Microwave Sensing, Signals and Systems
Mekelweg 4,
2628 CD Delft
The Netherlands
https://radar.tudelft.nl/

MS3-2025

M.Sc. Thesis

Combined ego-motion estimation and multiple extended object tracking with automotive radar

Taoyue Wang

Abstract

Automotive radar can provide robust measurements that are crucial for autonomous driving localization and perception tasks, especially under adverse weather, low-light, and long-range conditions. This thesis proposes two radar-only, combined estimation methods that both provide ego-motion information and track multiple extended objects. The key to achieve this is addressing the challenge of distinguishing static and moving targets, using either radar raw signals or radar point clouds.

The raw signal based method is validated in its theoretical feasibility through simulations, while the point cloud based method is shown to improve ego-motion estimation in practical dynamic driving scenarios. With simulation of such scenarios generated using a dedicated MATLAB tool, the proposed approach outperforms the RANSAC-based baseline by reducing the APE metric by 2.00 m/s and the RTE metric by 1.58 m, and achieves accurate position and size estimates for tracked objects. Further validations on the experimental *RadarScenes* dataset show average APE reductions of 25.9% over entire scenes and 56.9% in critical 10-second segments across 7 dynamic traffic scenes. These results demonstrate the effectiveness of radar-only combined estimation of ego-motion and multiple object tracks, for robust localization and perception in complex traffic scenarios.

Faculty of Electrical Engineering, Mathematics and Computer Science
Delft University of Technology

Combined ego-motion estimation and multiple extended object tracking with automotive radar

Thesis

submitted in partial fulfillment of the
requirements for the degree of

Master of Science

in

Electrical Engineering

by

Taoyue Wang
born in Jilin, China

This work was performed in:

Microwave Sensing, Signals and Systems Group
Department of Microelectronics
Faculty of Electrical Engineering, Mathematics and Computer Science
Delft University of Technology

Delft University of Technology

Delft University of Technology
Department of
Microelectronics

The undersigned hereby certify that they have read and recommend to the Faculty of Electrical Engineering, Mathematics and Computer Science for acceptance a thesis entitled **"Combined ego-motion estimation and multiple extended object tracking with automotive radar"** by **Taoyue Wang** in partial fulfillment of the requirements for the degree of **Master of Science**.

Dated: August, 2025

Chairman: _____
dr. F. Fioranelli

Daily supervisor: _____
dr. S. Yuan

Committee Member: _____
dr. R.T. Rajan

Abstract

Automotive radar can provide robust measurements that are crucial for autonomous driving localization and perception tasks, especially under adverse weather, low-light, and long-range conditions. This thesis proposes two radar-only, combined estimation methods that both provide ego-motion information and track multiple extended objects. The key to achieve this is addressing the challenge of distinguishing static and moving targets, using either radar raw signals or radar point clouds.

The raw signal based method is validated in its theoretical feasibility through simulations, while the point cloud based method is shown to improve ego-motion estimation in practical dynamic driving scenarios. With simulation of such scenarios generated using a dedicated MATLAB tool, the proposed approach outperforms the RANSAC-based baseline by reducing the APE metric by 2.00 m/s and the RTE metric by 1.58 m, and achieves accurate position and size estimates for tracked objects. Further validations on the experimental *RadarScenes* dataset show average APE reductions of 25.9% over entire scenes and 56.9% in critical 10-second segments across 7 dynamic traffic scenes. These results demonstrate the effectiveness of radar-only combined estimation of ego-motion and multiple object tracks, for robust localization and perception in complex traffic scenarios.

Acknowledgments

Looking back on these two years, I realize how challenging it has been to reach this point. I am truly grateful for the generous support from all of you, without which this journey would not have been possible. First and foremost, I would like to express my sincere gratitude to my professor, Dr. Francesco Fioranelli, for his invaluable guidance and advice on both my research and life. Thank you for teaching me the methods of academic research and the importance of paying attention to detail. Thank you for always thinking and acting from my perspective. Thank you for reminding me not to push myself harder than the boss. I wish you all the best in the future, and I am confident our paths will cross again.

I would also like to thank my daily supervisor, Dr. Sen Yuan, who is an amazingly brilliant person. The feeling is so good that there is always someone at my back. Thank you for the clear directions you give when I was confused. Thank you for all of these insightful and patient discussions with me. Thank you for your enthusiasm for both work and life. You have been a shining star, and will continue to shine brightly.

Next, I want to express my gratitude to the professors, postdocs, and PhD students in MS3 who have offered valuable advice throughout my journey. Thank you, Professor Olexander, for your outstanding radar courses and for teaching me the true attitude towards research. Thank you, Professor Hans, for your help after my midterm and for all the funny jokes. Thank you, Simin, for your important suggestions and for sharing the RadarScenes data. A special thanks to Dingyang, Tworit, Ignacio, Mujtaba, Apostolos, Kuitong, Peiyu, and Botao for these wonderful interactions.

I am also deeply grateful to my colleagues and friends, Ben, Jiaqi, Junhan, and Yuhan, who made my study abroad experience truly unforgettable. I wish you all bright futures. Thank you, Raj, for being the coordinator of my track and for being part of my thesis committee.

Finally, I would like to give a lot of special thanks to my mother and father, who support me wholeheartedly, especially when things were not going smoothly and when there was no sunshine in Delft. I have grown into a better person because of your love and encouragement. Now we can celebrate this amazing achievement together. I love you forever.

I will cherish all the memories, both the good and bad moments in Delft. Everything I have gained from this journey will stay with me as I move forward into the unpredictable future.

Taoyue Wang
Delft, The Netherlands
August, 2025

Contents

Abstract v

Acknowledgments vii

List of Figures xi

List of Tables xiii

List of Abbreviations xv

I Introduction and Related Work 1

1 Introduction 3
- 1.1 Background and motivation . . . 3
- 1.2 Problem description . . . 4
- 1.3 Thesis contributions . . . 6
- 1.4 Thesis outline . . . 7

2 Literature Review 9
- 2.1 Ego-motion estimation with automotive radar . . . 9
  - 2.1.1 Scan-matching methods . . . 9
  - 2.1.2 Instantaneous methods . . . 11
- 2.2 Radar-based Multiple Extended Object Tracking (MEOT) . . . 15
- 2.3 Combined ego-motion estimation and MEOT approaches . . . 18
- 2.4 Summary of research gaps . . . 19

II Raw Signal based Combined Method 21

3 Methodology 23
- 3.1 Problem formulation . . . 23
- 3.2 Algorithm overview . . . 25
  - 3.2.1 Initialization phase . . . 28
  - 3.2.2 Combined estimation phase . . . 29
- 3.3 Signal preprocessing . . . 30
- 3.4 Tracking-aided ego-motion estimation . . . 34
  - 3.4.1 Tracking-aided segmentation . . . 34
  - 3.4.2 Iterative ego-motion estimation . . . 37
  - 3.4.3 Velocity transformation and state update . . . 39
- 3.5 Ego-motion compensation . . . 41
- 3.6 Multiple object tracking . . . 43

3.6.1 Clustering . . . 43
3.6.2 State estimation: Prediction and Update . . . 45
3.6.3 Gating and data association . . . 46
3.6.4 Tracking management . . . 50
3.7 Performance metrics . . . 51
3.7.1 Ego-motion estimation metrics . . . 51
3.7.2 Multiple object tracking metrics . . . 52

4 Simulation Results **55**
4.1 Simulation Setup . . . 55
4.2 Results and Discussions . . . 57

III Point Cloud based Combined Method **61**

5 Methodology **63**
5.1 Problem formulation . . . 63
5.2 Algorithm overview . . . 66
5.3 Tracking-aided ego-motion estimation . . . 68
5.3.1 Tracking-aided segmentation: adaptive gating . . . 68
5.3.2 Ego-motion estimation with RANSAC . . . 70
5.4 Multiple extended object tracking . . . 72

6 Simulation Results **77**
6.1 Simulation Setup and Performance Metrics . . . 77
6.2 Scene 1: A Single Truck . . . 82
6.2.1 Ego-motion estimation . . . 83
6.2.2 Multiple extended object tracking . . . 85
6.3 Scene 2: Multiple Trucks . . . 87
6.3.1 Ego-motion estimation . . . 88
6.3.2 Multiple extended object tracking . . . 91
6.4 Evaluation on More Scenes . . . 92

7 Real Dataset Results **95**
7.1 RadarScenes Dataset . . . 95
7.2 Ego-motion Estimation . . . 96

IV Closing Remarks **101**

8 Conclusion and Future Work **103**
8.1 Conclusion . . . 103
8.2 Future work . . . 104

Bibliography **107**

A Appendix **115**

List of Figures

2.1.1 Sketch showing the key idea behind instantaneous methods: the sinusoidal relationship between the Doppler/velocity and the azimuth angle of static targets (from [15]) 11

3.2.1 The overall processing pipeline of the raw signal based combined method for ego-motion estimation & multiple object tracking, with the 2 phases of initialization on N_{init} frames and combined estimation on the generic frame K at steady state. 26
3.3.1 Hardware high-level block diagram of an FMCW MIMO automotive radar [64] 31
3.3.2 Concept of 2D CA-CFAR detector applied on a range-Doppler matrix . 32
3.4.1 Concept and spatial relationship among three coordinate systems . . . 40
3.6.1 High-level block diagram of MOT algorithms with related submodules . 44
3.6.2 Concept of DBSCAN algorithm [42], with core points (red), border points (blue) and noise points (yellow). 45
3.7.1 Concept of GOSPA metric with three cases contributing to the overall error metric. 53

4.2.1 Ground truth and estimated trajectories for the ego-vehicle and eight moving objects from an example scenario when the ego-velocity is 10 m/s 59

5.1.1 An image captured by camera in RadarScenes: scene 1, frame 1143, where a large vehicle on the left is about to pass by the ego vehicle . . 64
5.1.2 The problem of RANSAC algorithm for static vs dynamic points segmentation, as observed using the example frame in Figure 5.1.1 65
5.2.1 The overall processing pipeline of the point cloud based combined method for ego-motion estimation & multiple extended object tracking 67

6.1.1 The basic simulation environment: road, static guardrails, and the ego vehicle in blue on the left-hand side 78
6.1.2 Installation configuration of the front-corner radar with its field of view 79
6.1.3 Example of ground truth generation for shape estimation: a right-front car, of which only two sides are visible to the radar on the ego-vehicle . 81
6.2.1 Simulated Scene 1: a single truck and two sedans 82
6.2.2 The tracking-aided segmentation results for an example frame: 2D positions of point cloud 84
6.2.3 The RANSAC curve fitting results for an example frame (in the Doppler-azimuth plane) 85
6.2.4 APE per frame for Scene 1: RANSAC-based method and proposed combined method (visualized in logarithmic scale) 86
6.2.5 Tracking results for simulated Scene 1: ground truth and estimated trajectories 88
6.3.1 Simulated Scene 2: three equally-spaced trucks 88

6.3.2 The tracking-aided segmentation results for an example frame: 2D positions of point cloud 89
6.3.3 The RANSAC curve fitting results for an example frame (in the Doppler-azimuth plane) 90
6.3.4 APE per frame for simulated Scene 2: RANSAC-based method and proposed combined method (visualized in logarithmic scale) 91
6.3.5 Tracking results for simulated Scene 2: ground truth and estimated trajectories 92

7.1.1 Radar configuration and field of view on the test vehicle [77] 96
7.2.1 RANSAC fitting results after tracking-aided segmentation for the same frame analyzed in Figure 5.1.2 98

List of Tables

1.2.1 Qualitative comparison between two input data formats: radar raw signal and radar point cloud 5

2.1.1 Summary of major scan-matching methods for radar-based ego-motion estimation 10
2.1.2 Summary of major instantaneous methods for radar-based ego-motion estimation 13

3.6.1 Number of hypotheses for $m(k)$ detections and $n(k)$ targets 47

4.1.1 Radar parameters in the simulations in Chapter 4 55
4.2.1 Performance evaluation with different ego-velocity in 3D space 58

6.1.1 Radar parameters in the 2D simulations run in Chapter 6 79
6.1.2 Ground truth values of shape estimation parameters 81
6.2.1 Performance metrics of ego-motion estimation: a comparison between RANSAC-based method and combined method 84
6.2.2 Performance metrics of multiple extended object tracking for simulated Scene 1 86
6.3.1 Ego-motion estimation performance comparison between RANSAC-based method and combined method for simulated scene 2 90
6.3.2 Performance metrics of multiple extended object tracking for simulated scene 2 92
6.4.1 Ego-motion estimation performance: averaged among 20 diverse scenes 93
6.4.2 Multiple extended object tracking performance: averaged among 20 diverse scenes 93

7.1.1 Parameters of corner Radar 3 in *RadarScenes* dataset 96
7.2.1 APE (m/s) over the full scene: comparison between RANSAC-based method and combined method 97
7.2.2 APE (m/s) during the 10-second encounter window with large vehicles: comparison between RANSAC-based method and combined method . . 98
7.2.3 APE (m/s) for static scenes: comparison between RANSAC-based method and combined method 99

List of Abbreviations

ADC Analog-to-Digital Converter

ADAS Advanced Driver Assistance Systems

APE Absolute Pose Error

ATE Absolute Trajectory Error

CA-CFAR Cell-Averaging Constant False Alarm Rate

CNN Convolutional Neural Network

CTA Cross Traffic Alert

CUT Cell Under Test

CV Constant Velocity

DBSCAN Density-Based Spatial Clustering of Applications with Noise

DoF Degree of Freedom

EKF Extended Kalman Filter

FFT Fast Fourier Transform

FMCW Frequency Modulated Continuous Wave

GMM Gaussian Mixture Models

GNC Graduated NonConvexity

GNSS Global Navigation Satellite System

GNN Global Nearest Neighbor

GOSPA Generalized Optimal Sub-Pattern Assignment

IF Intermediate-Frequency

IMM Interacting Multiple Models

IMU Inertial Measurement Unit

I/Q In-phase and Quadrature

JPDA Joint Probabilistic Data Association

LiDAR Light Detection and Ranging

LS Least Squares

LSTM Long Short-Term Memory

MAP Maximum A Posteriori

ME-LSQ M-Estimated Least SQuare

MEOT Multiple Extended Object Tracking

MHT Multiple Hypothesis Tracking

MIMO Multiple-Input Multiple-Output

MMSE Minimum Mean Squared Error

MOT Multiple Object Tracking

MVEE Minimum Volume Enclosing Ellipse

NDT Normal Distributions Transform

ODR Orthogonal Distance Regression

PDF Probability Density Function

PHD Probability Hypothesis Density

PMBM Poisson Multi-Bernoulli Mixture

PRI Pulse Repetition Interval

RANSAC RANdom SAmple Consensus

RCS Radar Cross Section

RDS Range-Doppler Spectrum

RFS Random Finite Set

RMSE Root Mean Square Error

RMM Random Matrix Model

RTE Relative Trajectory Error

SAR Synthetic Aperture Radar

SLAM Simultaneous Localization and Mapping

SLAMMOT Simultaneous Localization, Mapping, and Moving Object Tracking

SNR Signal-to-Noise Ratio

TEMPSAC TEMPoral SAmple Consensus

TWLSQ Temporally Weighted Least SQuare

UKF Unscented Kalman Filter

WLS Weighted Least Squares

Part I
Introduction and Related Work

1 Introduction

This chapter begins with the background and motivation of the thesis in Section 1.1. In the context of automotive radar signal processing, the combined problem of ego-motion estimation and multiple object tracking is then described in Section 1.2. Next, Section 1.3 presents the general contributions of this work, while Section 1.4 outlines the overall structure of the thesis, with the proposed methodology broadly divided into two parts.

1.1 Background and motivation

In autonomous driving and mobile robotics applications, Frequency Modulated Continuous Wave (FMCW) Multiple-Input Multiple-Output (MIMO) automotive radar offers significant capabilities in both localization and perception tasks [1–4]. By transmitting FMCW waveforms and leveraging MIMO antenna technology, it can accurately measure the range, relative velocity, azimuth, and elevation angle of multiple objects, enabling full three-dimensional situational awareness. Compared with camera and Light Detection and Ranging (LiDAR), radar is notably more robust under adverse weather conditions (e.g., rain, fog, smoke) and in low-light environments, and it provides direct Doppler-based velocity measurements. Moreover, radar systems can achieve long-range coverage exceeding 300 meters [1]. These attributes collectively make radar an essential sensor for safe, reliable, and robust autonomous decision-making in complex and dynamic environments.

Measuring the vehicle's own motion with radar, namely 'ego-motion estimation', is a fundamental self-localization task in autonomous driving systems. Accurate ego-motion estimates are crucial for downstream modules such as multiple object tracking and environment mapping [5]. A variety of sensor modalities, including Global Navigation Satellite System (GNSS), Inertial Measurement Unit (IMU), wheel encoders, cameras, and LiDAR, have been used to address this task [6]. However, radar-based ego-motion estimation is gaining increasing attention due to its unique advantages. Unlike inertial sensors or wheel odometry, radar is not prone to drift or slippage, making it more stable over time. It also operates reliably in challenging conditions such as rain, fog, low light, and in environments like urban canyons or tunnels, where vision, LiDAR, and GNSS systems often face limitations [3, 6]. Furthermore, unlike cameras and LiDAR, radar provides velocity measurements via the Doppler effect, enabling ego-motion estimation using only one data frame.

