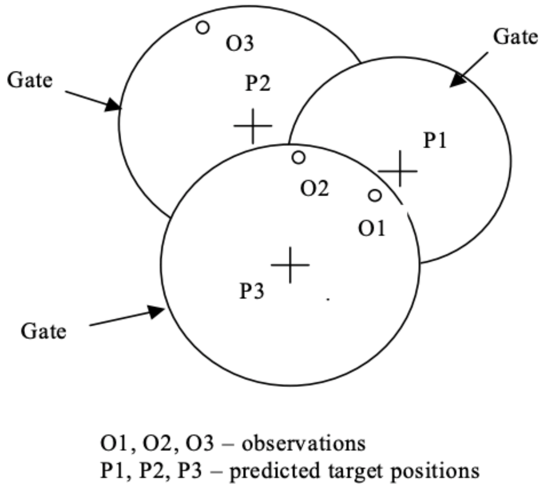

O3
Gate
Gate
P2
P1
O2
O1
P3
Gate
O1, O2, O3 – observations
P1, P2, P3 – predicted target positions

Figure 5. Interpretation of Gating Process

In figure 5, for points P3, which is the predicted position of an existing track. Only observation O1 and O2 in its validation gate. We will only consider match

P3 with O1 and O2 in Precise level data association.

4.1.2 Precise Level Data Association

In 4.1.1 we know, only the observation falls in within the gate of the predicted positions will be considered to update the tracks. This is the coarse level in data association.

After the process in 4.3.1, we use global nearest neighbor search method to finish data association. In the literature, current part is called assignment problem. Target is to assign each observation to the existing tracks. And the optimal assignment minimizes a total distance function which is the sum of the distances for all the individual assignments. Thus, the first step is to define a distance measurement from the predicted positions of the track i to observation j.

A general method used in this work is Hungarian method. Let N_{k-1} and N_k represent the number of existing obstacles and new measurement, respectively. And we define $i \in \{1, \dots, N_{k-1}\}$ and $j \in \{1, \dots, N_k\}$, c_{ij} is the cost between obstacle i at sequence k-1 and obstacle j at sequence k. Where $d_{ij} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}$, and $(x_i, y_i), (x_j, y_j)$ are the center of the obstacle i and obstacle j respectively.

Next, we construct a cost matrix $C \in \mathbb{R}^{N_{k-1} * N_k}$ for assignment problem.

$$C = \begin{bmatrix} d_{11} & \cdots & d_{1N_{k-1}} \\ \vdots & \ddots & \vdots \\ d_{N_{k-1}1} & \cdots & d_{N_k N_{k-1}} \end{bmatrix}$$

To represent the result in gating process, we set

The desired solution of the assignment matrix is the one that minimizes the summed total distance. When the correspondence of the existed obstacles and new measurements is built, we update the Kalman Filter.

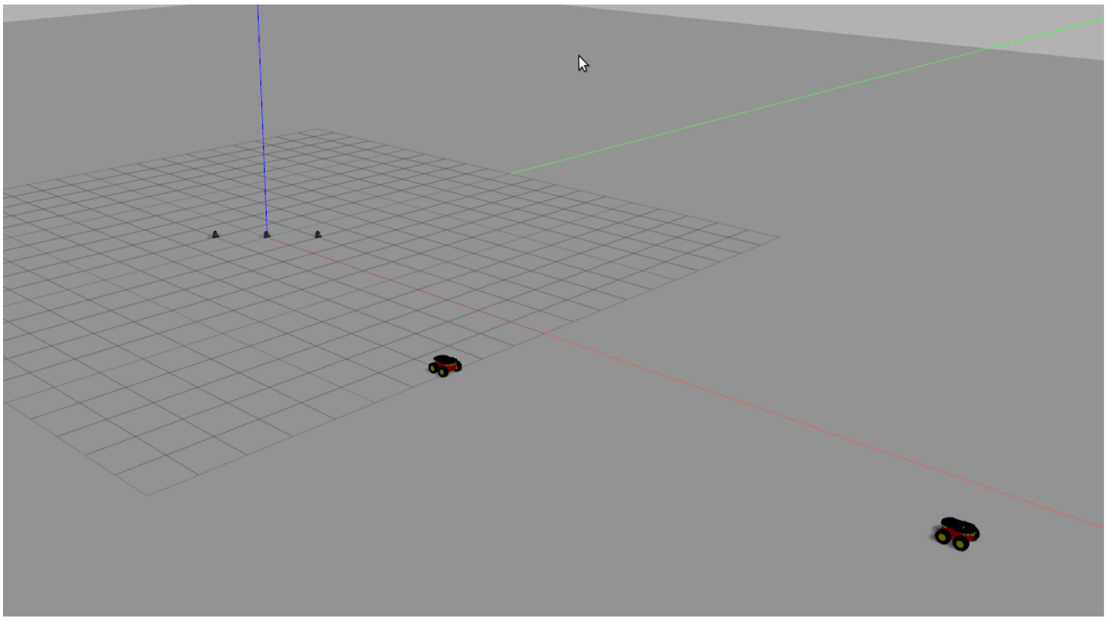
4.2 Preliminary test based on GNN

The setup used for conducted experiments consisted of one Laser Range Finder, LDS-01. The laser range finder is mounted on the top of the robot, Turtlebot3. In simulation, we change the laser range of this sensor to 10 meters and angular resolution to $0.25°$ with $360°$ angular range. We also set the accuracy as $\pm 30\,mm$. The sensor also provides the scan frequency at 10Hz. This modified sensor can be substituted by two back to back mounted Hokuyo UTM-10LX laser range finder, which is a 2D LiDAR.

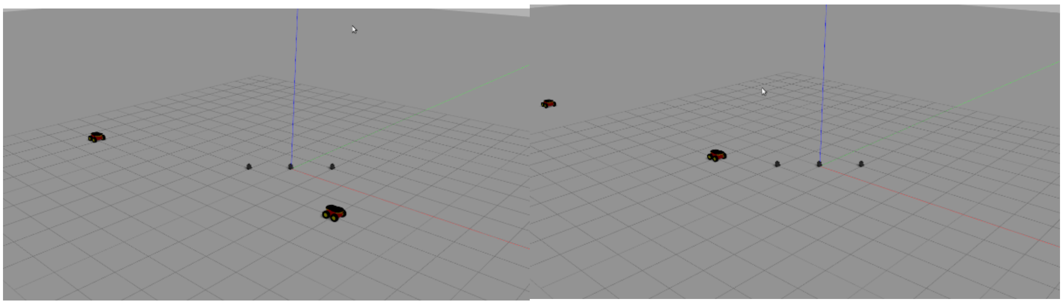
In the simulation environment, we let two vehicle-like robots move in the environment, and we let two cylinder-like robots stay in their position. The sensor was mounted on the top of another robot, which was located at the origin point. In Fig. 6a, one grid represents two-by-two square meters.

During motion, the platform is tracked based on its IMU and odometry sensor. The obstacle in simulation environment are two vehicle-like Pioneer-3AT robots and two cylinder-like Turtlebot3 robots, these robots are also equipped with the IMU and odometry sensor. We obtain the true kinetic states from the sensor on obstacle and then we compare the estimated kinetic states to the true kinetic states to validate the algorithm. The algorithm is implemented in ROS (version Kinetic)[39] and MATLAB ROS Toolbox. And the simulation environment is based on Gazebo[40] simulation software.