Despite these advantages, ego-motion estimation with radar remains challenging in dynamic environments, where a significant portion of surrounding objects may be moving [7–9]. In principle, ego-motion estimation relies on the measurements of static objects in the scene. However, when the environment contains a high ratio of moving

objects, the accuracy of ego-motion estimation can deteriorate significantly. Thus, the core challenge lies in reliably distinguishing between static and moving objects from radar measurements, especially when the sensor itself is in motion.

In addition to localization, environment perception for autonomous driving systems involves detecting and tracking moving objects within the sensor's field of view [2]. For common road participants such as vehicles, bicycles, and pedestrians, automotive radar typically produces multiple spatially distributed detections per object [10]. Based on these detections, the task of Multiple Extended Object Tracking (MEOT) aims to jointly estimate the kinematic states and spatial extents of multiple objects over time. Traditionally, MEOT has been treated as a separate, downstream task that assumes access to accurate ego-motion estimates. Indeed, existing radar-based MEOT algorithms commonly rely on ego-motion information from external sensors such as GNSS or IMU. However, these sources may be unreliable or unavailable in challenging environments such as urban canyons, tunnels, or cost-constrained systems.

To address these challenges, this thesis investigates the integration and combination of ego-motion estimation and MEOT into a unified radar-only estimation framework. Performing both tasks together can mitigate issues that arise when they are handled separately. On the one hand, MEOT offers temporally coherent information about moving objects, which can help distinguish them from static objects. On the other hand, radar-based ego-motion estimation enables self-contained motion compensation for MEOT, avoiding dependency on external sensors. This mutual reinforcement enhances the overall robustness in complex and dynamic environments. While camera- and LiDAR-based approaches have explored such combined strategies [11–13], radar-based methods remain to the best of the author's knowledge largely underexplored. Consequently, the key research question of this thesis can be stated as follows: **How can we develop a unified framework that leverages only automotive radar data to achieve accurate static/moving object segmentation, thereby enabling combined ego-motion estimation and multiple extended object tracking in dynamic environments?**

1.2 Problem description

In this thesis, the combined problem of ego-motion estimation and multiple extended object tracking is addressed using measurements from a single automotive radar sensor. In the context of autonomous driving, when a vehicle is driving on the road, there are both static and moving objects in its surrounding environment. Static objects may include roadside trees, buildings, and parked vehicles, while moving objects often consist of moving cars, bicycles, and pedestrians. This thesis mainly focuses on dynamic driving scenarios. Unlike static scenarios where most radar detections originate from stationary structures, dynamic scenarios are characterized by a significant proportion of reflections from moving objects. Such situations frequently occur in real-world driving, for example, when large vehicles (e.g., trucks or buses) are approaching the ego-vehicle on an open road, or when the ego-vehicle is parking with other moving cars in a parking lot [7, 9].

The input to the problem is the measurement data obtained from a single automotive

radar mounted on the ego-vehicle. In automotive radar signal processing, two common forms of radar data are typically used: radar raw signal and radar point cloud. Radar raw signal refers to the low-level complex signals directly obtained from the radar receiver channels. These signals retain full phase and amplitude information across multiple antenna channels. Another format is the radar point cloud, which is widely applied for object detection, semantic segmentation, tracking and localization [1, 14]. Point clouds are generated by applying signal processing techniques—such as range, Doppler and angle Fast Fourier Transform (FFT)s—to the radar raw signal, followed by target detection and optional clustering, resulting in a set of discrete points. Each point contains information including range, Doppler velocity, azimuth angle, and Radar Cross Section (RCS).

Table 1.2.1 summarizes the key differences between the two input data formats. Radar raw signal contains richer information, including noise and background effects, and enables a faster update rate (i.e., more frequent measurements per second compared to forming point clouds), which benefits low-latency estimation of ego motion and other applications such as high-resolution imaging [6]. However, due to its large storage demands, raw signal is rarely included in public datasets, and few existing ego-motion estimation and tracking algorithms operate directly on this format. In contrast, radar point clouds offer lower storage requirements and lower processing complexity, since only the information of detected targets is preserved after target detection. This format is also readily available in most public radar datasets, making it a well-established choice for localization and tracking [14].

In summary, each format has its own advantages and limitations. Raw signal offers richer data and faster update time, while point cloud is efficient and accessible. Therefore, this thesis investigates both data formats to explore their respective advantages in solving the combined problem as stated in the previous section.

Table 1.2.1: Qualitative comparison between two input data formats: radar raw signal and radar point cloud

Property	**Radar Raw Signal**	**Radar Point Cloud**
Information Richness	Full (includes noise and background)	Limited (only detected targets)
Storage Space	High	Low
Update rate	Fast	Slow
Processing Complexity	High	Low / Moderate
Algorithm Maturity	Low	High
Availability in Datasets	Rare / Limited	Easy

The output of the combined problem consists of two components: ego-motion estimation and multiple extended object tracking. Ego-motion estimation refers to estimating the vehicle's own motion, typically its translational and rotational velocities in 2D plane (without the elevation dimension) or 3D space (with the elevation dimension). Over time, the trajectory of the ego-vehicle in each dimension is also of interest. This

task is essential for localization and often serves as the front end of a Simultaneous Localization and Mapping (SLAM) processing pipeline. Multiple Object Tracking (MOT) traditionally focuses on estimating the kinematic states (e.g., position and velocity) of multiple moving objects over time, assuming each object is a point target. This is a reasonable assumption for example in surveillance radars with relatively low resolution. In contrast, multiple extended object tracking additionally estimates the spatial extent (e.g., size and orientation) of each object, accounting for the fact that real-world objects often generate multiple radar detections in a single frame, which is the common situation for high-resolution automotive radars. The tracking output therefore includes the position, velocity, size, and orientation of each moving object, along with their trajectories across time.

Summarizing, as radar raw signals and point clouds have their own advantages and disadvantages, both are considered in this thesis. Specifically, **Part II** presents a method based on radar raw signals to solve the combined problem of ego-motion estimation and multiple object tracking. As an earlier-stage effort in the thesis journey, it focuses more on theoretical formulation and does not include yet spatial extent estimation.

Next, **Part III** introduces a method based on radar point clouds, which solves the combined problem of ego-motion estimation and multiple extended object tracking. This method incorporates spatial extent estimation and evaluation, making it more applicable to realistic scenarios in automotive.

1.3 Thesis contributions

The main contributions of this thesis are summarized in the following aspects:

- In this thesis, a novel combined ego-motion estimation and multiple object tracking method based on radar raw signals was proposed, demonstrating its theoretical feasibility through simulation-based validation.

- This work then developed and evaluated a combined estimation framework which uses only automotive radar point cloud data to perform both ego-motion estimation and multiple extended object tracking. In simulation, it outperformed the RANSAC-based baseline method [15], by reducing the APE metric by 2.00 m/s and the RTE metric by 1.58 m, with tracking metric GOSPA of 3.70 and size estimation RMSEs of 1.04 m (major axis), 0.55 m (minor axis), and 8.40° (orientation).

- The proposed method significantly improved ego-motion estimation performance in real-world dynamic driving scenarios from the *RadarScenes* dataset, with APE reductions of **25.9%** (the entire scene) and **56.9%** (local segments) over the state-of-the-art method [15], effectively addressing challenges such as oncoming large vehicles while driving. Prior tracking information is used to identify and filter out detections from moving objects, which enhances the reliability of the static point selection and contributes to more accurate ego-motion estimation.

- Part of this thesis has been accepted for publication at the European Radar Conference (EuRAD 2025), which will take place in Utrecht, The Netherlands, in September 2025. The paper text is provided in Appendix.

1.4 Thesis outline

This thesis is organized into four main parts, with **Part II** and **Part III** focusing on the investigation of the proposed raw signal-based and point cloud-based methods, respectively:

- **Part I** presents the introduction and related work of this thesis. In Chapter 2, the related work on ego-motion estimation and multiple extended object tracking with automotive radar is reviewed.
- **Part II** proposes a combined method using as input the radar raw signals. It includes two chapters:
  - Chapter 3: This chapter first formulates the combined problem of ego-motion estimation and multiple object tracking using radar raw signals. Next, the raw-signal-based method is explained in detail, including signal preprocessing, tracking-aided ego-motion estimation, ego-motion compensation, and multiple object tracking. Finally, the performance metrics for validating the combined method are described.
  - Chapter 4: This chapter demonstrates simulation results of the proposed raw signal method, and discusses its advantages and limitations.
- **Part III** provides another formulated combined method based on radar point cloud data, aiming to apply the theoretical framework to practical challenges in dynamic driving scenarios. It contains three chapters:
  - Chapter 5: This chapter introduces the practical challenges in radar-based ego-motion estimation and presents the proposed combined method designed to address them, with a focus on its differences and improvements over the method in Part II.
  - Chapter 6: This chapter presents simulation results using synthetic radar point cloud data from a dedicated MATLAB tool, evaluating the performance of ego-motion estimation and multiple extended object tracking in various driving scenarios.
  - Chapter 7: This chapter validates the proposed method on the real-world *RadarScenes* dataset, demonstrating its superior performance over traditional methods in handling ego-motion estimation in dynamic scenes.
- **Part IV** serves as a closing discussion for the whole thesis. Chapter 8 summarizes the main conclusions and outlines potential directions for future work.

2 Literature Review

This chapter reviews related work on ego-motion estimation and multiple extended object tracking with automotive radar from the literature. In Section 2.1, previous methods for estimating the vehicle ego-motion using radar data are introduced. Section 2.2 provides a discussion about existing approaches on radar-based MEOT. Next, Section 2.3 illustrates the combined approaches for ego-motion estimation and MEOT, focusing primarily on camera- or LiDAR-based methods, as radar-specific approaches remain scarce in the literature. Finally, Section 2.4 summarizes the key insights and identifies the research gaps that motivate the proposed work.

2.1 Ego-motion estimation with automotive radar

In recent years, radar-based ego-motion estimation methods have developed rapidly in the area of autonomous driving and mobile robotics. Existing methods can generally be divided into two categories: scan-matching methods and instantaneous methods. These two types of methods differ in the number of input data frames required for each estimation. While scan-matching methods estimate ego-motion based on the alignment of measurements from two consecutive radar frames, instantaneous methods require only the current radar frame as input. The related work on both methods is analyzed in Sections 2.1.1 and 2.1.2, respectively.

2.1.1 Scan-matching methods

The core principle of scan-matching methods is to estimate the relative Euclidean transformation between consecutive radar point clouds [16]. These approaches leverage point cloud registration techniques adapted from LiDAR-based methods to exploit spatio-temporal consistency and perform data association. If the positions of the same static objects can be reliably identified by matching the point cloud of the current frame with that of the previous frame, ego motion can be inferred from the positional differences. Table 2.1.1 summarizes typical scan-matching methods in the literature, along with their key characteristics.

In 2015, Barjenbruch et al. [16] proposed a joint optimization framework that incorporates both spatial and Doppler-based metrics. To compare spatial information, they represent the reference and the new frame measurements with Gaussian Mixture Models (GMM) and align them using a robust point set registration algorithm [17]. Building on this approach, Rapp et al. [18] introduced a probabilistic ego-motion estimation framework that improves accuracy by employing the Normal Distributions Transform (NDT) [19] algorithm for more efficient spatial alignment. Their method reduces computational complexity while demonstrating reliable performance on real-

Table 2.1.1: Summary of major scan-matching methods for radar-based ego-motion estimation

Paper	**Year**	**Scenario**	**Sensor Input**	**Data Association Method**
Barjenbruch et al. [16]	2015	Automotive	Single-radar point cloud	Robust point set registration [17]
Rapp et al. [18]	2017	Automotive	Multi-radar point cloud	NDT [19]
Haggag et al. [20]	2022	Automotive	Single-radar point cloud	Point-to-distribution registration
Lu et al. [21]	2020	Robotics	Single-radar point cloud + IMU	Deep learning

world data. Subsequently, to mitigate the local maxima issue induced by the summing approximation [18], a sophisticated noise model was incorporated into the objective function in [20]. This enhancement results in credible and robust performance even in scenarios of high outlier rates.

Despite these advancements, scan-matching methods still exhibit notable limitations. First, since radar point clouds are inherently sparser and more prone to noisy artifacts compared to those from LiDAR [14], the scintillating behavior observed across multiple frames complicates precise temporal alignment. Due to multipath effects, harmonics and other noises, after radar detection, radar point clouds for each frame can contain lots of false alarms. Moreover, in driving scenarios, the static objects, such as roadside fence and trees, are typically detected as spread and sparse, which easily leads to missed detections. As a result, it is difficult to achieve reliable data association for static objects across multiple radar scans [22]. In order to solve the issue of noisy correspondences, a data-driven deep learning based approach was proposed in [21]. With Convolutional Neural Network (CNN) feature extractors and Long Short-Term Memory (LSTM) layers to model the temporal dependency, it can jointly optimize the ego-motion estimation and perform implicit correspondence association between consecutive frames. However, deep learning methods require a very large dataset to train the neural network, and the network architecture design is complicated and less interpretable.

Second, scan-matching methods exhibit reduced robustness in dynamic environments, where moving objects and rapid environmental changes may induce misalignments. For example, if there are substantial changes in local scenes (e.g., turning, upslope), such methods are less robust and prone to error accumulation [23]. Moreover, the computational burden of these methods —particularly those involving probabilistic optimization or iterative registration— can hinder real-time performance, which is critical in autonomous driving applications. Given these limitations, this thesis focuses on instantaneous methods, which are better suited for radar data and will be discussed in the following section.

2.1.2 Instantaneous methods

Compared to other competitive sensors such as camera and LiDAR, radar has the unique advantage of directly measuring the velocity of targets via the Doppler effect. This capability enables ego-motion estimation using a single radar frame, eliminating the need to compare measurements across multiple consecutive frames as in scan-matching methods. Instantaneous methods exploit the sinusoidal relationship between Doppler or velocity and the azimuth angle of a stationary object to estimate motion. As illustrated in Figure 2.1.1, for all points detected on static objects, the Doppler velocity (green arrows) is a projection of the true ego velocity (blue arrows) in the radial direction between the stationary point and the radar sensor line of sight (indicated by black dashed lines). Therefore, the Doppler/velocity is a sinusoidal function of azimuth angle and ego-vehicle velocity.

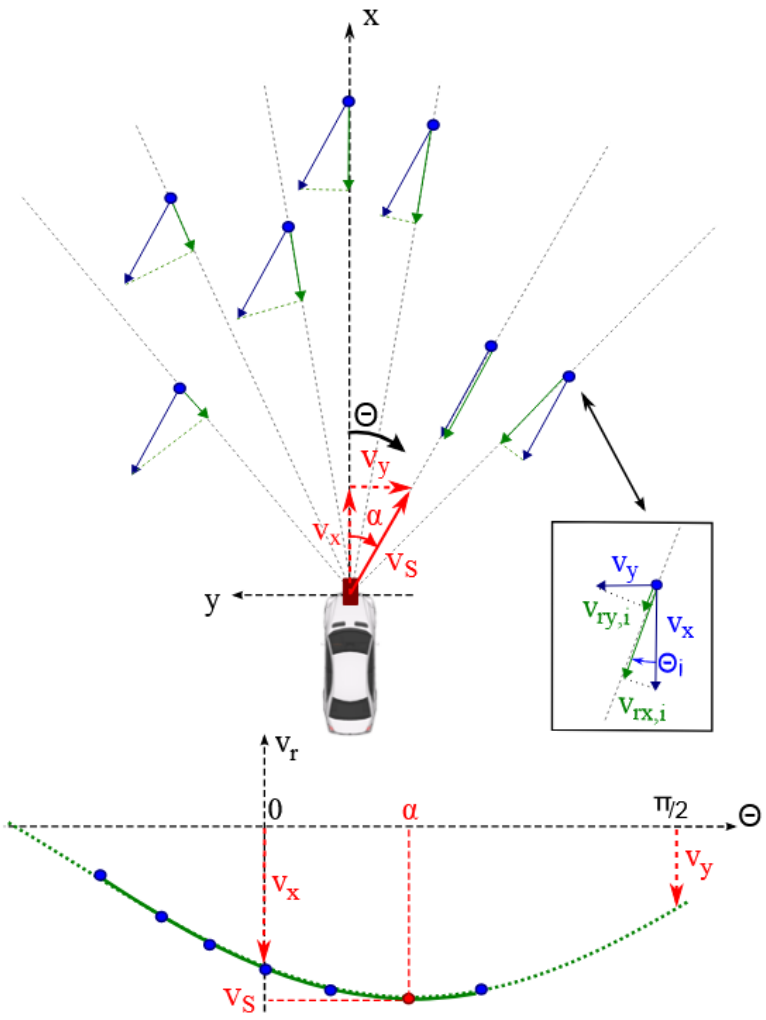


Figure 2.1.1: Sketch showing the key idea behind instantaneous methods: the sinusoidal relationship between the Doppler/velocity and the azimuth angle of static targets (from [15])

In real driving scenarios, radar detections include not only reflections from static objects, but also those from moving targets and clutter caused by multipath propagation, possible interference, and hardware imperfections [24,25]. Therefore, the primary objective is to identify the detections truly originating from stationary objects (i.e., inliers). Once the inliers have been identified, the ego-vehicle velocity can be inferred using appropriate estimation techniques. Table 2.1.2 provides a summary of representative instantaneous methods. Different approaches adopt various techniques to segment static detections and perform robust motion estimation. Based on the table, the following discussion is organized around three key aspects: static target detection method, estimation solver, and sensor input.