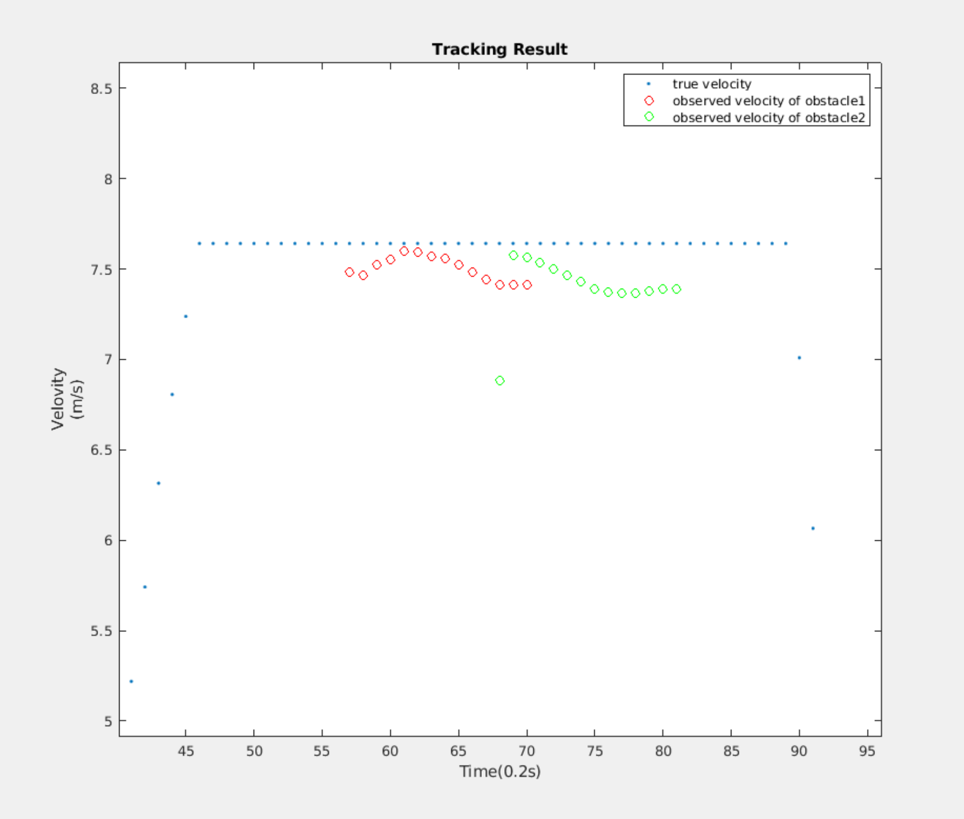
The laser range finder mounted on the top of the robot should be able to track multiple objects in the environment correctly. This test was carried out in a simulated environment with four moving objects.



(a) Initial Pose



(b) Two objects move in the environment



(c) Tracking results

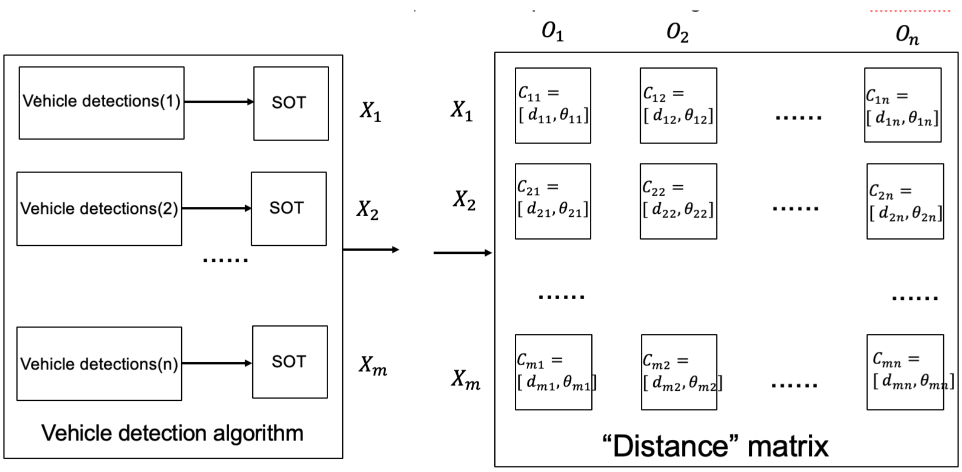
Figure 6. Preliminary test based on GNN.

These two moving objects share the same control commands and have same velocity. We get the true velocity from the odometry on the moving robots. In figure 6, the observed velocity is the tracking result of the algorithm. The blue points are the true velocity of the two moving robots. The red point and the green points are the observed velocity of the moving robots. The x-axis represents the time stamp and the y-axis represents the velocity value. As is shown in this figure 6, the predicted velocity of the moving objects is tracked by the GNN algorithm. And the error level is about no more than 5%.

5. Deep Learning-based MOT Method

5.1 Overview of the Deep Learning-based MOT Method

In figure 7, we introduce the distance matrix first. From the vehicle detection system, we will get the vehicles properties sequence by sequence. From the detection, we can deduct the kinetic state of the vehicles, which can be represented by X_1 to X_m, suggested we have m tracked vehicles. Suppose we have n newly detected vehicles, which are represented by O_1 to O_m. Then we build up a "distance matrix" based on the distance and pose difference between each tracked vehicles and new detected vehicles. The calculation of distance matrix can be found in 5.3.



O_1
O_2
O_n
Vehicle detections(1)
SOT
Vehicle detections(2)
SOT
......
Vehicle detections(n)
SOT
Vehicle detection algorithm
X_1
X_2
X_m
X_1
X_2
X_m
$C_{11} = [d_{11}, \theta_{11}]$
$C_{12} = [d_{12}, \theta_{12}]$
......
$C_{1n} = [d_{1n}, \theta_{1n}]$
$C_{21} = [d_{21}, \theta_{21}]$
$C_{22} = [d_{22}, \theta_{22}]$
......
$C_{2n} = [d_{2n}, \theta_{2n}]$
......
......
$C_{m1} = [d_{m1}, \theta_{m1}]$
$C_{m2} = [d_{m2}, \theta_{m2}]$
......
$C_{mn} = [d_{mn}, \theta_{mn}]$
"Distance" matrix

Figure 7. "Distance" matrix

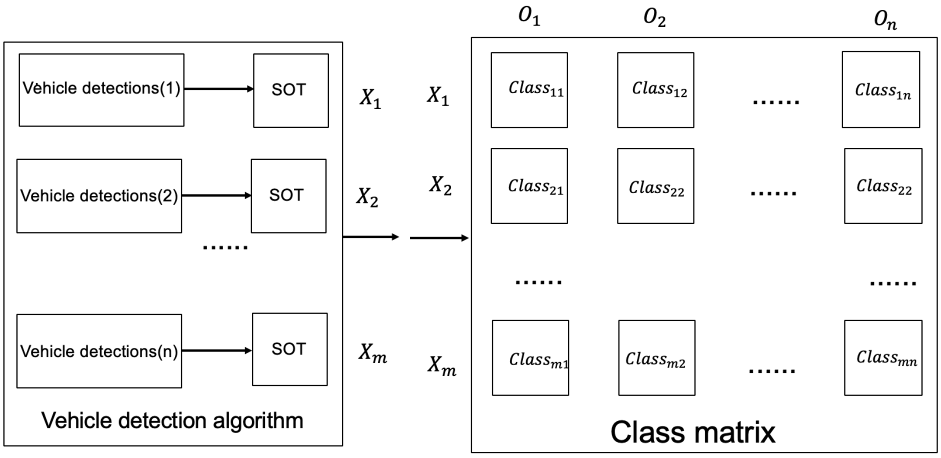


Figure 8. Class matrix

From figure 8, we introduce the class matrix. We take the assignment relationship from the ground truth in dataset. Then we build up cost matrix. We set $Class_{ij}$ =1 if O_j is associated to X_i, and $Class_{ij}$ = 0 if O_j isn't associated to X_i.

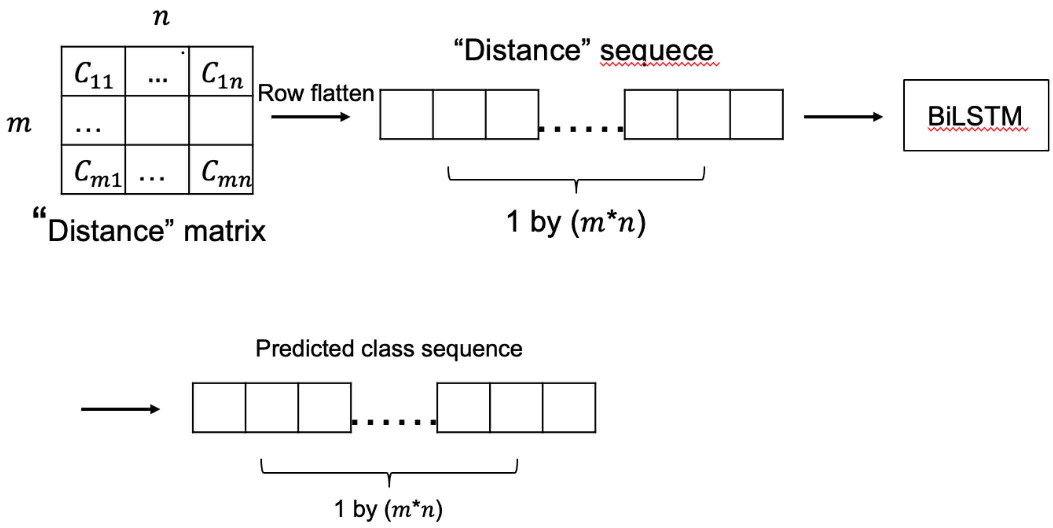


Figure 9. Prediction by BiLSTM in training

In training process of BiLSTM, figure 9, we flatten the distance by row-wise and generate a sequence, we call it as the "distance" sequence. The sequence is fed

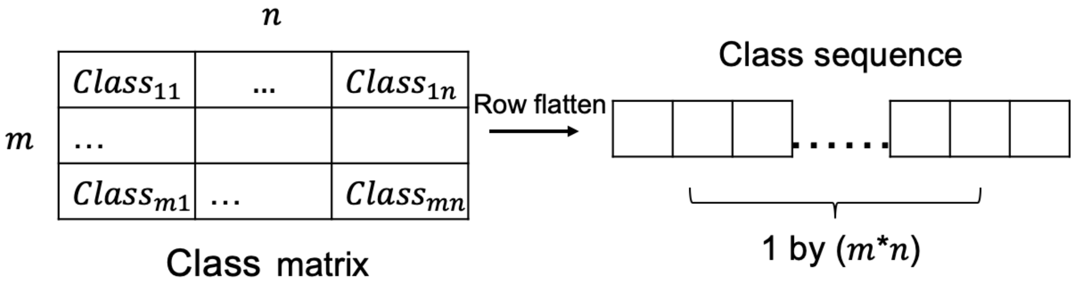


Figure 10. Class sequence

to BiLSTM network and the network will generate a prediction of the assignment relationship. Then, we compare this predicted class sequence to the true class sequence, which is generated by the flattened class matrix in figure 10.

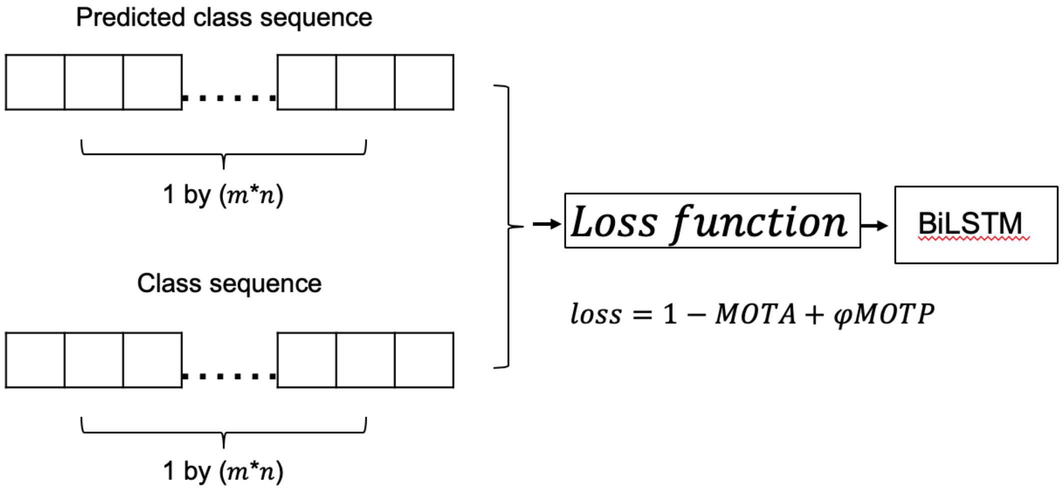


Figure 11. Loss function

In figure 11, we calculate the loss function by comparing class sequence and predicted class sequence. Where $loss$ is calculated by the both of MOTA and MOTP which is generated from the class sequence and predicted class sequence. After doing gradient descent to all parameters in BiLSTM to decrease the loss function, we will get a trained BiLSTM network.

The trained BiLSTM network will generate the predicted class sequence which will be reformed to the new assignment matrix, in figure 12.

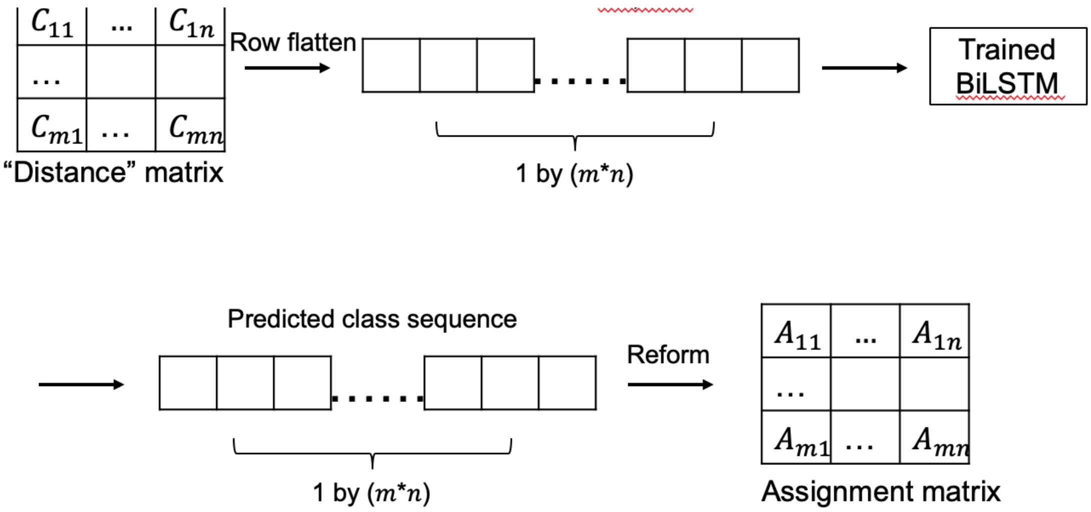


Figure 12. Trained BiLSTM network

5.2 Structure of BiLSTM

The objective of MOT is to predict the trajectories of all vehicles at each time stamps, including their position and pose. The design of the proposed BiLSTM considers the following two points. First, the size of input to the network varies over time and the network should cope with this input. Second, for each element in assignment matrix, should be the global optimal rather than the local optimal assignment. From this point, all elements in the input, the distance matrix, should have influence on each element in assignment matrix. In this way, each element in assignment matrix in influenced by the whole distance matrix. BiLSTM suits for our circumstance well.