1. **Static target detection method**
   The first instantaneous ego-motion estimation method was proposed by Kellner et al. [15] in 2013. Their approach employs RANdom SAmple Consensus (RANSAC) [26] to identify stationary targets and estimate the ego-motion. Originally developed for image processing, RANSAC is an iterative method for robustly estimating model parameters from data containing outliers, effectively serving as an outlier detection technique. Assuming a predominantly static environment and an Ackermann-steered vehicle with no lateral displacement, their approach requires at least two measurements from static targets to compute a valid 2 Degree of Freedom (DoF) solution for the ego-motion estimation problem. In a subsequent study [27], the method is extended to incorporate multiple radar sensors, enabling the estimation of the complete 2D motion state (3 DoF) of the ego-vehicle, including longitudinal and lateral velocities as well as yaw rate.

   Kellner's method serves as a foundational instantaneous approach and remains a widely recognized benchmark in the field of radar-based ego-motion estimation. By employing RANSAC, the method can achieve high accuracy under a majority of driving conditions. However, it relies on the assumption that most radar detections originate from static objects, with only a small fraction attributed to moving targets and clutter. This assumption makes the method vulnerable in situations with a high outlier ratio (i.e., when more than 50% of detections are outliers) [8]. For instance, the RANSAC solution may fail when most of the radar detections come from large moving objects such as trucks or buses. Similar failure cases can occur in parking lots where multiple vehicles are in motion [9].

   Recently, several instantaneous methods have been proposed to enhance robustness in dynamic environments containing large moving objects. Some approaches [7, 28] leverage the prior that radar returns from the ground are always static. Park et al. [28] placed two radar sensors orthogonally to ensure more returns from the static ground. In [7], both Doppler and geometric characteristics of the ground were used to robustly identify ground points from radar point clouds. Assuming that the ground forms a horizontal plane at a certain height, the authors designed a joint algorithm to simultaneously detect ground points and estimate radar velocity. However, unlike LiDAR sensors, real radar data often exhibit sparse ground returns and poor elevation resolution [14], making it difficult to segment ground points based on height and use them for accurate motion estimation.

Table 2.1.2: Summary of major instantaneous methods for radar-based ego-motion estimation

Paper	**Year**	**Scenario**	**Sensor Input**	**Static Target Detection Method**	**Estimation Solver**
Kellner et al. [15]	2013	Automotive	Single-radar point cloud	RANSAC [26]	LS
Kellner et al. [27]	2014	Automotive	Multi-radar point cloud	RANSAC	LS, WLS, ODR
Park et al. [28]	2021	Robotics	Multi-radar point cloud + IMU	Ground segmentation, RANSAC	LS
Chen et al. [7]	2023	Robotics	Single-radar point cloud + IMU	Ground segmentation	LS
Zhu et al. [5]	2023	Automotive	Single-radar point cloud	Deep learning	WLS
Zhu et al. [8]	2025	Automotive	Multi-radar point cloud	Deep learning	WLS
Herraez et al. [9]	2024	Automotive	Single-radar point cloud	RANSAC with initial filtering step	LS
Lovett et al. [29]	2025	Robotics	Single-radar point cloud	TEMPSAC, TWLSQ	LS, WLS
Kramer et al. [30]	2020	Robotics	Single-radar point cloud + IMU	Cauchy robust loss	Levenberg-Marquardt optimizer
Huang et al. [31]	2024	Robotics	Multi-radar point cloud + IMU	Cauchy robust loss	Levenberg-Marquardt optimizer
Zhuang et al. [23]	2023	Robotics	Single-radar point cloud + IMU	GNC [32]	LS
Kim et al. [33]	2024	Automotive	Single-radar point cloud	ME-LSQ	LS
Doer et al. [34]	2020	Robotics	Single-radar point cloud + IMU	RANSAC	LS
Galeote et al. [35]	2024	Automotive	Single-radar point cloud	RANSAC	LS
Yuan et al. [6]	2023	Automotive	Single-radar raw signal	None	LS
Yuan et al. [36]	2024	Automotive	Single-radar raw signal	Iterative method	LS

Other methods [5, 8] employ deep learning techniques to learn the probability of each detection being static based on real-world data. In [5], Zhu et al. used neural networks to extract complex features from radar point clouds to predict pointwise weights and offsets. The weights represent the likelihood of a point being a static inlier. Once the weights are learned, ego-motion is estimated using the Weighted Least Squares (WLS) approach. In their follow-up work [8], they incorporated radar data from multiple sensors via homogeneous late fusion, addressing practical issues such as non-zero vehicle acceleration and low inlier availability. Their method demonstrates promising results, outperforming RANSAC especially in high outlier ratio scenarios (i.e., $> 50\%$). However, deep learning methods typically require large-scale training datasets with carefully annotated labels from other sensors. Moreover, these methods often suffer from limited interpretability, making it difficult to understand or trust their internal decision processes.

A different research direction is to improve on RANSAC or adopt alternative outlier rejection strategies [9, 23, 29–31, 33]. To improve RANSAC, most methods incorporate the temporal information from previous estimates to reject outliers. For instance, Herraez et al. [9] introduced an initial filtering step before RANSAC that adopts a constant velocity model to filter out outliers. The method predicts ego-motion from two previous frames and retains only samples that match the predicted motion within a certain margin. Similarly, Lovett et al. [29] proposed two RANSAC-based variants: TEMPoral SAmple Consensus (TEMPSAC) and Temporally Weighted Least SQuare (TWLSQ), which estimate ego-motion over a temporally weighted sliding window. While TEMPSAC adopts weighted sampling based on their temporal proximity, TWLSQ uses random sampling with a temporally WLS estimator. These methods achieve a 27% improvement in position accuracy over standard RANSAC in cluttered indoor environments.

In [30, 31], the use of Cauchy robust loss function and the nonlinear Levenberg-Marquardt optimizer was proposed to mitigate the effect of large residuals, thus improving estimation robustness. Zhuang et al. [23] replaced RANSAC with the Graduated NonConvexity (GNC) method [32], a more robust non-minimal solver capable of handling up to 80% outliers. Moreover, Kim et al. [33] adopted the M-Estimated Least SQuare (ME-LSQ) method, which is based on the M-estimation [37]. By initializing with the previous motion estimate, the algorithm can better filter noise and inconsistencies caused by moving objects, resulting in smoother and more reliable trajectory estimation.

While these approaches significantly improve robustness by leveraging the temporal consistency of ego-motion estimates, they generally ignore the temporal continuity of surrounding moving objects. Incorporating such information through object tracking could further mitigate the impact of dynamic outliers, which motivates the combined approaches proposed in this work.

2. **Estimation solver**
   Given sufficient Doppler and azimuth angle measurements of static targets, the ego-motion estimation problem can be formulated as a linear estimation or curve fitting problem. Most instantaneous methods [7, 9, 15, 23, 28, 33–35] adopt Least

Squares (LS) to estimate ego-motion from the detected static points due to its simplicity and efficiency. To enhance robustness, some approaches [5, 8, 29] apply WLS, where each point is weighted according to its likelihood of being static, derived from learning-based models or temporal consistency. A few studies [27] explore Orthogonal Distance Regression (ODR) to account for errors in both dependent and independent variables. While ODR can theoretically improve accuracy, its increased computational cost often outweighs the marginal performance gains demonstrated in automotive scenarios, making it less commonly used [34].

3. **Sensor input**
   The type and configuration of input sensors play a critical role in determining the feasibility and accuracy of instantaneous ego-motion estimation. While a single radar sensor is generally sufficient to estimate translational velocity, estimating rotational velocity is more challenging due to the limited observability of rotational motion. To address this, some works [5, 15, 33] assume the vehicle satisfies the Ackermann kinematic model, under which lateral and vertical velocities are neglected. This assumption allows the yaw rate to be derived directly from translational velocity and sensor geometry. Other studies improve observability and robustness by employing multi-radar fusion [8, 27, 31] or integrating radar and IMU data [28, 30, 34]. These methods rely on sensor fusion techniques, often implemented within filtering or graph optimization frameworks and implying some form of synchronization between the different sensors.

   While most existing methods take radar point clouds as input, a recent trend explores the use of raw radar signals (i.e., baseband signals prior to range-Doppler processing). In [6], a two-stage optimization method leverages phase differences between chirp groups within a single frame to estimate ego-velocity. Building on this, [36] proposes an iterative approach to detect static targets, enhancing robustness in dynamic environments. Compared to point-cloud-based methods, which depend on multiple signal processing stages, raw-signal-based approaches enable faster estimation—potentially within a single frame or even from chirp to chirp. Moreover, they offer better compatibility with high-resolution imaging or automotive Synthetic Aperture Radar (SAR) algorithms [6]. Despite these advantages, evaluating such methods on real-world datasets remains challenging, as raw radar signals are rarely available due to their large storage requirements.

2.2 Radar-based Multiple Extended Object Tracking (MEOT)

In recent years, Multiple Extended Object Tracking (MEOT) has gained increasing attention in radar-based perception, particularly in the context of autonomous driving and intelligent transportation systems [10, 38, 39]. Building upon conventional MOT frameworks, which models each object as a single point, MEOT accounts for the fact that radar returns from one object may consist of multiple detections per frame, reflecting the object's shape and size information. This is especially true for automotive radar systems, where vehicles or pedestrians often generate multiple radar detections across their surface due to the high resolution of the radar and the relatively short

wavelength.

To effectively handle such data, MEOT algorithms jointly estimate the kinematic state (e.g., position, velocity) and the spatial extent (e.g., size, shape, orientation) of each object. Given the complexity of the task, three core challenges must be addressed:

1. **Handling multiple detections and estimating spatial extent:** Since extended objects often generate multiple detections per frame, robust modeling is required to infer the object's physical dimensions and shape from scattered and noisy detections.

2. **Clutter removal and false alarm suppression:** Radar data often contain false alarms due to multipath propagation, interference, and sensor noise [14], which must be suppressed in the tracking module to isolate true object measurements from false detections.

3. **Tracking multiple targets:** In complex dynamic environments, tracking of multiple objects requires solving data association problems (i.e., how to assign measurements to multiple existing tracks) and state estimation problems (i.e., how to accurately infer object states from noisy observations) accurately and efficiently.

Existing approaches to these challenges are analyzed in the following discussion:

1. **Handling multiple detections and estimating spatial extent**
   Radar returns from an extended object typically consist of multiple detections per frame due to the distributed scattering nature of radar sensing [40]. These detections encode the shape and size information of the object. For the processing of multiple detections, a widely adopted solution is spatial clustering, especially the Density-Based Spatial Clustering of Applications with Noise (DBSCAN) clustering algorithm [41] which is proven to be effective for many automotive radar tracking applications [42].

   After detections are grouped into different clusters, the spatial extent information of each cluster is estimated according to a certain shape model. A widely used shape model is the Random Matrix Model (RMM) proposed by Koch [43–45], in which the extent of an object is represented by a symmetric positive-definite matrix that models the object shape as an ellipse or ellipsoid. The combined state is estimated using a joint Gaussian-inverse Wishart distribution. This model provides an analytically tractable framework for simultaneously updating the state and extent based on the spread of associated measurements.

   Recently, another approach to process multiple detections utilizes deep learning techniques to follow the tracking-by-detection paradigm [46, 47]. Such a paradigm involves first performing object detection with a deep neural network in each frame independently, and then linking these detected object types and 3D bounding boxes across consecutive frames to form continuous object trajectories. In [47], the PointPillars [48] neural network is for example adopted for object detection. With input point clouds, this network can predict the position, size, and type of each object by assigning a 3D bounding box.

2. **Clutter removal and false alarm suppression**
   Clutter is a significant challenge in radar-based tracking systems, arising from multipath propagation, sidelobe effects, possible interference, and hardware imperfections. A common strategy to remove clutter is to use pre-filtering techniques, where radar detections are thresholded based on Doppler or velocity values and RCS. For instance, in [49], detections with low radial velocity or low RCS are discarded as probable clutter before tracking begins. While efficient, such heuristics may also remove valid weak detections, especially for vulnerable targets like pedestrians. More principled approaches incorporate gating techniques using the Mahalanobis distance between the predicted target state and incoming measurements. This allows elliptical gating regions that adapt to the shape and uncertainty of each target, improving robustness compared to fixed-radius gating [44].

3. **Tracking multiple targets**
   To track multiple targets simultaneously, there are mainly two steps. The first step solves the data association problem, where the associations between the new measurements and the existing tracks are established. The second step handles the state estimation problem, where the hidden states of each track are estimated based on the assigned data association hypothesis.

   Data association is a combinatorial optimization problem that aims to determine the most likely assignment between measurements and existing tracks. Existing solutions include the Hungarian algorithm [50], which computes the optimal one-to-one assignment and discards all other possibilities, and Murty's algorithm [51], which generates a ranked list of the top-K most likely association hypotheses based on their likelihood scores.

   State estimation involves inferring the hidden state of targets (e.g., position, velocity) from noisy measurements. The Kalman Filter is the standard solution for linear Gaussian systems. For nonlinear systems, the Extended Kalman Filter (EKF) and the Unscented Kalman Filter (UKF) provide first- and second-order approximations. In highly nonlinear and non-Gaussian environments, the Particle Filter offers better flexibility by representing the posterior with a set of weighted samples.

   Several classic multi-target tracking algorithms integrate data association and state estimation. The Global Nearest Neighbor (GNN) algorithm [52] is a deterministic method that approximates the multi-target posterior by selecting the single most likely association hypothesis at each update. In contrast, the Joint Probabilistic Data Association (JPDA) filter [53] considers all feasible associations, merging their contributions into a single Gaussian distribution. As a result, even hypotheses with lower likelihoods are partially retained, improving robustness in cluttered environments. The Multiple Hypothesis Tracking (MHT) algorithm [54] further extends this idea by maintaining a Gaussian mixture, where each component corresponds to a global association hypothesis. This allows MHT to explicitly propagate uncertainty over multiple time steps, offering higher accuracy at the cost of increased computational complexity.

More recently, algorithms based on Random Finite Set (RFS) theory have gained attention for their principled handling of an unknown and time-varying number of targets. The Probability Hypothesis Density (PHD) filter [55] propagates the first-order moment of the target set, offering efficient multi-target tracking without explicit data association. The Poisson Multi-Bernoulli Mixture (PMBM) filter [56] further models both detected and undetected targets explicitly and provides a full Bayesian solution with scalable approximations, achieving state-of-the-art performance in challenging tracking scenarios.

In summary, existing radar-based MEOT algorithms tackle key challenges such as extent estimation, clutter suppression, and multi-target tracking. Recent advances combine model-based filtering with clustering and deep learning, enabling accurate perception in dynamic environments.

2.3 Combined ego-motion estimation and MEOT approaches

Accurate scene understanding of a dynamic environment requires estimating the motion of both the ego vehicle and surrounding moving objects. Traditionally, ego-motion estimation and MEOT have been treated as two separate modules. Solutions usually rely on certain assumptions, such as the static environment assumption for ego-motion estimation and the accurate ego velocity assumption for MEOT. However, in complex dynamic environments, it is difficult or even impossible to satisfy these assumptions [57].

To guarantee the performance of both tasks under such conditions, combined ego-motion estimation and MEOT approaches have gained increasing attention in recent years. These methods leverage the inherent interdependence between the two tasks to achieve mutual benefits. On the one hand, considering the impact of moving objects enables better segmentation between moving and static objects, thereby improving the robustness of ego-motion estimation. On the other hand, accurate ego-motion estimation results can provide a stable reference for more consistent and precise object tracking.

A foundational effort in this direction was laid in 2007 when Wang et al. [11] introduced a groundbreaking Bayesian-based theory of Simultaneous Localization, Mapping, and Moving Object Tracking (SLAMMOT). This work formulates the combined problem as a unified state estimation task, and proposes a solution using two decoupled estimators based on simplified assumptions. However, this approach oversimplifies the complex interactions between static and dynamic targets in real-world environments, limiting its applicability in practical scenarios.

Since then, ego-motion estimation and multiple object tracking have increasingly been treated as a combined problem, with extensive research showing that these two tasks can mutually benefit from each other. Most existing methods have been developed utilizing either camera or LiDAR sensors rather than radar [12, 13, 58–60], and are reviewed in the rest of this section. There are two primary challenges in solving this combined problem: accurate separation of static and moving objects and strong coupling between the two tasks.

First, the accurate separation of static and dynamic objects is crucial. Ego-motion estimation relies on static objects' features, while tracking requires correctly identifying and associating moving entities. Misclassification can lead to errors in localization and fragmented or incorrect target trajectories. Camera-based methods typically leverage semantic and texture information to perform feature matching across multiple consecutive frames. For instance, VDO-SLAM [12] employs the Mask R-CNN semantic segmentation neural network to segment dynamic objects, while DynaSLAM II [58] combines semantic information with multi-view geometry to detect moving regions. LiDAR-based approaches rely more on geometric consistency in 3D point clouds to perform instance segmentation and object detection. DL-SLOT [13] and IMM-SLAMMOT [59] both use the 3D object detection neural network PointRCNN [61] to identify potentially dynamic objects and filter out their corresponding points prior to ego-motion estimation.

Second, the strong coupling between ego-motion estimation and tracking introduces a major algorithmic challenge. Ego-motion is used to compensate for sensor motion and stabilize tracking, while accurate tracking can improve motion segmentation and provide feedback for localization. When treated independently, the two tasks may propagate errors to one another. To address this, many existing methods adopt graph optimization or joint filtering frameworks. In VDO-SLAM [12], both the camera and object trajectories are jointly refined using bundle adjustment in a graph-based back-end. Similarly, LIO-SEGMOT [60] presents a dynamic LiDAR SLAMMOT method based on factor graph optimization. Another approach is joint filtering. In IMM-SLAMMOT [59], the motion of the ego vehicle and the tracked objects are simultaneously estimated using Interacting Multiple Models (IMM) filters, enabling real-time feedback between localization and tracking modules.

2.4 Summary of research gaps

In this section, based on the literature reviewed in this chapter, the key research gaps are summarized in terms of radar-based ego-motion estimation, multiple extended object tracking, and combined methods.

Ego-motion estimation: Although many methods have been proposed to enhance the robustness of RANSAC in dynamic environments, such as using static ground priors, deep learning, and advanced outlier rejection techniques, most of them only leverage the temporal consistency of the ego vehicle data by incorporating the previous estimates. However, to the best of the author's knowledge, existing methods typically ignore the temporal consistency of surrounding moving objects, which can be obtained from the MEOT task. With prior tracking information, the positions, shapes, and sizes of moving objects can be determined, making it possible to exclude their negative effects in ego-motion estimation. This could further mitigate the challenges caused by large moving objects and enhance the accuracy of static target detection. Therefore, this thesis aims to develop a radar-only framework that integrates MEOT into ego-motion estimation, thereby improving the robustness of ego-motion estimation in outlier-intensive scenarios.

Multiple extended object tracking: Recent advances in MEOT tackle key chal-

lenges such as clutter suppression, data association, and extended object state estimation. Nevertheless, the existing radar-based MEOT systems typically assume externally provided ego-motion information to compensate the motion of radar sensors, which relies on GNSS, IMU, or LiDAR. As a result, the tracking accuracy may degrade when ego-motion estimation is inaccurate or unavailable. Therefore, a key open challenge is to develop radar-only MEOT frameworks that incorporate ego-motion estimation and tracking, particularly in dynamic and cluttered environments.

Combined methods: Although significant progress has been made for Camera and LiDAR-based systems, to the best of the author's knowledge, not much literature is available on automotive radar data for combined ego-motion estimation and MEOT. As discussed in Sections 2.1 and 2.2, most radar-based approaches address the two tasks independently, without exploiting their inherent coupling. Developing a radar-only method is crucial because it enables robust perception in challenging environments, such as low light, fog, rain, or snow, where camera and LiDAR sensors may fail or degrade significantly. Moreover, compared to LiDAR systems, radar is cost-effective, making it attractive for large-scale development.

The lack of radar-specific combined methods can be attributed to several inherent limitations. First, radar data is typically sparse and low-resolution [14], making it difficult to extract stable semantic or geometric features for cross-frame matching. Sparse point clouds generated by radars, with only a few points on pedestrians and a small number on cars, are insufficient for accurately outlining object contours and extracting detailed shape information [62]. Second, radar can generate a high rate of false detections, which directly affect the accuracy of motion segmentation and data association. Due to the characteristics of millimeter-wave propagation, radar measurements are highly susceptible to clutter and noise from various sources, including multi-path propagation, inter-radar interference, and hardware imperfections [24,25]. These effects can degrade the precision and reliability of radar measurements.

Given these limitations, it is challenging to directly adapt existing approaches developed for camera or LiDAR to radar. Instead, it is essential to design dedicated frameworks that consider the unique characteristics of radar data. This thesis addresses the identified research gap by proposing a radar-only framework that performs combined ego-motion estimation and MEOT, leveraging the mutual constraints between the two tasks—without dependence on external localization systems or high-resolution sensors.

Part II

Raw Signal based Combined Method

Some sections of this part are supposed to be published in: S. Yuan, T. Wang, A. Yarovoy and F. Fioranelli, *"Joint ego-motion estimation and multiple object tracking using automotive radar"*, 2025 22nd European Radar Conference (EuRAD).(Submitted). See Appendix A for the full version of this paper.

3 Methodology

This chapter illustrates the proposed raw signal based method for combined ego-motion estimation and multiple object tracking. Section 3.1 formulates the combined problem as a Bayesian filtering task. Section 3.2 provides an overview of the algorithm, which is structured into two main phases: an initialization phase and a combined estimation phase. From Section 3.3 to Section 3.6, the four modules of the proposed method are presented in detail, including signal preprocessing, tracking-aided ego-motion estimation, ego-motion compensation, and multiple object tracking. Finally, Section 3.7 introduces the evaluation metrics used to assess the performance of both ego-motion estimation and multiple object tracking.

3.1 Problem formulation

In this section, the combined problem of ego-motion estimation and multiple object tracking is formulated as a Bayesian filtering problem. As introduced in Section 1.2, the objective is to estimate the motion states of both the ego vehicle and surrounding moving objects over time, based on sequential radar measurements.

At each discrete time step k (corresponding to radar frame k), the system state consists of two parts:

- **Ego vehicle state**:

$$\mathbf{x}_{ego}^{k} = \begin{bmatrix} \mathbf{p}_{ego}^{k} \\ \mathbf{v}_{ego}^{k} \end{bmatrix} \tag{3.1.1}$$

  where $\mathbf{p}_{ego}^{k}$ is the position of the ego-vehicle, and $\mathbf{v}_{ego}^{k}$ is the translational velocity of the ego-vehicle. It is worth noting that in this thesis, rotational velocity is not estimated.

- **Moving object states**: Assume there are M_k moving objects in total, for each object $j \in \{1, \ldots, M_k\}$, its state also includes the position and velocity vectors:

$$\mathbf{x}_{j}^{k} = \begin{bmatrix} \mathbf{p}_{j}^{k} \\ \mathbf{v}_{j}^{k} \end{bmatrix}, \quad j \in \{1, \ldots, M_k\} \tag{3.1.2}$$

  where $\mathbf{p}_{j}^{k}$ is the position of the j-th object, and $\mathbf{v}_{j}^{k}$ is the translational velocity of the j-th object.

Based on the above two equations, the full system state $\mathbf{X}_k$ is defined as:

$$\mathbf{X}_k = \left[\mathbf{x}_{ego}^{k},\ \mathbf{x}_{1}^{k}, \ldots, \mathbf{x}_{M_k}^{k}\right] \tag{3.1.3}$$

At each time step, the radar measurements are represented by $\mathbf{Z}_k$, which can be either raw signals or processed point clouds. These measurements may originate from static or moving objects, and typically encode range, Doppler velocity, and angle information.

The ego vehicle state is assumed to follow a discrete-time motion model as follows:

$$\mathbf{x}_{ego}^k = f_{ego}(\mathbf{x}_{ego}^{k-1}) + \mathbf{w}_{ego}^k, \quad \mathbf{w}_{ego}^k \sim \mathcal{N}(\mathbf{0}, \mathbf{Q}_{ego}^k) \tag{3.1.4}$$

where $f_{ego}(\cdot)$ denotes the motion transition function, and $\mathbf{w}_{ego}^k$ is zero-mean Gaussian process noise with covariance $\mathbf{Q}_{ego}^k$. Similarly, each moving object evolves according to:

$$\mathbf{x}_j^k = f_{obj}^j(\mathbf{x}_j^{k-1}) + \mathbf{w}_j^k, \quad \mathbf{w}_j^k \sim \mathcal{N}(\mathbf{0}, \mathbf{Q}_j^k), \quad j \in \{1, \ldots, M_k\} \tag{3.1.5}$$

where $f_{obj}^j(\cdot)$ denotes the motion transition function of the j-th object, with the zero-mean Gaussian process noise $\mathbf{w}_j^k$ with covariance $\mathbf{Q}_j^k$.

Each radar measurement is modeled as a function of both the ego vehicle state and the object state. The measurement model is defined as:

$$\mathbf{z}_{k,j} = h(\mathbf{x}_{ego}^k, \mathbf{x}_j^k) + \mathbf{n}_{k,j}, \quad \mathbf{n}_{k,j} \sim \mathcal{N}(\mathbf{0}, \mathbf{R}_k) \tag{3.1.6}$$

where $h(\cdot)$ transforms the global position and velocity states into the radar measurement domain (e.g., range, angle, Doppler), and $\mathbf{n}_{k,j}$ is the additive observation noise with covariance $\mathbf{R}_k$.

Given a sequence of radar measurements up to time k, denoted by $\mathcal{Z}_{1:k} = \{\mathbf{Z}_1, \ldots, \mathbf{Z}_k\}$, the goal is to estimate the posterior distribution over all system states:

$$p(\mathbf{X}_k \mid \mathcal{Z}_{1:k}) = p\left(\mathbf{x}_{ego}^k, \{\mathbf{x}_j^k\}_{j=1}^{M_k} \,\middle|\, \mathcal{Z}_{1:k}\right) \tag{3.1.7}$$

This posterior encapsulates the uncertainty in both ego-motion and object states, given noisy sensor observations and uncertain data associations. In practice, the final state estimate $\hat{\mathbf{X}}_k$ is typically obtained by applying a Bayesian inference criterion. A common choice is the Minimum Mean Squared Error (MMSE) estimator, defined as:

$$\hat{\mathbf{X}}_k^{\text{MMSE}} = \mathbf{E}\left[\mathbf{X}_k \mid \mathcal{Z}_{1:k}\right] \tag{3.1.8}$$

Alternatively, the Maximum A Posteriori (MAP) estimate may be used:

$$\hat{\mathbf{X}}_k^{\text{MAP}} = \arg\max_{\mathbf{X}_k} p(\mathbf{X}_k \mid \mathcal{Z}_{1:k}) \tag{3.1.9}$$

If both the motion and measurement models are linear and corrupted by Gaussian noise, the resulting posterior distribution is also Gaussian. In this case, the MMSE estimate is the same as the MAP estimate.

Since the input originates from a single radar sensor in the scenarios considered in this thesis, the estimation problem is inherently under-constrained. To make the problem tractable, an important assumption is that radar measurements originating from static objects in the environment can be reliably distinguished from those corresponding to moving objects. These static measurements serve as environmental references

and provide essential constraints for ego-motion estimation. Therefore, the separation of static and dynamic measurements is a critical component of the combined estimation algorithm.

Although the ideal solution involves joint estimation of the full state variables, this is difficult for a single radar due to limited observability. Therefore, a sequential estimation approach is adopted in this work: ego-motion state $\mathbf{x}_{ego}^k$ is first estimated from static detections, followed by ego-motion compensation, and then moving object states $\mathbf{x}_j^k$ are estimated. This strategy reduces both the computational complexity and the design difficulty of the overall algorithm. However, the proposed method enhances the coupling between ego-motion estimation and object tracking by incorporating the estimated motion states of moving objects from the previous frame $\mathbf{x}_j^{k-1}$, into the ego-motion estimation process at the current frame $\mathbf{x}_{ego}^k$. This temporal interaction serves as a promising step toward achieving joint and simultaneous state estimation in future work.

Finally, this part of the thesis solves the combined problem of ego-motion estimation & multiple object tracking directly by using raw radar signals, which introduces additional challenges. Unlike point clouds, which are intuitive and compact representations, raw radar data contain complex I/Q signals that require more sophisticated signal processing techniques, especially the utilization of phase information. Moreover, the data volume of raw signals is substantially larger, increasing demands on memory and computational resources. These issues have limited the use of raw signals in past work. To address these challenges, this thesis proposes an innovative combined algorithm, which will be overviewed in Section 3.2.

3.2 Algorithm overview

This work proposes a radar-only algorithm for combined ego-motion estimation and multiple object tracking, operating in a frame-by-frame manner. For each frame, the algorithm takes raw radar signals as input and outputs the motion state of both the ego vehicle $\mathbf{x}_{ego}^k$ and the surrounding moving objects $\mathbf{x}_j^k$.

In temporal order, the algorithm is divided into two phases: the initialization phase and the combined estimation phase. Each phase consists of four key modules: signal preprocessing, ego-motion estimation, ego-motion compensation, and multiple object tracking. The primary difference between the two phases lies in the ego-motion estimation module. In the combined estimation phase, a tracking-aided mechanism is incorporated into ego-motion estimation, referred to as "tracking-aided ego-motion estimation", which enhances the robustness and accuracy of the overall estimation. This difference also leads to a variation in the way static and moving measurements are separated. In the initialization phase, the separation relies solely on Doppler and angle information, whereas in the combined estimation phase, it utilizes Doppler, velocity, and position information. The overall processing pipeline is illustrated in Figure 3.2.1.

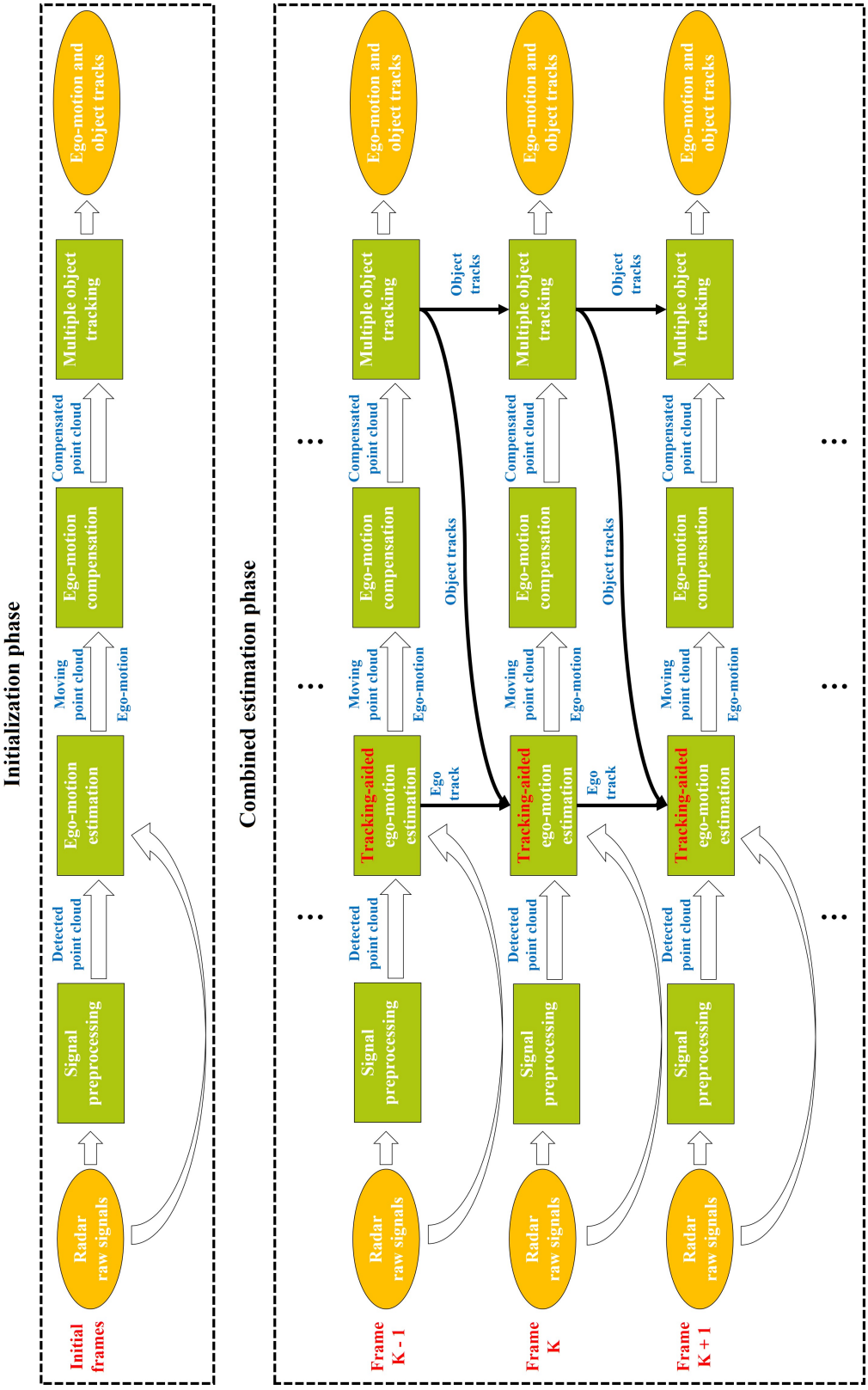


Figure 3.2.1: The overall processing pipeline of the raw signal based combined method for ego-motion estimation & multiple object tracking, with the 2 phases of initialization on N_{init} frames and combined estimation on the generic frame K at steady state.