5.2.1 Introduction to LSTM

In RNN, a great problem is the vanishing gradients. From input to output of

RNN, the chain is very long in multiplication. If we do derivative to each parameter, according to the chain rules, the gradient is easy to be vanished or exploded. In this case, we can't finish backpropagation and we can update the parameters in RNN.

To solve the problem of exploding of vanishing of the gradient, LSTM[42] helps to preserve the error that can be backpropagated through time and layers. The maintenance of the error helps to maintain the gradient of LSTM over many time steps. LSTM preserves the information of the recurrent network in a gated cell outside the normal transportation flow. The information is stored in the cell and the cell controls the storage of the information via gates in LSTM. Generally, these gates use sigmoid function as activation function. Sigmoid function is easy to be differentiable and suitable for backpropagation.

Those gates' reaction depends on the received signals. The strength of the information decides if it can pass the gates or not.

LSTM can be divided into cell, input gate, output gate and a forget gate. The forward process of the LSTM is:

$$g^{(t)} = \varphi(W^{gx}x^{(t)} + W^{gh}h^{(t-1)} + b_g) \tag{5.1}$$

$$i^{(t)} = \sigma(W^{ix}x^{(t)} + W^{ih}h^{(t-1)} + b_i) \tag{5.2}$$

$$f^{(t)} = \sigma(W^{fx}x^{(t)} + W^{fh}h^{(t-1)} + b_f) \tag{5.3}$$

$$o^{(t)} = \sigma(W^{ox}x^{(t)} + W^{oh}h^{(t-1)} + b_o) \tag{5.4}$$

$$g^{(t)} = \varphi(W^{gx}x^{(t)} + W^{gh}h^{(t-1)} + b_g) \tag{5.5}$$

$$s^{(t)} = g^{(t)} * i^{(t)} + s^{(t-1)} * f^{(t)} \tag{5.6}$$

$$h^{(t)} = s^{(t)} * o^{(t)} \tag{5.7}$$

For the hidden states is forward process is $h^{(t)}$, where $h^{(t-1)}$ is the last hidden state, $x^{(t)}$ is the input vector to LSTM unit, $f^{(t)}$ is the forget gate where σ is the activation function, $i^{(t)}$ is the input gate and $s^{(t)}$ is the cell state vector.

The loss function is

$$l^{(t)} = ||h^{(t)} - y^{(t)}||^2 \tag{5.8}$$

The total loss is

$$\mathrm{L} = \sum_{t=1}^{T} l^{(t)} \tag{5.9}$$

And the loss can be backpropagated by do partial derivative to W.

For general LSTM, it is only forward direction. Alone with it, there are bidirectional recurrent neural network and bidirectional long short-term memory. BILSTM showed better performance.

5.2.2 BiLSTM in proposed method

In this paper, the network structure is as follows. In figure 13, the input to this layer is the row-wise flattened distance matrix, and length of this sequence varies in different sequence. The second and the fourth layer are BiLSTM layer[43]. The output of BiLSTM layer are fully connected layer and softmax layer. These two layers help to classify the output.

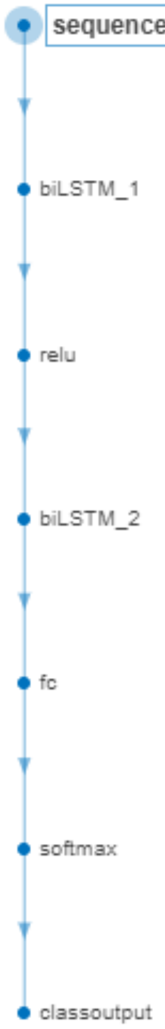


Figure 13. Structure of BiLSTM used in our algorithm

5.3 Assignment criterion

First, we need to define the distance and angles between existing tracks and observations.

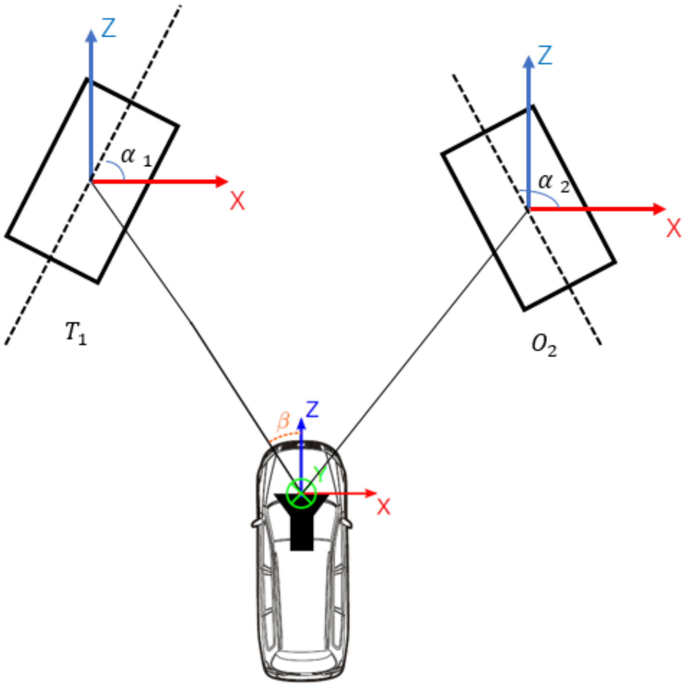


Figure 14. Interpretation of distance matrix and pose matrix

In figure 14, distance between track i and observation j is the Euclidean distance d_{ij}. From the detection algorithm, we will get the pose of existing tracks and detections. And the pose between track i and observation j is $\alpha_{ij} = \min\left(\left|\alpha_i - \alpha_j\right|, (2\pi - |\alpha_i - \alpha_j|)\right)$. Then, we combine d_{ij} and α_{ij} together, where we define $D_{ij} = [\, d_{ij}, \alpha_{ij}]$. In this way, we form a matrix D. It's worth to mention that, this matrix support expansion if we can extract more detailed information of objects from detection algorithm.

From the training data, we will get the matrix D and its associated assignment matrix A, which has the same size with D and is composed on the assignment information, in other words, "assigned" or "not assigned". If observation j is assigned to track i, then the element in A is labeled as "assigned", else, it is labeled as "not assigned". Then we use the data to train the BiLSTM network. The loss function is sum of each element in A-A_{pred}, where A_{pred} is the prediction given by BiLSTM.

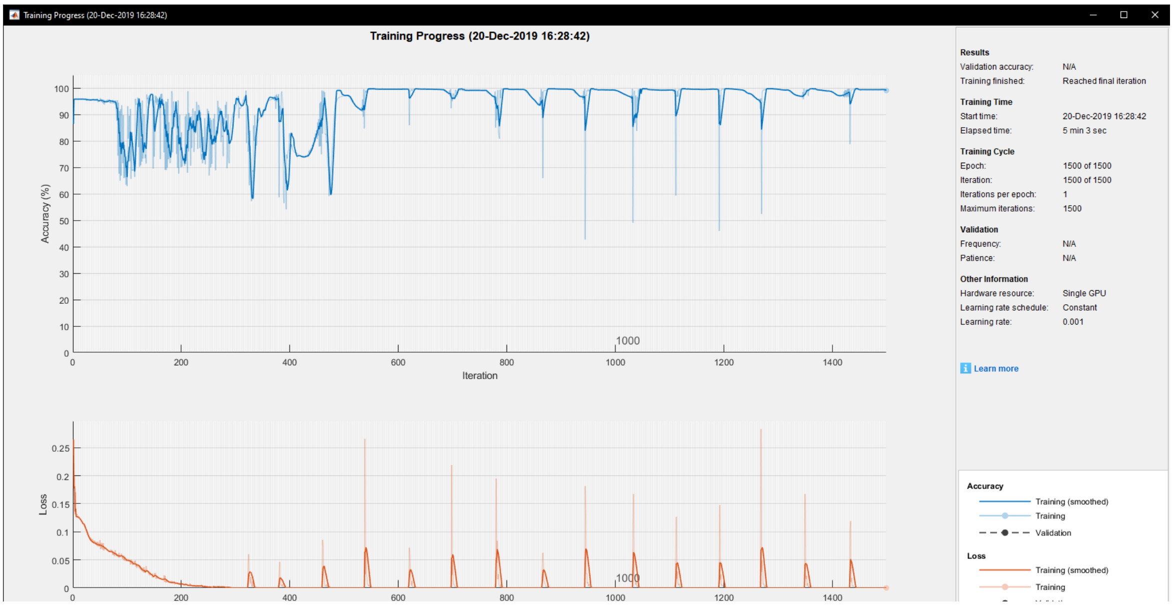
Once the network is trained, the BiLSTM will not change, then we use this net to make the prediction, A_{pred}. The final cost matrix to GNN algorithm is

$$D_{pred} = d - A\mu, \tag{5.10}$$

And μ is a small value. We use this expression to lead in our prediction and strength the association between observation and the existing tracks.

5.4 Training results

The training result is in figure 15.



Results
Validation accuracy: N/A
Training finished: Reached final iteration

Training Time
Start time: 20-Dec-2019 16:28:42
Elapsed time: 5 min 3 sec

Training Cycle
Epoch: 1500 of 1500
Iteration: 1500 of 1500
Iterations per epoch: 1
Maximum iterations: 1500

Validation
Frequency: N/A
Patience: N/A

Other Information
Hardware resource: Single GPU
Learning rate schedule: Constant
Learning rate: 0.001

Figure 15. Training process of BiLSTM

The blue line represents the accuracy of the model, and the orange line represents the loss of the network. Generally, the mean of the accuracy value increase over the iteration and the mean of the loss value decrease over the iteration.

However, we notice that the accuracy and loss fluctuate over time. There are two main reasons. First is the length of each input sequence, with the change of numbers of existing tracks and observations, the size of input sequence may vary significantly. If the size of one sequence is very short, one misprediction will lead to great change in loss and accuracy. The second is the way we define the loss. Restricted to the fact that most of elements in the assignment matrix is 0, and the expected elements with 1 value is same to the number of existing tracks. However, we still use the traditional mean square error methods to calculate the loss.

6. Experiments on new method

6.1 Experiment Environments

As is mentioned in previous part, our experiment is based on KITTI data set. The KITTI data set is a novel real-world computer vision benchmark. Their interests include stereo, optical flow, visual odometry, 3D object detections and 3D tracking. And provides the sensor data from video cameras and Velodyne LiDAR. Their detections are captured by the driving scenarios near Karlsruhe, including the rural areas and urban areas.