As the figure shows, the core innovation of the proposed algorithm lies in the combined estimation phase, where the tracking results from the previous frame are utilized to guide the ego-motion estimation in the current frame. This design allows ego-motion estimation and object tracking to benefit from each other in a more integrated manner. By doing so, the negative impact of moving targets on ego-motion estimation is significantly reduced. Meanwhile, ego-motion compensation and multiple object tracking can fully rely on the ego-motion derived from radar data.

To address the state estimation problem described in Section 3.1, a Kalman filter is employed for both the ego vehicle and each moving object to perform state prediction and update. In this thesis, both the ego vehicle and moving targets are assumed to move in 3D space with constant velocity. Therefore, the 3D Constant Velocity (CV) motion model is adopted, which is suitable for steady-state moving cars without brisk acceleration or deceleration. Mathematically, the model can be expressed in Equation 3.2.1:

$$\begin{aligned} \mathbf{x}_{ego}^{k} &= \mathbf{F}\mathbf{x}_{ego}^{k-1} + \mathbf{w}_{ego}^{k} \\ \mathbf{x}_{j}^{k} &= \mathbf{F}\mathbf{x}_{j}^{k-1} + \mathbf{w}_{j}^{k} \end{aligned} \tag{3.2.1}$$

Where $\mathbf{F}$ is the state transition matrix, with T the update time:

$$\mathbf{F} = \begin{bmatrix} 1 & 0 & 0 & T & 0 & 0 \\ 0 & 1 & 0 & 0 & T & 0 \\ 0 & 0 & 1 & 0 & 0 & T \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.2.2}$$

The process noise vectors $\mathbf{w}_{ego}^{k}$ and $\mathbf{w}_{j}^{k}$ represent the process uncertainties for the ego vehicle and the j-th moving object, respectively. Both are modeled as zero-mean Gaussian random variables with identical constant covariance matrix $\mathbf{Q}$, given by:

$$\mathbf{Q} = \sigma_q^2 \begin{bmatrix} \frac{T^4}{4} & 0 & 0 & \frac{T^3}{2} & 0 & 0 \\ 0 & \frac{T^4}{4} & 0 & 0 & \frac{T^3}{2} & 0 \\ 0 & 0 & \frac{T^4}{4} & 0 & 0 & \frac{T^3}{2} \\ \frac{T^3}{2} & 0 & 0 & T^2 & 0 & 0 \\ 0 & \frac{T^3}{2} & 0 & 0 & T^2 & 0 \\ 0 & 0 & \frac{T^3}{2} & 0 & 0 & T^2 \end{bmatrix} \tag{3.2.3}$$

In this chapter, the update time T is set to 10.24 milliseconds, which corresponds to the radar frame interval in the implemented simulation framework in Chapter 4. This value can be adjusted depending on the specific radar configuration, and so the method discussed in this chapter remains valid even in those cases. The process noise variance σ_q^2 is set to 3 based on empirical tuning.

In terms of the measurement model, for the ego vehicle, the 3D CV model is adopted. Since the velocity is directly observed through ego-motion estimation, it is used as the measurement. Mathematically, the model can be expressed in Equation 3.2.4:

$$\mathbf{z}_{ego}^{k} = \mathbf{H}_{\mathbf{ego}}\mathbf{x}_{ego}^{k} + \mathbf{v}_{ego}^{k} \tag{3.2.4}$$

Where $\mathbf{H_{ego}}$ is the measurement matrix for the ego vehicle:

$$\mathbf{H_{ego}} = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.2.5}$$

$\mathbf{v}_{ego}^k$ is the measurement noise vector, a Gaussian variable with zero mean and covariance matrix $\mathbf{R}_{ego} = \sigma_{r,ego}^2 \times \mathbf{I}_{3\times3}$. The measurement noise variance $\sigma_{r,ego}^2$ is set to 0.2 through parameter tuning.

For moving objects, the state estimation relies on position measurements. Therefore, the 3D CV measurement model becomes:

$$\mathbf{z}_j^k = \mathbf{H_{obj}}\mathbf{x}_j^k + \mathbf{v}_j^k \tag{3.2.6}$$

Where $\mathbf{H_{obj}}$ is the measurement matrix for moving objects:

$$\mathbf{H_{obj}} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix} \tag{3.2.7}$$

$\mathbf{v}_j^k$ is the measurement noise vector, a Gaussian variable with zero mean and covariance matrix $\mathbf{R}_{obj} = \sigma_{r,obj}^2 \times \mathbf{I}_{3\times3}$. The measurement noise variance $\sigma_{r,obj}^2$ is set to 1 through parameter tuning.

Next, the initialization phase and the combined estimation phase are described in detail in Sections 3.2.1 and 3.2.2, respectively.

3.2.1 Initialization phase

Since ego-motion estimation depends on the tracking results from the previous frame, the algorithm requires an initialization phase to bootstrap the tracker before entering the combined estimation phase. During this phase, which processes the first N_{init} frames, no prior knowledge about the motion of the ego vehicle or surrounding objects is assumed. In particular, ego-motion estimation is performed without relying on any tracking feedback. The algorithm assumes that during the first frames, the environment contains primarily static objects, and the influence of moving targets is negligible. Under this assumption, ego-motion estimation and object tracking can be performed independently and still yield reliable results.

As illustrated in Figure 3.2.1, for each frame, the pipeline consists of the following four modules, executed sequentially as follows:

1. **Signal preprocessing**
   For each radar frame, the raw signals are first processed using 2D FFT and a Cell-Averaging Constant False Alarm Rate (CA-CFAR) detector [63] to extract detections in the range-Doppler domain. Angle information is subsequently estimated via an optimization-based approach [6]. The resulting detections serve as inputs to the downstream estimation modules.

2. **Ego-motion estimation**
   By using a signal-based iterative optimization method [36], the static and moving measurements are separated, and the ego vehicle's translational velocities are estimated with all static measurements. This velocity estimate is used to initialize a Kalman filter based on a constant-velocity motion model, enabling recursive prediction and correction of the ego vehicle state across frames.

3. **Ego-motion compensation**
   Points that significantly deviate from the expected Doppler profile of a static scene are identified as candidate moving points. These points are compensated for ego-motion and integrated over several frames to accumulate a dense point cloud representing dynamic targets.

4. **Multiple object tracking**
   A DBSCAN clustering algorithm [41] is applied first to the compensated point cloud to generate object-level measurements. These measurements are then used to initialize a multi-object tracker based on a Gaussian mixture implementation of the Global Nearest Neighbor algorithm [52], with Kalman filtering used to estimate the state of each object also under the constant-velocity model.

The covariance matrices of both the ego-vehicle and moving-object Kalman filters are initialized with large values to account for the high initial uncertainty, and are progressively reduced as the tracks are initialized and confirmed. After a sufficient number of frames, which is set in this work to $N_{\text{init}} = 10$ (i.e., 0.1 s in terms of time) according to empirical verifications, the tracker reaches a stable state, allowing the system to transition to the combined estimation phase, where ego-motion estimation and multiple object tracking interact in a tightly coupled manner.

3.2.2 Combined estimation phase

Following the initialization phase, the algorithm transitions into the combined estimation phase, which is applied to all subsequent frames. While maintaining the same four-module pipeline, the main improvement lies in ego-motion estimation, where a tracking-aided approach leverages previous tracking results to better separate static and dynamic points, improving robustness in dynamic scenes.

As illustrated in Figure 3.2.1, for each frame K, the following steps are performed:

1. **Signal preprocessing**
   Identical to the initialization phase, radar raw signals undergo 2D FFT processing, followed by CA-CFAR detection and angle estimation to extract target detections. This module is detailed in Section 3.3.

2. **Tracking-aided ego-motion estimation**
   In addition to the ego-motion estimation method utilized in the initialization phase, which will be detailed in Section 3.4.2, this module incorporates a tracking-aided segmentation step to identify and exclude dynamic targets, as described in Section 3.4.1. The separation of static and moving measurements is jointly accomplished by the processes in Sections 3.4.1 and 3.4.2: in Section 3.4.1, position

information is used to eliminate obvious dynamic measurements, while in Section 3.4.2, the remaining measurements are further classified based on Doppler/velocity and angle information. The final set of static points is then used to estimate the ego vehicle's translational motion. A Kalman filter is employed to recursively fuse the estimates over time, ensuring smooth and consistent ego-motion trajectories.

3. **Ego-motion compensation**
   Similar to the initialization phase, points identified as moving are compensated based on the estimated ego-motion to remove the influence of the ego vehicle's motion and align them across frames. This step is further described in Section 3.5.

4. **Multiple object tracking**
   Consistent with the initialization phase, the compensated point cloud is clustered using DBSCAN to remove clutter and generate object-level measurements. These measurements are then associated with existing tracks using the GNN algorithm, and states are updated with the Kalman filter. Details are provided in Section 3.6.

3.3 Signal preprocessing

This section describes the signal preprocessing module that transforms raw signals into detected point clouds with range, Doppler, and angle information. These detected point clouds serve as partial input to downstream modules.

When using an FMCW MIMO automotive radar system, the radar raw signal refers to the In-phase and Quadrature (I/Q) Intermediate-Frequency (IF) signals directly obtained from each receiver channel after the Analog-to-Digital Converter (ADC) and before the FFT signal processing. As shown in Figure 3.3.1, the transmit antenna emits a frequency-modulated chirp signal generated by a synthesizer, and the reflected signal is captured by the receive antenna. This received signal is mixed with a copy of the transmitted signal to generate the IF signal, which encodes range and Doppler information. After passing through a low-pass filter and an ADC, the complex I/Q samples are digitized. These digitized signals are referred to as the 'raw signals', and they serve as the input to the proposed method. In a MIMO radar system, where multiple transmit and receive antennas are employed, the I/Q samples are collected independently from each receive channel and subsequently combined to form a multi-channel data cube for further processing. If there are I receive channels in the azimuth dimension and J channels in the elevation dimension, with each chirp containing M fast-time samples and each frame consisting of N chirps, the resulting four-dimensional complex data cube can be represented as $z(I, J, M, N)$.

In a radar signal processing pipeline, the first step is to convert the raw signals from the time domain to the frequency domain with the FFT. Since ego-motion estimation relies on the spatial alignment between different chirp groups, two chirp groups u_0 and u_1, one from chirp u_0 to chirp $\frac{N}{2} + u_0$, and the other from chirp u_1 to chirp $\frac{N}{2} + u_1$, are extracted. The original four-dimensional data cube $z(I, J, M, N)$ is therefore split into

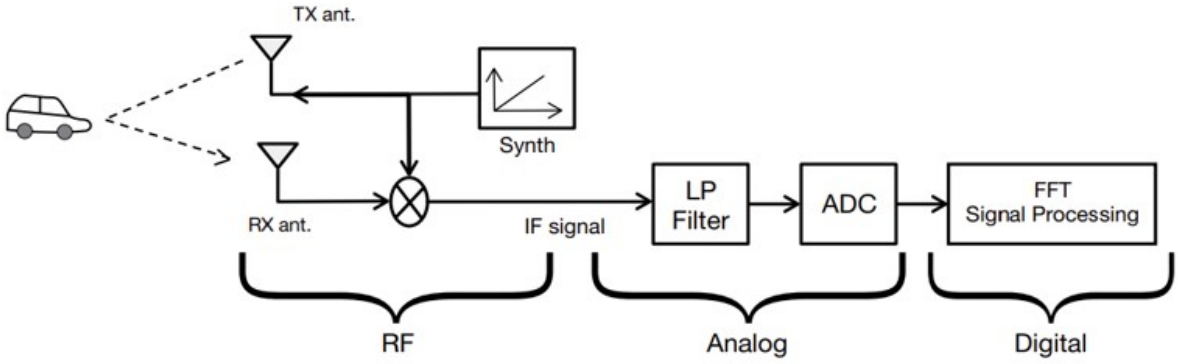


Figure 3.3.1: Hardware high-level block diagram of an FMCW MIMO automotive radar [64]

two subsets as follows:

$$z(I, J, M, N) \longrightarrow \left\{ z_{u_0}(I, J, M, \tfrac{N}{2}),\ z_{u_1}(I, J, M, \tfrac{N}{2}) \right\} \tag{3.3.1}$$

For each chirp group and each antenna channel, a M-point FFT is first applied along the fast-time dimension to obtain the range profile, followed by a $\frac{N}{2}$-point FFT along the slow-time dimension to extract the Doppler information. This two-dimensional FFT yields the Range-Doppler Spectrum (RDS) Z, computed as:

$$Z_{u_0}(I, J, M, \tfrac{N}{2}) = \text{2D-FFT}\big(z_{u_0}(I, J, M, \tfrac{N}{2})\big) \tag{3.3.2}$$

$$Z_{u_1}(I, J, M, \tfrac{N}{2}) = \text{2D-FFT}\big(z_{u_1}(I, J, M, \tfrac{N}{2})\big) \tag{3.3.3}$$

For each chirp group and each antenna channel, the RDS is a matrix of range-Doppler bins, where each bin $[m, n]$ is identified by a range index $m \in \{0, 1, \ldots, M-1\}$ and a Doppler index $n \in \{0, 1, \ldots, \frac{N}{2}-1\}$. Each bin corresponds to a specific combination of range and Doppler values, and its complex amplitude reflects the signal response from targets at the associated distance and relative velocity. These bins provide the basis for subsequent target detection and angle estimation procedures.

Next, a target detection algorithm is employed to determine which range-Doppler bins contain target reflections. The two-dimensional CA-CFAR [63] is implemented on the RDS to detect targets and reduce false alarms. The method works by dynamically adjusting the detection threshold based on an estimate of the background noise. As illustrated in Figure 3.3.2, for each Cell Under Test (CUT), two types of cell are defined: the guard cells and training cells. Training cells are the surrounding cells used to estimate the background noise level. In CA-CFAR, these cells are averaged to get the noise estimate. To ensure the noise estimate is not biased by the target signal, guard cells are introduced immediately adjacent to the CUT. These guard cells are excluded from the noise estimation to prevent interference from the target's energy or other strong signals nearby. This separation ensures accurate threshold calculation and reliable target detection. The noise estimate from the reference cells is scaled by a factor corresponding to the desired false alarm rate, and this scaled value is used as

the detection threshold. If the signal in the CUT exceeds the threshold, a target is detected.

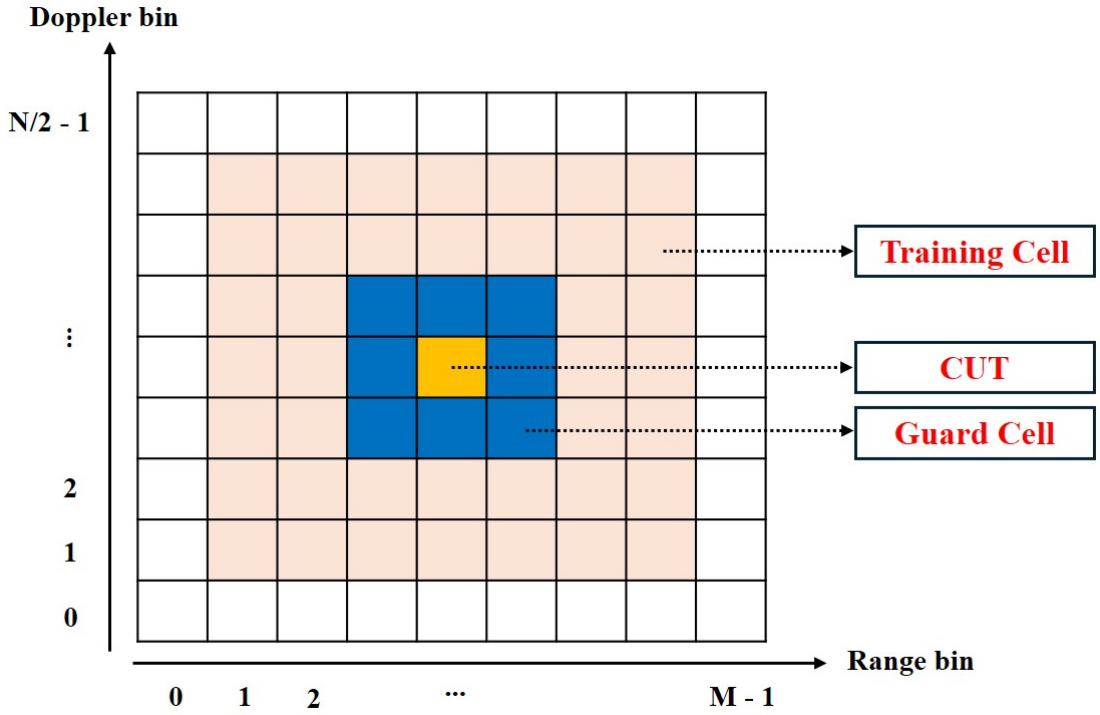


Figure 3.3.2: Concept of 2D CA-CFAR detector applied on a range-Doppler matrix

In the specific implementation, the magnitude of RDS is taken, accumulated across all antennas, and normalized. The number of training cells is set to 2 for both dimensions, and the number of guard cells is set to 1 for both dimension. The scaling factor is set to 5. With these empirically-selected parameters, the detector can make a good balance between robust detection of targets and a low false alarm rate.