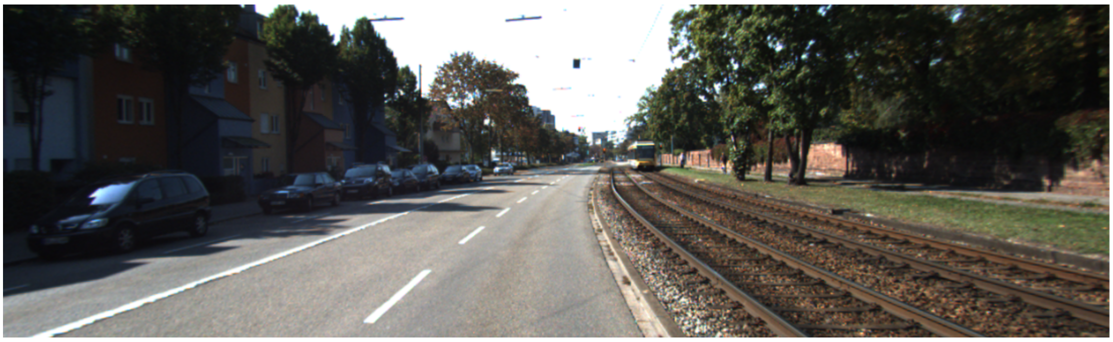


Figure 16. Scenarios in urban area, captured by camera

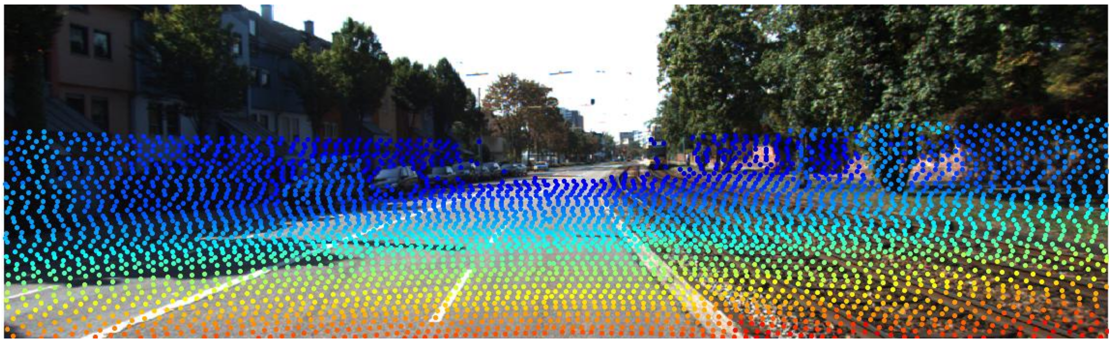


Figure 17. Project the point clouds on synchronized images

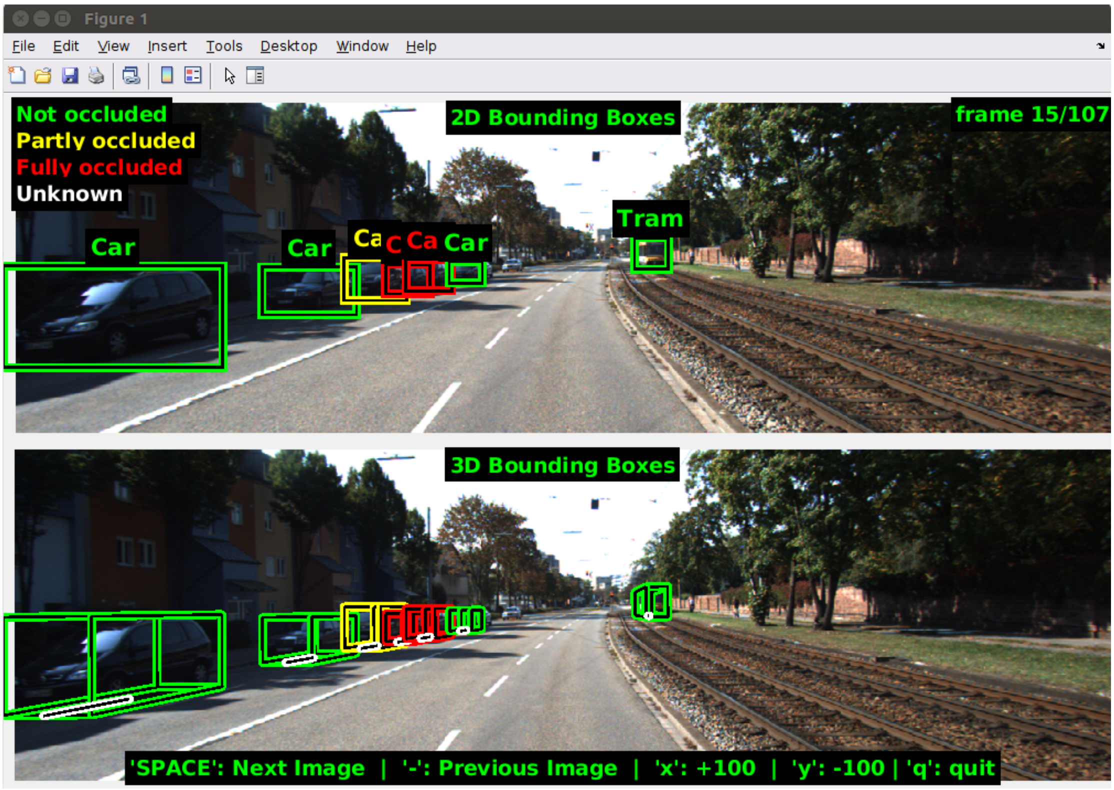


Figure 1
File Edit View Insert Tools Desktop Window Help
Not occluded
Partly occluded
Fully occluded
Unknown
2D Bounding Boxes
frame 15/107
Car
Car
Ca
C
Ca
Car
Tram
3D Bounding Boxes
'SPACE': Next Image | '-': Previous Image | 'x': +100 | 'y': -100 | 'q': quit

Figure 18. Bounding boxes generated by vehicle detection algorithm

Figure 2D and 3D bounding box detection for training

In figure 16, it shows the scenarios in rural environment recorded by the camera. In figure 17, KITTI dataset projects the point clouds on the relative image, which helps to build an intuitive idea about how the point clouds look like and helps to explain the relationship between image and point clouds. It's worth to mention that, the point clouds and images have been synchronized. In figure 18, the upper part is the vehicle detection results based on 2D image and the bottom part is the detected vehicles based on 3D point clouds.

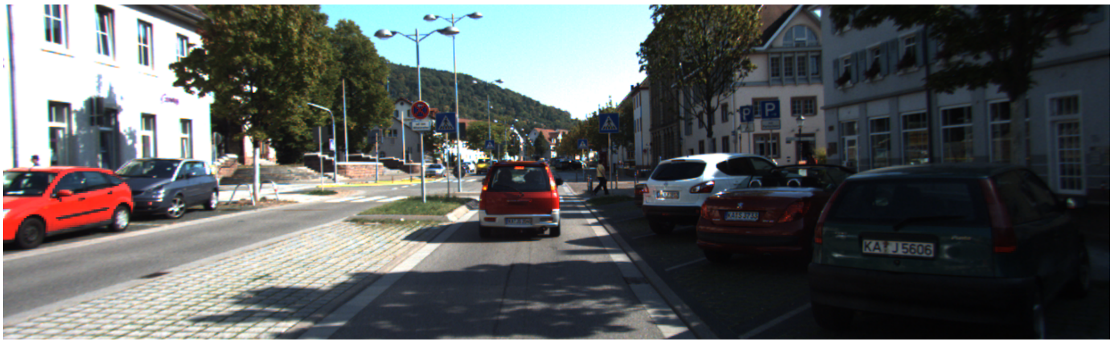


Figure 19. Scenarios in complex urban environment, captured by camera

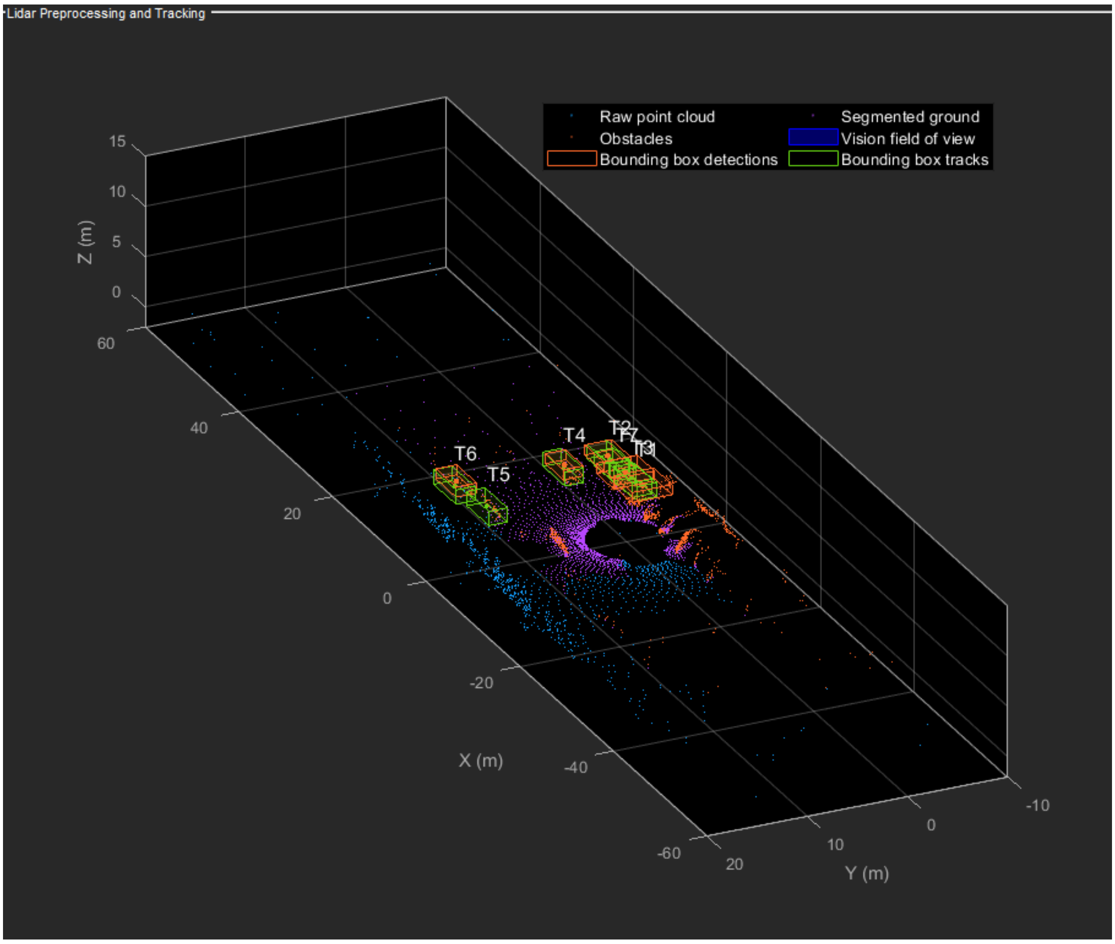


Figure 20. Scenarios in complex urban environment, captured by LIDAR

In figure 19, we plot all the point clouds detected in figure 14 in this 3D scenario. In Figure 20, the purple points represent the ground. The blue points represent the point clouds out of the range. We only use the vehicle detection algorithm to detect the vehicles in the range. The orange points represent the remaining point clouds, where we detect the vehicles from.

The orange bounding boxes are the detected vehicles and the green bounding boxes are the existing tracks. It's worth to mention that, in vehicle detection, we set the detection area as the visible parts in camera, which means, for the invisible parts in camera, the vehicle detection algorithm doesn't work.