After finding the range-Doppler bins corresponding to detected targets, the algorithm adopts an optimization approach to estimate the azimuth and elevation angles of each bin. Assume that inside one short radar frame (equivalent to approximately 0.01 s), a target will be in the same range-Doppler bin. Moreover, since the spacing between antenna elements is very small compared to the radar-to-target distance, we can also assume a target will be in the same range-Doppler bin for different antennas. Then at the same slow time, for different antennas, the phase difference between the RDS contains the angle information of targets.

Assume the k-th target scatter o_k exists in the range-Doppler bin $[m_k, n_k]$, and u_0 is the starting index of the first chirp group, the relationship between the RDS of the antenna element $[i, j]$ (i.e., the i^{th} element in the azimuth dimension and the j^{th} element in the elevation dimension) $Z_{u_0}(i, j, m_k, n_k)$ and the RDS of the reference antenna element $Z_{u_0}(1, 1, m_k, n_k)$ can be written in Equation 3.3.4:

$$Z_{u_0}(i, j, m_k, n_k) \propto Z_{u_0}(1, 1, m_k, n_k) \exp\left[j\Phi(o_k, i, j, u_0)\right] \tag{3.3.4}$$

Essentially, these RDSs only differ in a phase difference $\Phi(o_k, i, j, u_0)$ defined in Equation 3.3.5. The phase difference depends on the azimuth angle of target $\hat{\theta}_{o_k}(u_0)$, the elevation angle of target $\hat{\phi}_{o_k}(u_0)$, the spacing between adjacent antennas in both azimuth dimension and elevation dimension d, and the radar center frequency f_0:

$$\Phi(o_k, i, j, u_0) = 2\pi f_0 \left(\frac{id}{c} \sin \hat{\theta}_{o_k}(u_0) \cos \hat{\phi}_{o_k}(u_0) + \frac{jd}{c} \sin \hat{\phi}_{o_k}(u_0) \right) \tag{3.3.5}$$

Based on the above equations, the azimuth angle $\hat{\theta}_{o_k}(u_0)$ and elevation angle $\hat{\phi}_{o_k}(u_0)$ of the detected targets can be estimated from the phase difference in the presence of noise. This task can be formulated as an optimization problem which fits the measurement data to a known signal model with unknown parameters.

From the Equation 3.3.4, the measured data in a matrix form $\mathbf{X} \in \mathbf{C}^{N_a \times N_e}$ can be obtained:

$$\mathbf{X} = \frac{Z_{u_0}(i, j, m_k, n_k)}{Z_{u_0}(1, 1, m_k, n_k)} + \mathbf{N_x} \tag{3.3.6}$$

where N_a and N_e denote the number of antenna elements along the azimuth and elevation dimensions, respectively, with $i \in \{1, \ldots, N_a\}$ and $j \in \{1, \ldots, N_e\}$. The ratio is computed element-wise across all antenna positions. $\mathbf{N_x} \in \mathbf{C}^{N_a \times N_e}$ is the complex-valued noise matrix.

The signal model $\mathbf{Y} \in \mathbf{C}^{N_a \times N_e}$, which is a function of the azimuth angle $\hat{\theta}_{o_k}(u_0)$ and elevation angle $\hat{\phi}_{o_k}(u_0)$, can be represented as:

$$\mathbf{Y}(\hat{\theta}_{o_k}(u_0), \hat{\phi}_{o_k}(u_0)) = \exp\left(j2\pi f_0 \left(\frac{id}{c} \sin \hat{\theta}_{o_k}(u_0) \cos \hat{\phi}_{o_k}(u_0) + \frac{jd}{c} \sin \hat{\phi}_{o_k}(u_0) \right) \right) \tag{3.3.7}$$

The problem of angle estimation then becomes an unconstrained optimization problem as in 3.3.8, with the objective defined in Equation 3.3.9.

$$\arg \min_{\hat{\theta}_{o_k}(u_0), \hat{\phi}_{o_k}(u_0)} f(\hat{\theta}_{o_k}(u_0), \hat{\phi}_{o_k}(u_0)) \tag{3.3.8}$$

where

$$f(\hat{\theta}_{o_k}(u_0), \hat{\phi}_{o_k}(u_0)) = [\mathbf{Y} - \mathbf{X}]^H [\mathbf{Y} - \mathbf{X}], \quad \text{with } \mathbf{X}, \mathbf{Y} \in \mathbf{C}^{N_a \times N_e}. \tag{3.3.9}$$

This optimization problem can be solved with the pattern search [65] algorithm in the Matlab Global Optimization toolbox [66]. This algorithm does not require the calculation of gradient, making it suitable for functions that are noisy, discontinuous, or non-differentiable. It is also robust to non-convex functions and complex search spaces. After solving the optimization problem, the azimuth and elevation angles of each detected target o_k are obtained. Combined with the range and Doppler values corresponding to the range-Doppler bin $[m_k, n_k]$, the range, velocity, and angle information for each target is all estimated.

In summary, the signal preprocessing module processes raw radar signals through a sequence of operations including chirp group separation, two-dimensional FFT, CA-CFAR-based target detection, and angle estimation. Assuming a total of N targets are

detected, the output is a detected point cloud including the four values of estimated range, Doppler velocity, azimuth angle, and elevation angle of all targets. The point cloud $\mathcal{P}$ is formally represented as:

$$\mathcal{P} = \{(\hat{R}_i, \hat{V}_{D,i}, \hat{\theta}_i, \hat{\phi}_i)\}_{i=1}^{N} \tag{3.3.10}$$

3.4 Tracking-aided ego-motion estimation

After signal preprocessing, the tracking-aided ego-motion estimation module utilizes both the raw signals and the detected point cloud to estimate the ego-motion. The point cloud is used to identify the range-Doppler bins associated with static objects. These bins are then referenced back to the raw signal domain, where a robust ego-motion estimation is performed using only the static objects. This module consists of three key steps. As described in Section 3.4.1, previous tracking results are employed to guide the segmentation of static and moving objects. In Section 3.4.2, an iterative algorithm estimates the ego-motion directly from the raw signals, using only the previously identified static objects bins. Finally, Section 3.4.3 converts the estimated ego-velocity of the radar to the ego-velocity of the vehicle, and updates the Kalman filter for the ego vehicle using the estimated velocity. It is worth noting that during the initialization phase, when tracking information is not yet available, only the steps in Section 3.4.2 and Section 3.4.3 are executed.

3.4.1 Tracking-aided segmentation

To ensure robust ego-motion estimation in dynamic environments, it is essential to distinguish static points from those belonging to moving objects. Tracking-aided segmentation is the first step of the ego-motion estimation. During the segmentation, the range-Doppler bins associated with moving objects are excluded and passed to the subsequent tracking module, while those remaining bins are retained for iterative ego-motion estimation detailed in Section 3.4.2.

This segmentation strategy is based on the assumption that objects that were previously tracked as moving are likely to continue moving in the current frame. Moreover, their positions can be reliably predicted using the state estimates from the tracking module. This assumption leverages the temporal continuity of object motion, which is commonly observed in real-world driving scenarios. By exploiting the predicted motion of tracked objects, the algorithm is able to proactively remove potential moving points from the radar point cloud, thereby reducing their interference with the ego-motion estimation process.

After the initialization phase, the system obtains the tracks of both the ego vehicle and moving objects. Each track contains position and velocity of the moving entity. During the combined estimation phase, for each radar frame, the input to the segmentation module consists of:

- the current radar point cloud
- the ego vehicle track from the previous frame

- the moving object tracks from the previous frame

The output of the module is the point cloud segmentation result, which includes:

- the indices of points classified as static
- the indices of points classified as moving

The segmentation algorithm consists of three main steps: **prediction**, **compensation**, and **gating**, as detailed in Algorithm 1 and following text.

Algorithm 1 Tracking-Aided Segmentation

Require: Detected point cloud at the current frame $\mathcal{P} = \{(\hat{R}_i, \hat{V}_{D,i}, \hat{\theta}_i, \hat{\phi}_i)\}_{i=1}^{N}$; ego vehicle track from the previous frame $\{\hat{\mathbf{x}}_{ego}, \mathbf{P}_{ego}\}$; object tracks from the previous frame $\{\hat{\mathbf{x}}_j, \mathbf{P}_j\}_{j=1}^{M}$; the rotation matrix $\mathbf{R}_{\text{r2v}}$ and the translation vector $\mathbf{t}_{\text{r2v}}$ from radar to vehicle coordinate system; the rotation matrix $\mathbf{R}_{\text{v2w}}$ from vehicle to world coordinate system

Ensure: Static point indices $\mathcal{I}_{\text{static}}$, moving point indices $\mathcal{I}_{\text{moving}}$

Step 1: Prediction

Predict the ego vehicle state $\hat{\mathbf{x}}_{ego}^{-}$ and covariance matrix $\mathbf{P}_{ego}^{-}$ using Kalman filter
Extract predicted position of the ego vehicle $\mathbf{x}_{ego} = \hat{\mathbf{x}}_{ego}^{-}(1{:}3)$
for each object track $j = 1$ to M **do**
  Predict object state $\hat{\mathbf{x}}_j^{-}$ and covariance matrix $\mathbf{P}_j^{-}$ using Kalman filter
  Extract predicted position $\boldsymbol{\mu}_j = \hat{\mathbf{x}}_j^{-}(1{:}3)$
end for

Step 2: Compensation

Convert point cloud $\mathcal{P}$ to Cartesian coordinates $\mathbf{p}_{ir} = (x_i, y_i, z_i)$
Switch from the radar coordinate to the vehicle coordinate: $\mathbf{p}_{iv} \leftarrow \mathbf{R}_{\text{r2v}} \cdot \mathbf{p}_{ir} + \mathbf{t}_{\text{r2v}}$
Apply ego-motion compensation: $\mathbf{p}_i \leftarrow \mathbf{R}_{\text{v2w}} \cdot \mathbf{p}_{iv} + \mathbf{x}_{ego}$

Step 3: Gating

Initialize $\mathcal{I}_{\text{static}} \leftarrow \{1, 2, \ldots, N\}$
for each point $\mathbf{p}_i$ **do**
  for each object $j = 1$ to M **do**
    Compute Euclidean distance $d^2 = (\mathbf{p}_i - \boldsymbol{\mu}_j)^\top (\mathbf{p}_i - \boldsymbol{\mu}_j)$
    if $d^2 < \gamma$ **then**
      Point $\mathbf{p}_i$ is inside the gating region
      Mark point $\mathbf{p}_i$ as moving: remove i from $\mathcal{I}_{\text{static}}$
      break
    end if
  end for
end for
$\mathcal{I}_{\text{moving}} \leftarrow$ indices not in $\mathcal{I}_{\text{static}}$
return $\mathcal{I}_{\text{static}}, \mathcal{I}_{\text{moving}}$

Step 1: Prediction This step aims to predict the spatial relationship between the ego vehicle and moving objects based on their previous states and motion models. As described in Section 3.2, both the ego vehicle and each tracked object are modeled using

Kalman filters. Therefore, it is straightforward and convenient to apply the Kalman prediction step to solve the task. In this method, a 3D CV motion model is selected for all entities. The prediction of the state and covariance is computed as in Equation 3.4.1 and 3.4.2:

$$\hat{\mathbf{x}}_{k|k-1}^{-} = \mathbf{F}\hat{\mathbf{x}}_{k-1} \tag{3.4.1}$$

$$\mathbf{P}_{k|k-1}^{-} = \mathbf{F}\mathbf{P}_{k-1}\mathbf{F}^{\top} + \mathbf{Q} \tag{3.4.2}$$

Where $\hat{\mathbf{x}}_{k|k-1}^{-}$ and $\mathbf{P}_{k|k-1}^{-}$ are the predicted state and covariance at time step k, and $\hat{\mathbf{x}}_{k-1}$ and $\mathbf{P}_{k-1}$ are the estimated state and covariance at time step $k-1$.

Step 2: Compensation This step compensates for ego-motion, in order to remove its influence on the observed radar point cloud. The compensation method is the same as described later in Section 3.5, with the only difference being that it uses the predicted vehicle position instead of the updated one. More details on this compensation method can be found in Section 3.5.

After this step, all detected points are transformed to the world coordinate system, enabling correct comparison with predicted object positions.

Step 3: Gating The objective of this step is to classify radar detections as either moving or static by evaluating their spatial proximity to previously tracked moving objects. To accomplish this, we adopt a Euclidean gating strategy, a technique commonly used in Multiple Object Tracking algorithms for data association [10,38]. In this work, Euclidean gating is adapted to address the point cloud segmentation problem, enabling the separation of dynamic and static measurements prior to ego-motion estimation.

In the context of tracking, gating is a method used to constrain the number of feasible associations between predicted object states and new detections. It defines a bounded region around each predicted track, and only detections falling within this region are considered as viable association candidates. This effectively reduces computational complexity and improves the robustness of data association, especially in cluttered environments.

Here, the same principle is applied to segment the radar point cloud. For each detection $\mathbf{p}_i$, the Euclidean distance to the predicted center $\boldsymbol{\mu}_j$ of each tracked moving object is computed. Assume there are M tracks, if the distance between the detection and any track is smaller than a threshold, the point is considered as belonging to a moving object. Otherwise, it is labeled as static as:

$$\begin{cases} \text{Label moving target} & \text{if } \min_{j\in 1,\dots,M}(\mathbf{p}_i - \boldsymbol{\mu}_j)^{\top}(\mathbf{p}_i - \boldsymbol{\mu}_j) < \gamma \\ \text{Label static target} & \text{if } \min_{j\in 1,\dots,M}(\mathbf{p}_i - \boldsymbol{\mu}_j)^{\top}(\mathbf{p}_i - \boldsymbol{\mu}_j) \geq \gamma \end{cases} \tag{3.4.3}$$

The threshold radius γ defines the spatial extent of the gate, and is empirically set to 4 meters in this implementation. This gating-based segmentation effectively filters out candidate moving points from the ego-motion estimation process, while preserving them for use in the subsequent tracking module. In Chapter 5, this gating strategy is further extended to adaptively consider the spatial extent of each object, enabling more accurate segmentation in the presence of size variations.

3.4.2 Iterative ego-motion estimation

Based on the segmentation result from Section 3.4.1, a subset of radar detections is classified as static. These static indices $\mathcal{I}_{\text{static}}$ are then used to identify the corresponding range-Doppler bins in the raw signal domain. To be specific, for each static index $k \in \mathcal{I}_{\text{static}}$, the estimated range $\hat{R}_k$ and Doppler values $\hat{V}_{D,k}$ are mapped back to its range-Doppler bin $[m_k, n_k]$. Only the raw signals in these bins are extracted and used to perform a robust ego-motion estimation. For the initialization phase, since no segmentation is performed, all range-Doppler bins associated with detected targets are used in ego-motion estimation.

The major principle of ego-motion estimation with radar is to utilize measurements from static targets within the field of view and estimate the motion of the radar platform in reverse, i.e., based on the indirect motion of the static targets. Although the tracking information in Section 3.4.1 helps exclude moving targets, the remaining static range-Doppler bins may still contain detections of clutter or newly-appeared moving objects. To ensure a more reliable segmentation between moving and static objects, and to perform ego-motion estimation using only strictly static targets, a robust method is needed.

In this section, the iterative ego-motion estimation method proposed in [36] is adopted and implemented. The method iteratively recognizes static targets and uses the phase difference at different chirp groups inside one frame to estimate the ego-velocity.

This method contains two main processing steps: initial ego-motion estimation and iterative ego-motion estimation. In the initial ego-motion estimation step, an initial rough ego-velocity estimation based on optimization is conducted based on the information of all static range-Doppler bins from Section 3.4.1. After that, in the iterative ego-motion estimation step, the algorithm iteratively labels the range-Doppler bin based on the results of the last ego-motion estimation, and uses the information of all static range-Doppler bins to perform a new estimation. After several iterations, the results converge and the final estimated ego-velocity is obtained. Moreover, the algorithm also outputs the moving point cloud for the subsequent tracking task.