6.2 Evaluation criterion

Multiple Object Tracking accuracy and precision (MOTA and MOTP) are two standard and widely-used matrix to assess the quality of multiple object tracker. They are specifically designed to encode the challenges and difficulties of tracking multiple objects. Here are the formulas to compute the MOTA and MOTP:

In MOTA, the matching criterion can vary according to the applications, and we define the criterion threshold as ε. If the distance between two objects is smaller than ε, the track is considered as a true positive,TP_t. Otherwise, it is a false positive (FP_t) and a missed ground-truth is considered as a false negative (FN_t). For a track marked as TP at time t and at the most recent previous time step, if it is assigned to different ground truth identifies, then it is counted as identity switch (IDS_{wt}).

$$\text{MOTA} = 1 - \frac{\sum_t (FP_t + FN_t + IDS_{wt})}{\sum_t (M_t)}$$

MOTP calculates the average distance of all tracks among the TP_t tracks, and their associated ground truth, which is formally defined as:

$$\text{MOTP} = \frac{\sum_{i,t} d_{i,t}}{\sum_t c_t}$$

Let c_t be the number of matches found for time t. For each of theses matches, calculate the distance d_t^i between the object o_t and its corresponding hypothesis.

It is the total position error for matched object hypothesis pairs over all frames, averaged by the total number of matches made. It shows the ability of the tracker to estimate precise object positions, independent of its skill at recognizing object configurations, keeping consistent trajectories, etc.

6.3 Detection results

According to the reported detection results of PIXOR, the average precision (AP) is validation set is 60.7% and the precision of detection is 75.3%. We chose PIXOR as our detection algorithm because its fast calculation speed, which supports simultaneously detection with the newly detected data.

6.4 Results and Analysis

The evaluation of the proposed algorithm is based on two evaluation parameters. As is introduced in 3.6, MOTA and MOTP are used in the evaluation.

Due to the fact the from the detection algorithm can only detect the vehicles in the range of the camera, the detection result can’t directly compare to that of the benchmark. What ‘s more, this algorithm means to find a new strategy to solve the assignment problem in data association. In classical solutions to the data association in MOT, GNN is our comparison target. Because both our algorithm and the GNN, they all need the distance matrix. However, our algorithm is capable to make full use of the detection information, especially the pose of the targets. Once we have better sensors and better detection algorithms, which provide more details about moving targets, we can lead these new features into our algorithm. The more features detected by the detection algorithm, the better data association result is expected to be.

To test our algorithm, we select 2 different scenarios with different difficulty in MOT in KITTI dataset. They are 0014, and 0059 dataset.

Dataset 0059	GNN		Our algorithm	
vocScore = 0.05	rateFP	7%	rateFP	3.9%
	rateTP	62.1%	rateTP	60.0%
	rateFN	35.4%	rateFN	36.1%
	MOTP	0.430	MOTP	0.628
	MOTA	0.549	MOTA	0.561
vocScore = 0.1	rateFP	3.65%	rateFP	1.647%
	rateTP	64.2%	rateTP	61.4%
	rateFN	31.9%	rateFN	34.1%
	MOTP	0.416	MOTP	0.631
	MOTA	0.605	MOTA	0.598
vocScore = 0.2	rateFP	8.1%	rateFP	5.87%
	rateTP	61.4%	rateTP	61.4%
	rateFN	36.4%	rateFN	34.1%

	MOTP	0.434	MOTP	0.643
	MOTA	0.531	MOTA	0.526

Table 3. Comparison between our algorithm and GNN in dataset 0059

vocScore determines if we can associate an object to an existing track. If the distance between one detection and an existing track is smaller than the vocScore, we believe the detection and an existing tracking belong to same object.

Dataset 0014	GNN		Our algorithm	
vocScore = 0.05	rateFP	16.2%	rateFP	11.9%
	rateTP	90.4%	rateTP	100%
	rateFN	0%	rateFN	0%
	MOTP	0.730	MOTP	0.728
	MOTA	0.849	MOTA	0.861
vocScore = 0.1	rateFP	16.2%	rateFP	16.7%
	rateTP	90.4%	rateTP	95.2%
	rateFN	0%	rateFN	4.76%
	MOTP	0.716	MOTP	0.831
	MOTA	0.805	MOTA	0.898
vocScore = 0.2	rateFP	20.0%	rateFP	16.7%
	rateTP	90.4%	rateTP	95.2%
	rateFN	4.8%	rateFN	4.%

	MOTP	0.734	MOTP	0.843
	MOTA	0.831	MOTA	0.826

Table 4. Comparison between our algorithm and GNN in dataset 0014

In dataset 0059, a complex autonomous driving scenario, MOTA is low for both algorithms, they are nearly on the same level. However, our algorithm performed much better on MOTP than GNN.

In dataset 0014, a simple autonomous driving scenario, both of the algorithms performed good on MOTA and MOTP criterion. Our algorithm performed better than GNN on MOTA and MOTP.

7. Conclusions

The objective of this thesis is to build up a framework to detect and track the moving objects in autonomous driving scenarios. This framework contains two parts, which are detection and MOT. We proposed a new data association method in MOT and our framework performed better than the traditional MOT method.

In details, once receive the detection results from the detection algorithm, the main difference between our algorithm and traditional MOT algorithm is the way to associate the tracks and new detections. Traditional MOT method associates the existing tracks and new detections based on the distance between each detection and each existing track. The assumption in traditional MOT is each object in detection can be view as a point. They track these points based on their kinetic states. However, traditional MOT will certainly lose the features of exterior of the objects, which will decrease the match precision. In our algorithm, which belongs to deep learning-based MOT, we associate the data based on a pre-trained model. The model mainly uses BiLSTM to solve the data association in MOT problem. In training process, we combine two kinds of features in detection algorithm. The first feature is same to that in GNN algorithm, it is the distance between the detections and the existing tracks. The second is the pose features, which contains the difference in directions between the tracks and the detections. The two features are used in training process, the output is the predicted assignment matrix. By comparing the predicted assignment matrix with true assignment matrix, we define the loss function. Then, we use the trained network to process the data association problem.

The adventure of our method is our method enables to use more detected feature of moving objects in data association rather than only use the distance information as the solution to association. Our algorithm is much more general to cope with all kinds of sensors and their relative detection algorithms. And our algorithm shows better performance based than traditional MOT method.

The disadvantage is we didn’t use the MOTA and MOTP criterion in loss function, which may lead to kind of diverse between our trained model and the optimal model.

References

[1] Betz J. et al. (2019) What can we learn from autonomous level-5 motorsport?. In: Pfeffer P. (eds) 9th International Munich Chassis Symposium 2018. Proceedings. Springer Vieweg, Wiesbaden

[2] Pendleton, S.D.; Andersen, H.; Du, X.; Shen, X.; Meghjani, M.; Eng, Y.H.; Rus, D.; Ang, M.H. Perception, Planning, Control, and Coordination for Autonomous Vehicles. Machines 2017, 5, 6.

[3] Claudine Badue, Rânik Guidolini, Raphael Vivacqua Carneiro, Pedro Azevedo, Vinicius Brito Cardoso, Avelino Forechi, Luan Jesus, Rodrigo Berriel, Thiago Paixão, Filipe Mutz, Lucas Veronese, Thiago Oliveira-Santos, Alberto Ferreira De Souza, arXiv:1901.04407

[4] M. S. Darms, P. E. Rybski, C. Baker and C. Urmson, "Obstacle Detection and Tracking for the Urban Challenge," in IEEE Transactions on Intelligent Transportation Systems, vol. 10, no. 3, pp. 475-485, Sept. 2009.

[5] Kim, B.; Kim, D.; Park, S.; Jung, Y.; Yi, K. Automated Complex Urban Driving based on Enhanced Environment Representation with GPS/map, Radar, LiDAR and Vision. IFAC-PapersOnLine 2016, 49, 190–195.

[6] Zhang, L., Li, Q., Li, M., Mao, Q., & N¨uchter, A. (2013). Multiple vehicle-like target tracking based on the velodyne lidar. IFAC Proceedings Volumes, 46,126–131.