The principle of the initial ego-motion estimation is similar to the angle estimation approach in Section 3.3, as both methods optimize based on phase differences from different parts of the raw signals. This method relies on the fact that for the same antenna, at different slow time, the phase difference between the RDS contains the ego-velocity information of the vehicle. The relationship between the RDS of the first chirp group $Z_{u_0}(i, j, m_k, n_k)$ and the RDS of the second chirp group $Z_{u_1}(i, j, m_k, n_k)$ can be written in Equation 3.4.4:

$$Z_{u_1}(i, j, m_k, n_k) \propto Z_{u_0}(i, j, m_k, n_k) \exp\left[j\Gamma(o_k, u_1, u_0)\right] \tag{3.4.4}$$

Similar to Equation 3.3.4, these RDSs only differ in a phase difference $\Gamma(o_k, u_1, u_0)$ detailed in Equation 3.4.5, which depends on the relative velocity between the radar and the targets $\hat{v}_{rx}, \hat{v}_{ry}, \hat{v}_{rz}$, the time difference $(u_1 - u_0)T$ (with T denoting the Pulse Repetition Interval (PRI)), the azimuth angle of target $\hat{\theta}_{o_k}(u_0)$, the elevation angle of target $\hat{\phi}_{o_k}(u_0)$, and the center frequency f_0:

$$\Gamma(o_k, u_1, u_0) = 4\pi \frac{d_{r_{o_k}} f_0}{c} \tag{3.4.5}$$

where $d_{r_{o_k}}$ denotes the displacement of target o_k projected onto its line of sight, and is given by:

$$\begin{aligned} d_{r_{o_k}} &= \hat{v}_{rx} \cdot (u_1 - u_0) \cos\hat{\theta}_{o_k}(u_0) \cos\hat{\phi}_{o_k}(u_0) T \\ &+ \hat{v}_{ry} \cdot (u_1 - u_0) \sin\hat{\theta}_{o_k}(u_0) \cos\hat{\phi}_{o_k}(u_0) T \\ &+ \hat{v}_{rz} \cdot (u_1 - u_0) \sin\hat{\phi}_{o_k}(u_0) T \end{aligned} \tag{3.4.6}$$

The relative velocity $\hat{v}_{rx}, \hat{v}_{ry}, \hat{v}_{rz}$ can also be estimated by solving an optimization problem. The measured data can be obtained and written as the measurement matrix $\mathbf{P} \in \mathbf{C}^{N_k \times N_e N_a}$ in Equation 3.4.7. Only the static range-Doppler bins $[m_k, n_k]$ are used to construct the measurement matrix, where N_k denotes the number of static bins. The ratio is computed for each antenna positions, where N_a and N_e denote the number of antenna elements along the azimuth and elevation dimensions, respectively, with $i \in \{1, \ldots, N_a\}$ and $j \in \{1, \ldots, N_e\}$.

$$\mathbf{P} = \frac{Z_{u_1}(i, j, m_k, n_k)}{Z_{u_0}(i, j, m_k, n_k)} + \mathbf{N_p} \tag{3.4.7}$$

where $\mathbf{N_p} \in \mathbf{C}^{N_k \times N_e N_a}$ is the complex-valued noise matrix.

The signal model is designed as $\mathbf{Q} \in \mathbf{C}^{N_k \times N_e N_a}$:

$$\begin{aligned} \mathbf{Q}(\hat{v}_{rx}, \hat{v}_{ry}, \hat{v}_{rz}) = \exp\Bigg(j\, 4\pi \frac{f_0 (u_1 - u_0) T}{c} \Big(& \hat{v}_{rx} \cos\hat{\theta}_{o_k} \cos\hat{\phi}_{o_k} \\ + \hat{v}_{ry} \sin\hat{\theta}_{o_k} \cos\hat{\phi}_{o_k} \quad & + \hat{v}_{rz} \sin\hat{\phi}_{o_k} \Big) \Bigg) \cdot \mathrm{ones}(1, N_e N_a) \end{aligned} \tag{3.4.8}$$

The problem of estimating the relative velocity $\hat{v}_{rx}, \hat{v}_{ry}, \hat{v}_{rz}$ becomes another optimization problem as in 3.4.9, with the objective function defined in Equation 3.4.10:

$$\arg \min_{\hat{v}_{rx}, \hat{v}_{ry}, \hat{v}_{rz}} f(\hat{v}_{rx}, \hat{v}_{ry}, \hat{v}_{rz}) \tag{3.4.9}$$

where

$$f(\hat{v}_{rx}, \hat{v}_{ry}, \hat{v}_{rz}) = [\mathbf{Q} - \mathbf{P}]^H [\mathbf{Q} - \mathbf{P}] \tag{3.4.10}$$

This problem is also solved with pattern search [65] to estimate $\hat{v}_{rx}, \hat{v}_{ry}, \hat{v}_{rz}$.

After the initial estimation is completed, the algorithm enters a two-step feedback loop: divide targets into static or moving categories, and perform ego-motion estimation again with the above method based only on static targets. After several iterations, a given stopping criterion is met and the final estimated ego-velocity is obtained.

Assume the vector $[\hat{v}^i_{rx}, \hat{v}^i_{ry}, \hat{v}^i_{rz}]$ is the velocity estimation results from the i-th iteration. In the (i+1)-th iteration, for the k-th target, the ego-motion induced Doppler can be calculated as:

$$V^{i+1}_{induced} = \hat{v}^i_{rx} \cos\hat{\theta}_{o_k} \cos\hat{\phi}_{o_k} + \hat{v}^i_{ry} \sin\hat{\theta}_{o_k} \cos\hat{\phi}_{o_k} + \hat{v}^i_{rz} \sin\hat{\phi}_{o_k} \tag{3.4.11}$$

If the target is static, this ego-motion induced Doppler should be close to the measured Doppler. By obtaining the measured Doppler $V(m_k, n_k)$ from the RDS detections,

the difference between the ego-motion induced Doppler and the measured Doppler can be calculated. If this velocity difference is smaller than a given threshold, the target is labelled as a static target. Otherwise, it is labelled as a moving target:

$$\begin{cases} \text{Label moving target} & \text{if } \left|V(m_k, n_k) - V_{induced}^{i+1}\right| > \text{Threshold}^{i+1} \\ \text{Label static target} & \text{if } \left|V(m_k, n_k) - V_{induced}^{i+1}\right| \leq \text{Threshold}^{i+1} \end{cases} \tag{3.4.12}$$

In this work, the threshold is set to the mean of velocity difference for all static targets from the last iteration, as:

$$\text{Threshold}^{i+1} = \text{mean}(\left|V(m_k, n_k) - V_{induced}^{i}\right|), \quad \text{for all } \left|V(m_k, n_k) - V_{induced}^{i}\right| \leq \text{Threshold}^{i} \tag{3.4.13}$$

In this way, a new list of static targets is obtained. Using these $[m_k, n_k]$, the measurement matrix $\mathbf{P}$ in Equation 3.4.7 can be formed and the optimization problem in Equation 3.4.9 can be solved. The results are the velocity estimation results from the (i+1)-th iteration: $[\hat{v}_{rx}^{i+1}, \hat{v}_{ry}^{i+1}, \hat{v}_{rz}^{i+1}]$.

After several iterations, the list of static targets will remain unchanged and the loop can be exited. The criterion for the breaking point is that the velocity difference for all static targets should be smaller than the radar velocity resolution:

$$\max(|V(m_k, n_k) - V_{induced}|) < \Delta v \tag{3.4.14}$$

The velocity resolution is given by:

$$\Delta v = \frac{\lambda}{2TN_{Doppler}} \tag{3.4.15}$$

where λ is the wavelength, T is the PRI, and $N_{Doppler}$ denotes the number of chirps used for Doppler processing. In this method, N_{Doppler} equals to $\frac{N}{2}$.

Once the iteration ends, the relative velocity estimation results $[\hat{v}_{rx}, \hat{v}_{ry}, \hat{v}_{rz}]$ becomes the final ego-velocity estimation results $[\hat{v}_x^{ego,radar}, \hat{v}_y^{ego,radar}, \hat{v}_z^{ego,radar}]$.

3.4.3 Velocity transformation and state update

The velocity estimated in Section 3.4.2 corresponds to the motion of the radar sensor. However, since the radar is rigidly mounted on the vehicle with a certain angle and position offset, it is necessary to convert the estimated radar velocity into the ego vehicle's velocity in the vehicle coordinate system. This requires a coordinate transformation between the radar observation coordinate system and the vehicle coordinate system. Before describing the conversion process, the relevant coordinate systems are first introduced.

In autonomous driving, three kinds of coordinate systems are commonly involved in localization and tracking tasks: world coordinate system, vehicle coordinate system, and radar observation coordinate system [67]. Figure 3.4.1 visualizes the concept and spatial relationship among these coordinate systems.

1. **World Coordinate System**
   The world coordinate system serves as a fixed, global reference frame, with its

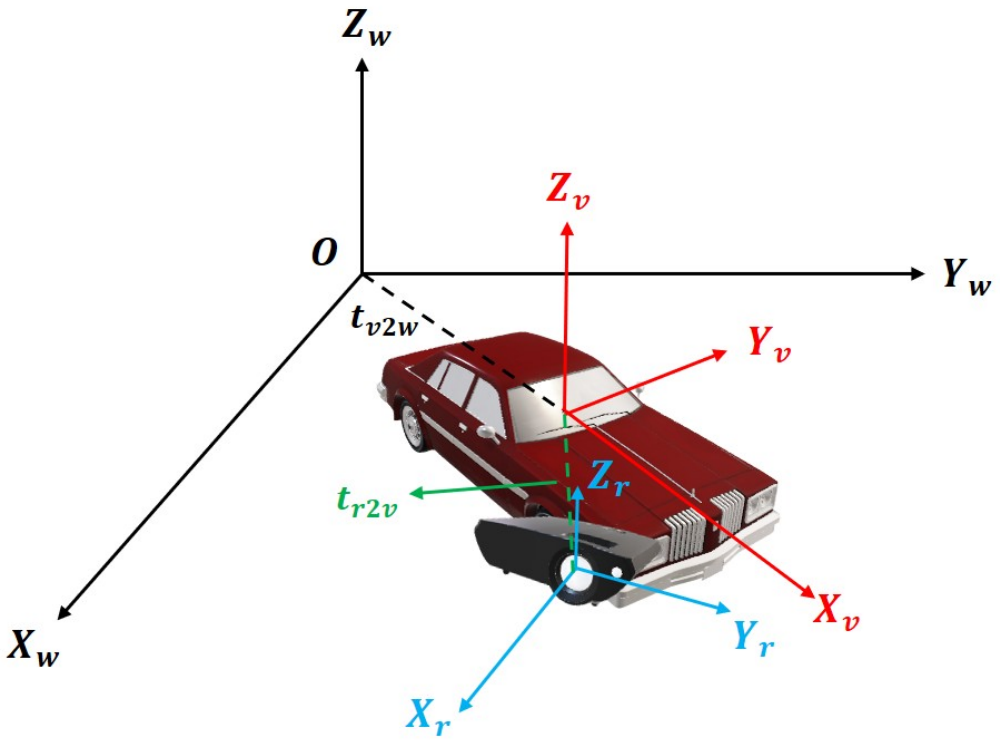


Figure 3.4.1: Concept and spatial relationship among three coordinate systems

origin defined at an arbitrary point on the Earth's surface. The proposed combined estimation algorithm in this thesis aims to estimate the state of both the ego vehicle and surrounding moving objects in the world coordinate frame. Generally, the axes follow the East-North-Up convention:

- X-axis (X_w) points east,
- Y-axis (Y_w) points north,
- Z-axis (Z_w) points upward, perpendicular to the local ground plane.

2. **Vehicle Coordinate System**
   The vehicle coordinate system is defined relative to the ego vehicle and typically follows the ISO 8855 standard: Road vehicles — Vehicle dynamics and road-holding ability — Vocabulary [68], which is widely adopted in the autonomous driving industry [67]. The axes are defined as follows:
   - X-axis (X_v): forward along the vehicle's longitudinal direction (i.e., driving direction),
   - Y-axis (Y_v): to the left of the vehicle (i.e., lateral direction),
   - Z-axis (Z_v): upward, normal to the road surface.
3. **Radar Observation Coordinate System**
   The radar observation coordinate system is specific to each radar sensor and is defined according to the geometry of its transceiver array:

- x-axis (X_r): points outward along the radar beam (sensing direction),
- y-axis (Y_r): aligns with the azimuthal (horizontal) direction of the MIMO antenna array,
- z-axis (Z_r): aligns with the elevation (vertical) dimension of the MIMO antenna array.

The radar's measurement data, including both raw signals and point clouds, are defined in this coordinate system.

To obtain the velocity of the ego vehicle, it is necessary to transform the velocity vector estimated in the radar observation coordinate system into the vehicle coordinate system. Given a known radar installation angle, this can be achieved by applying a rotation matrix:

$$\begin{bmatrix} \hat{v}_x^{ego,vehicle} \\ \hat{v}_y^{ego,vehicle} \\ \hat{v}_z^{ego,vehicle} \end{bmatrix} = \mathbf{R}_{\text{r2v}} \cdot \begin{bmatrix} \hat{v}_x^{ego,radar} \\ \hat{v}_y^{ego,radar} \\ \hat{v}_z^{ego,radar} \end{bmatrix} \tag{3.4.16}$$

where $\mathbf{R}_{\text{r2v}}$ is the rotation matrix from radar to vehicle coordinate system defined in Equation 3.5.3.

Once the velocity of the vehicle is obtained, it is used to update the Kalman filter which estimates the ego vehicle's motion state. The state and covariance matrix are updated using the Equations 3.4.17 to 3.4.19:

$$\mathbf{K}_k = \mathbf{P}_{k|k-1}^{-}\mathbf{H}^\top(\mathbf{H}\mathbf{P}_{k|k-1}^{-}\mathbf{H}^\top + \mathbf{R})^{-1} \tag{3.4.17}$$

$$\hat{\mathbf{x}}_k = \hat{\mathbf{x}}_{k|k-1}^{-} + \mathbf{K}_k(\mathbf{z}_k - \mathbf{H}\hat{\mathbf{x}}_{k|k-1}^{-}) \tag{3.4.18}$$

$$\mathbf{P}_k = (\mathbf{I} - \mathbf{K}_k\mathbf{H})\mathbf{P}_{k|k-1}^{-} \tag{3.4.19}$$

In this case, the measurement vector $\mathbf{z}_k$ contains the estimated ego-velocity in the vehicle coordinate system, and the observation matrix $\mathbf{H}$ maps the state vector to the measured velocity components. The formulation is given in Equation 3.4.20.

$$\mathbf{z}_k = \begin{bmatrix} \hat{v}_x^{ego,vehicle} \\ \hat{v}_y^{ego,vehicle} \\ \hat{v}_z^{ego,vehicle} \end{bmatrix}, \quad \mathbf{H} = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.4.20}$$

3.5 Ego-motion compensation

In Section 3.4, the ego-motion is estimated using all static targets. Meanwhile, this module also achieves the separation of moving and static targets and outputs the point cloud associated with moving targets. The point cloud of moving objects consists of two parts: the first part includes points identified as moving targets using tracking information in Section 3.4.1, and the second part includes all remaining points that are not used in the final ego-motion estimation in Section 3.4.2. These moving points will be used in the subsequent ego-motion compensation and multiple object tracking.

Ego-motion compensation refers to the process of removing the influence of the sensor platform's own motion from sensor measurements. In autonomous driving, onboard

sensors such as radar or LiDAR capture the relative motion of surrounding objects. However, since the ego vehicle itself is moving, these measurements contain components caused by both the ego-motion and the motion of objects. Once the ego-motion is estimated, ego-motion compensation aims to isolate the true motion of objects as if the sensors were stationary. This step is essential for accurate tracking of moving objects.

For radar measurements, especially detected point clouds, ego-motion compensation is implemented through coordinate system transformation. This transforms all measurements from the radar observation coordinate system to the world coordinate system. Since only position information is used as measurement input in the multiple object tracking algorithm, only the measured positions are compensated. The ego-motion compensation procedure can be divided into two steps:

- **Step 1: Coordinate Conversion from Radar Frame to Vehicle Frame.** The detected radar points are first converted from spherical coordinates (range, azimuth, elevation) to 3D Cartesian coordinates. Then, the points are transformed from the radar observation coordinate system to the vehicle coordinate system. This transformation accounts for the radar's installation position and orientation on the vehicle. If the extrinsic calibration parameters are known, the transformation can be represented as:

$$\mathbf{x}_{\text{veh}} = \mathbf{R}_{\text{r2v}} \cdot \mathbf{x}_{\text{radar}} + \mathbf{t}_{\text{r2v}} \tag{3.5.1}$$

  where $\mathbf{x}_{\text{radar}}$ is the position measurement in Cartesian coordinates obtained from the detected radar point cloud:

$$\mathbf{x}_{\text{radar}} = \begin{bmatrix} \hat{R}\cos\hat{\phi}\cos\hat{\theta} \\ \hat{R}\cos\hat{\phi}\sin\hat{\theta} \\ \hat{R}\sin\hat{\phi} \end{bmatrix} \tag{3.5.2}$$

  $\mathbf{R}_{\text{r2v}}$ is the rotation matrix and $\mathbf{t}_{\text{r2v}}$ is the translation vector from radar to vehicle coordinate system. Specifically, these are defined as:

$$\mathbf{R}_{\text{r2v}} = \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad \mathbf{t}_{\text{r2v}} = \begin{bmatrix} t_x \\ t_y \\ t_z \end{bmatrix} \tag{3.5.3}$$

  Here, θ denotes the rotational angle between radar and the vehicle in the horizontal plane, while t_x, t_y, t_z represent the radar's position offset along the vehicle's x, y, and z axes, respectively. These parameters are obtained according to sensor characteristics through sensor calibration.