[7] Urmson, C., Anhalt, J., Bagnell, D., Baker, C., Bittner, R., Clark, M., Dolan, J., Duggins, D., Galatali, T., Geyer, C. et al. (2008). Autonomous driving in urban environments: Boss and the urban challenge. Journal of Field Robotics, 25, 425–466.

[8] Montemerlo, M., Becker, J., Bhat, S., Dahlkamp, H., Dolgov, D., Ettinger, S., Haehnel, D., Hilden, T., Ho_mann, G., Huhnke, B. et al. (2008). Junior: The stanford entry in the urban challenge. Journal of Field Robotics, 25, 569–597.

[9] Gregor, R., Lutzeler, M., Pellkofer, M., Siedersberger, K.-H., & Dickmanns, E. D. (2002). Ems-vision: A perceptual system for autonomous vehicles. IEEE Transactions on Intelligent Transportation Systems, 3, 48–59.

[10] Darms, M. S., Rybski, P. E., Baker, C., & Urmson, C. (2009). Obstacle detection and tracking for the urban challenge. IEEE Transactions on intelligent transportation systems, 10, 475–485

[11] Multi-target multi-object tracking, sensor fusion of radar and infrared

[12] Single-camera and inter-camera vehicle tracking and 3D speed estimation based on fusion of visual and semantic features

[13] A lidar Perception Scheme for Intelligent Vehicle Navigation

[14] Luo, W.; Xing, J.; Milan, A.; Zhang, X.; Liu, W.; Zhao, X.; Kim, T.-K. Multiple object tracking: A literature review. arXiv, 2014; arXiv:1409.7618.

[15] A Multi-Sensor Fusion System for Moving Object Detection and Tracking in Urban Driving Environments

[16] Problem Analysis of Multiple Object Tracking System: A Critical Review

[17] PIXOR: Real-time 3D Object Detection from Point Clouds

[18] M. Engelcke, D. Rao, D. Z. Wang, C. H. Tong, and I. Posner. Vote3deep: Fast object detection in 3d point clouds using efficient convolutional neural networks. In Robotics and Automation (ICRA), 2017 IEEE International Conference on, pages 1355–1361. IEEE, 2017.

[19] B. Li, T. Zhang, and T. Xia. Vehicle detection from 3d lidar us- ing fully convolutional network. In Robotics: Science and Systems, 2016.

[20] D. Z. Wang and I. Posner. Voting for voting in online point cloud object detection. In Robotics: Science and Systems, 2015.

[21] X. Chen, H. Ma, J. Wan, B. Li, and T. Xia. Multi-view 3d object detection network for autonomous driving. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017.

[22] Yoon, J.H.; Yang, M.-H.; Lim, J.; Yoon, K.-J. Bayesian multi-object tracking using motion context from multipleobjects. In Proceedings of the 2015 IEEE International Conference on Applications of Computer Vision (WACV), Big Island, HI, USA, 6–9 January 2015; pp. 33–40.

[23] Bar-Shalom, Yaakov; Daum, Fred; Huang, Jim (December 2009). "The probabilistic data association filter". IEEE Control Systems Magazine. 29 (6): 82–100. doi:10.1109/MCS.2009.934469.

[24] H.W.Kuhn. The Hungarian method for the assignment problem. Naval research logistics quarterly, 2(1-2):83–97, 1955.

[25] A. Sadeghian, A. Alahi, and S. Savarese. Tracking the un- trackable: Learning to track multiple cues with long-term dependencies. In Proceedings of the IEEE International Con- ference on Computer Vision, pages 300–311, 2017.

[26] J.Zhu,H.Yang,N.Liu,M.Kim,W.Zhang,and M.-H.Yang. Online multi-object tracking with dual matching attention networks. In Proceedings of the European Conference on Computer Vision (ECCV), pages 366–382, 2018.

[27] Yihong Xu, Yutong Ban, Xavier Alameda-Pineda, Radu Horaud. DeepMOT: A Differentiable Framework for Training Multiple Object Trackers. arxiv:1906.06618v1

[28] Andreas Geiger and Philip Lenz and Raquel Urtasun, Are we ready for Autonomous Driving? The KITTI Vision Benchmark Suite, Conference on Computer Vision and Pattern Recognition (CVPR), 2012

[29] Andreas Geiger and Philip Lenz and Christoph Stiller and Raquel Urtasun, Vision meets Robotics: The KITTI Dataset, International Journal of Robotics Research (IJRR),2013

[30] A K Jain and M N Murty and P. J. Flynn(1999). Data Clustering: A Review

[31] Dominic Zeng Wang, Ingmar Posner and Paul Newman. Model-free detection and tracking of dynamic objects with 2D lidar. The International Journal of Robotics Research 2015, Vol. 34(7) 1039–1063

[32] TOBIAS NYSTRÖM JOHANSSON (2017) LiDAR Clustering and Shape Extraction for Automotive Applications. Master's thesis in CHALMERS UNIVERSITY OF TECHNOLOGY.

[33] Martin Ester, Hans-Peter Kriegel, Jörg Sander, Xiaowei Xu. A density-based algorithm for discovering clusters in large spatial databases with noise (1996). AAAI Press,pp. 226—231

[34] Leonard J, How J, Teller S, et al. (2008) A perception-driven autonomous urban vehicle. Journal of Field Robotics 25(10):727–774.

[35] Mertz C, Navarro-Serment LE, MacLachlan R, et al. (2013) Moving object detection with laser scanners. Journal of Field Robotics 30(1): 17–43.

[36] Arras K, Grzonka S, Luber M and Burgard W (2008) Efficient people tracking in laser range data using a multi-hypothesis leg-tracker with adaptive occlusion probabilities. In: IEEE international conference on robotics and automation, 2008 (ICRA 2008), pp. 1710–1715.

[37] Rusinkiewicz, Szymon; Marc Levoy (2001). Efficient Variants of the ICP Algorithm. Proceedings Third International Conference on 3-D Digital Imaging and Modeling. Quebec City, Quebec, Canada. pp. 145–152.

[38] Arras K, Mozos O and Burgard W (2007) Using boosted features for the detection of people in 2D range data. In: 2007 IEEE international conference on robotics and automation, pp.3402–3407.

[39] Morgan Quigley and Brian Gerkey and Ken Conley and Josh Faust and Tully Foote and Jeremy Leibs and Eric Berger and Rob Wheeler and Andrew Ng, ROS: an open-source Robot Operating System", "Proc. of the IEEE Intl. Conf. on Robotics and Automation (ICRA), Workshop on Open Source Robotics",2009,

[40] N. Koenig and A. Howard, "Design and use paradigms for Gazebo, an open-source multi-robot simulator," 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (IEEE Cat. No.04CH37566), Sendai, 2004, pp. 2149-2154 vol.3.

[41] Graves, A.; Liwicki, M.; Fernandez, S.; Bertolami, R.; Bunke, H.; Schmidhuber, J. (2009). "A Novel Connectionist System for Improved Unconstrained Handwriting Recognition" (PDF). IEEE Transactions on Pattern Analysis and Machine Intelligence. 31 (5): 855–868.

[42] Sak, Hasim; Senior, Andrew; Beaufays, Francoise (2014). "Long Short-Term Memory recurrent neural network architectures for large scale acoustic modeling"

[43] Schuster, Mike, and Kuldip K. Paliwal. "Bidirectional recurrent neural networks." Signal Processing, IEEE Transactions on 45.11 (1997): 2673-2681.2. Awni Hannun, Carl Case, Jared Casper, Bryan Catanzaro, Greg Diamos, Erich Elsen, Ryan

[44] Bar-Shalom Y., and Fortmann T., Tracking and Data Association, Academic Press, 1988.