- **Step 2: Compensation from Vehicle Frame to World Frame.** To remove the ego-motion effect, the points are further transformed from the vehicle coordinate system to the world coordinate system, using the current vehicle position estimated in Section 3.4. The transformation can be represented as:

$$\mathbf{x}_{\text{world}} = \mathbf{R}_{\text{v2w}} \cdot \mathbf{x}_{\text{veh}} + \mathbf{t}_{\text{v2w}} \tag{3.5.4}$$

where $\mathbf{R}_{\text{v2w}}$ is the rotation matrix and $\mathbf{t}_{\text{v2w}}$ is the translation vector from vehicle to world coordinate system. Specifically, these are defined as:

$$\mathbf{R}_{\text{v2w}} = \begin{bmatrix} \cos\alpha & \sin\alpha & 0 \\ -\sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad \mathbf{t}_{\text{v2w}} = \begin{bmatrix} \hat{x}^{ego} \\ \hat{y}^{ego} \\ \hat{z}^{ego} \end{bmatrix} \tag{3.5.5}$$

Here, α denotes the rotational angle between the vehicle's heading direction and the East direction, while $\hat{x}^{\text{ego}}, \hat{y}^{\text{ego}}, \hat{z}^{\text{ego}}$ are the 3D estimated positions of the ego vehicle obtained from Section 3.4.

The output of the ego-motion compensation module is the compensated position measurement $\mathbf{x}_{\text{world}}$ of each moving point, which serves as the input to the multiple object tracking module described in the subsequent Section 3.6.

3.6 Multiple object tracking

Given a radar point cloud after ego-motion compensation, the goal of MOT algorithms is to determine the number of moving objects and their state over time. The state variable of each object $\mathbf{x}_j^k$ includes its 3D position $\mathbf{p}_j^k$ and velocity $\mathbf{v}_j^k$. The output is a set of tracking trajectories for all moving objects, each associated with a unique object identity.

Figure 3.6.1 shows the functional modules within MOT algorithms. First, clustering is performed to group the detections of the same object and remove the clutter. For each time step, gating rejects invalid detections that are too far from the existing object tracks. Only valid detections enter the data association module, which assigns detections to each track. With the prediction step, assigned tracks are updated with the corresponding detections inside the state estimation module. Unassigned tracks are kept, only performing prediction but without updates. Unassigned detections are sent to the track management module for track initialization. Tracks after state estimation are also sent to the track management module for track confirmation and deletion. Finally, the estimated states for each confirmed object track are provided in the output. The submodules of the tracking system are described in detail in Sections 3.6.1 to 3.6.4, namely clustering, state estimation, gating and data association, and tracking management.

3.6.1 Clustering

As one target can cause multiple detections (i.e., extended target), it is necessary to group these detections into clusters and compute their centroid positions before passing them to the tracker. Additionally, radar point clouds often contain a large number of spurious points caused by noise or environmental clutter. These noisy points are typically spatially isolated and can be effectively removed using a clustering algorithm. In this work, the DBSCAN clustering algorithm [41] is employed to group multiple detections belonging to the same physical object and suppress clutter. DBSCAN identifies clusters of arbitrary shape based on the density of surrounding points. It is particularly

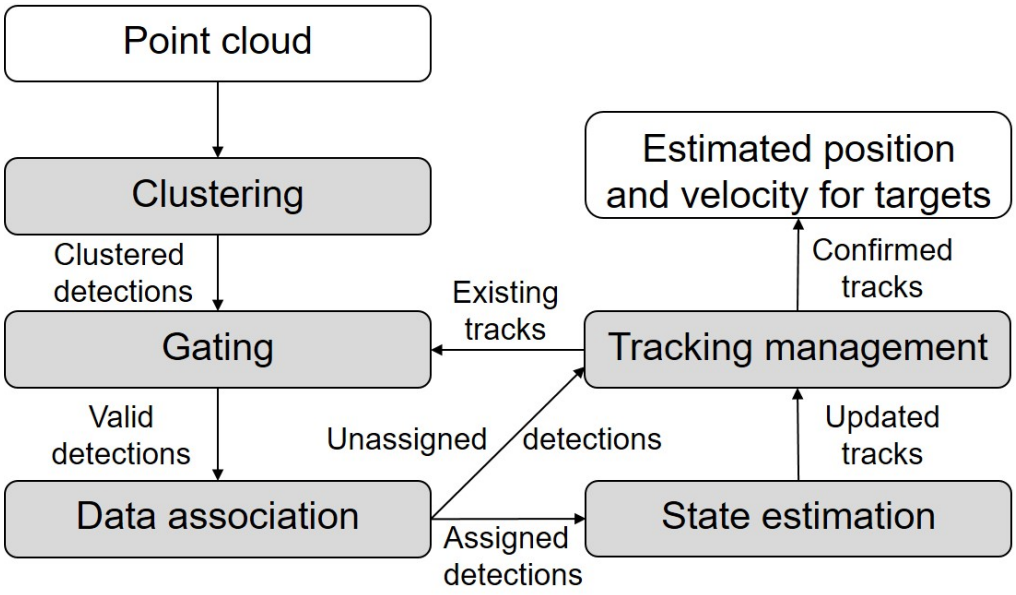


Figure 3.6.1: High-level block diagram of MOT algorithms with related submodules

effective for noisy radar point clouds, as it naturally clusters high-density regions while ignoring sparse areas that resemble noise. Compared to traditional K-means clustering, DBSCAN does not require the number of clusters to be specified in advance and can detect clusters of varying shapes and sizes. These properties make it highly suitable for radar-based tracking applications in autonomous driving.

Figure 3.6.2 shows the concept of DBSCAN algorithm. There are two hyperparameters: radius ε and minimum number of points $minPts$ in this algorithm. If the number of points inside the circle with radius ε is $minPts$ or more, the center point is denoted as a core point (the red point in the left figure). For each core point (red points in the right figure), the cluster can be expanded by including all points inside the circle. If the new point is also a core point, it can include all points inside its circle as well. However, if the new point is not a core point (i.e., the number of points inside the circle with radius ε is less than $minPts$), this is termed as a border point (blue points in the right figure). The border point cannot add other points to the cluster. After all clusters are formed, points that are not part of any cluster are labeled as noise points (yellow points in the right figure).

In this thesis, after tuning the hyperparameters based on the clustering performance, the radius ε is set to 2 meters, and the minimum number of points $minPts$ is set to 5. Since the radar point cloud is very sparse for a single frame, the point cloud of 4 adjacent frames are integrated for each time step in the subsequent simulations to perform clustering.

After DBSCAN, the noisy points are discarded, and the centroid of each cluster is calculated as an average of all points inside the cluster. Now each target can generate at most one detection. As in equation 3.6.1, the clustered detection for target i at time

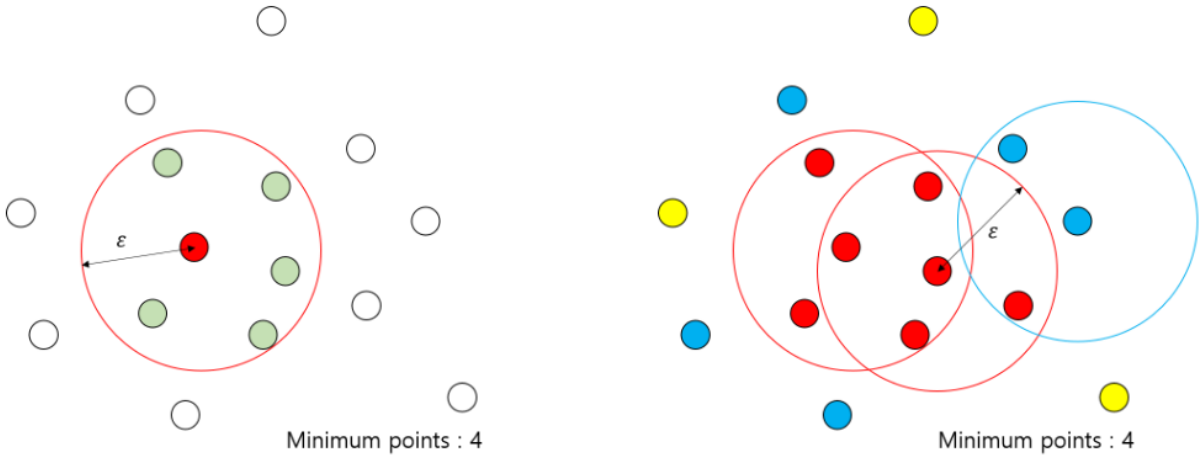


Figure 3.6.2: Concept of DBSCAN algorithm [42], with core points (red), border points (blue) and noise points (yellow).

step k includes the measured position of each dimension:

$$\mathbf{z}_k^i = \begin{bmatrix} p_x \\ p_y \\ p_z \end{bmatrix} \tag{3.6.1}$$

where p_x, p_y, and p_z denote the Cartesian coordinates of the centroid of the object in 3D space.

For each time step, clustered detections are collected and formed as the measurement matrix in Equation 3.6.2, which is the input of the following step of the MOT algorithm. $M(k)$ denotes the number of detections at time step k, which can be different for each time step.

$$\mathbf{Z}_k = [\mathbf{z}_k^1, \mathbf{z}_k^2, \ldots, \mathbf{z}_k^{M(k)}], \quad k = 1, \ldots, K \tag{3.6.2}$$

3.6.2 State estimation: Prediction and Update

After clustering, detections in Equation 3.6.2 are obtained to perform state estimation. As illustrated in Section 3.1, the goal is to estimate the state matrix $\mathbf{x}_j^k$ in Equation 3.1.2 for all time steps. This problem is addressed within the Bayesian inference framework. It assumes the state variable is a random variable characterized by a Probability Density Function (PDF). Recursive Bayesian filters are employed to iteratively predict and update this PDF based on the system's motion model and measurement model. The final state estimate is derived as:

- The mean of the posterior PDF when using the MMSE estimator
- The mode of the posterior PDF when using the MAP estimator

In this work, both the motion model and the measurement model are assumed to be linear with additive Gaussian noise. Under these assumptions, the Kalman filter provides the optimal solution by recursively estimating the state mean and covariance.

In the prediction step, the predicted mean $\hat{\mathbf{x}}_{k|k-1}$ and covariance $\mathbf{P}_{k|k-1}$ are calculated according to Equation 3.6.3 and 3.6.4.

$$\hat{\mathbf{x}}_{k|k-1} = \mathbf{F}\hat{\mathbf{x}}_{k-1|k-1} \tag{3.6.3}$$

$$\mathbf{P}_{k|k-1} = \mathbf{F}\mathbf{P}_{k-1|k-1}\mathbf{F}^{\top} + \mathbf{Q} \tag{3.6.4}$$

In the update step, the posterior mean and covariance are corrected based on the incoming measurements, according to Equation 3.6.5, 3.6.6 and 3.6.7.

$$\mathbf{K}_k = \mathbf{P}_{k|k-1}\mathbf{H}^{\top}\left(\mathbf{H}\mathbf{P}_{k|k-1}\mathbf{H}^{\top} + \mathbf{R}\right)^{-1} \tag{3.6.5}$$

$$\hat{\mathbf{x}}_{k|k} = \hat{\mathbf{x}}_{k|k-1} + \mathbf{K}_k\left(\mathbf{z}_k - \mathbf{H}\hat{\mathbf{x}}_{k|k-1}\right) \tag{3.6.6}$$

$$\mathbf{P}_{k|k} = \left(\mathbf{I} - \mathbf{K}_k\mathbf{H}\right)\mathbf{P}_{k|k-1} \tag{3.6.7}$$

Here, $\mathbf{K}_k$ is the Kalman gain, $\hat{\mathbf{x}}_{k|k}$ is the state estimate (mean vector), and $\mathbf{P}_{k|k}$ is the covariance matrix.

Moreover, after the prediction step, the mean and covariance of the predicted measurement can be obtained in Equation 3.6.8 and 3.6.9, which is useful in gating and data association in the next section.

$$\hat{\mathbf{z}}_k = \mathbf{H}\hat{\mathbf{x}}_{k|k-1} \tag{3.6.8}$$

$$\mathbf{P}_{z,k} = \mathbf{H}\mathbf{P}_{k|k-1}\mathbf{H}^{\top} + \mathbf{R} \tag{3.6.9}$$

3.6.3 Gating and data association

In this sub-section, the gating and data association steps are described.

3.6.3.1 The Number of Hypotheses

After detections are obtained, they are assigned to multiple objects to perform state estimation. If the task is tracking a single target with no clutter, it is extremely simple to solve by obtaining only one detection per time step and utilizing this detection to update the state with one Kalman filter as described in Section 3.6.2. However, as illustrated in Section 3.1, the multiple object tracking problem is much more challenging because there are multiple detections per time step and the source of each detection is not known as a priori. Each detection can be originated from any existing target, a new target, or clutter, and all possibilities should be taken into consideration. It is thus necessary to include a step to determine the association from detections to targets before state estimation. Each possibility can generate a distinct hypothesis, and the final association is determined based on those hypotheses. In this section, the number of hypotheses is first calculated.

A hypothesis is a specific joint association which associates each detection with a target or clutter identity. Consider there are $m(k)$ detections and $n(k)$ targets at time step k; the number of possible hypotheses is then shown in Equation 3.6.10:

$$N(m(k), n(k)) = \sum_{p=0}^{min(m(k),n(k))} C_{n(k)}^{p} C_{m(k)}^{p} A_{p}^{p} = \sum_{p=0}^{min(m(k),n(k))} \frac{m(k)!n(k)!}{p!(m(k)-p)!(n(k)-p)!} \tag{3.6.10}$$

For different values of $m(k)$ and $n(k)$, the total number is listed in Table 3.6.1 for an intuitive display:

Table 3.6.1: Number of hypotheses for $m(k)$ detections and $n(k)$ targets

m(k)\n(k)	1	2	3	4	5	6	7
0	1	1	1	1	1	1	1
1	2	3	4	5	6	7	8
2	3	7	13	21	31	43	57
3	4	13	34	73	136	229	358
4	5	21	73	209	501	1045	1961
5	6	31	136	501	1546	4051	9276
6	7	43	229	1045	4051	13327	37633
7	8	57	358	1961	9276	37633	130922

The above table is just the number of hypotheses from a single time step. Theoretically, if all hypotheses are considered at each time step, the total number of possible hypotheses will depend on all previous time steps, and this will increase very fast with time step k as:

$$N_k = \prod_{t=1}^{k} N(m(t), n(t)) \tag{3.6.11}$$

Therefore, considering all possible hypotheses is not computationally tractable, and approximation methods should be used to reduce the number of hypotheses [69]. Pruning methods aim to reduce the number of hypotheses by eliminating less likely or irrelevant hypotheses early in the process. Essentially, pruning methods cut down the search space, focusing only on the most probable or relevant associations. The Gating and GNN data association method are two possible pruning methods.

3.6.3.2 Gating

Gating is a common technique in tracking algorithms. The concept is to build a gate around the predicted measurement of each target and only accept the detections inside the gate for the following data association. In this way, the method can prune unlikely hypotheses where the distance between the predicted measurement and the detection becomes too large.

In this thesis, Mahalanobis gating [38] is implemented since the predicted density and posterior density are assumed to be Gaussian. First, a validation matrix is initialized with each row representing a target and each column representing a detection.

If a detection is inside the gate of a target, the corresponding element will be set to 1. Otherwise, it will be set to 0. Next, the approach to decide if a detection is inside the gate is calculating the Mahalanobis distance between the predicted measurement of the state and the detection, and comparing the distance with a gating threshold γ. The Mahalanobis distance is shown in Equation 3.6.12:

$$d^{i,j} = \sqrt{(\mathbf{z}^j - \hat{\mathbf{z}}^i)^T (\mathbf{P}_z^i)^{-1} (\mathbf{z}^j - \hat{\mathbf{z}}^i)} \tag{3.6.12}$$

where $\mathbf{z}^j$ is the detection, $\hat{\mathbf{z}}^i$ is the predicted measurement of the state given in Equation 3.6.8, and $\mathbf{P}_z^i$ is the covariance matrix for the predicted state given in Equation 3.6.9.

A common strategy to select the gating threshold γ is to first determine the gating probability P_G, which is the probability that a measurement originated by the target is inside the gate. [38]. Then the gating threshold γ can be set with the cumulative chi-square distribution. In this thesis, the gating probability P_G is set to 0.5.

After the validation matrix is determined, the detections that do not fall inside any gates are labeled as unassigned detections and deleted from the list of valid detections. In this way, the size of the cost matrix to be calculated in the data association step will be much smaller, decreasing both the computational complexity and the space storage.

3.6.3.3 Global Nearest Neighbor

Global Nearest Neighbor [52] is a simple but effective data association method. It performs global optimization in accordance with the maximum likelihood criteria. For each time step, it finds the single hypothesis with the largest likelihood, pruning all other hypotheses. This algorithm is simple to implement and has low computational complexity.

First, this algorithm calculates a cost matrix for each target and each valid detection. It also takes missed detections into consideration. The cost is the negative log-likelihood of assigning a detection to a target or deciding the target is not detected. In case of detection, the cost relates to the Mahalanobis distance between the predicted measurement and the detection as shown in Equation 3.6.12, and also the clutter intensity λ_c. Compared with the euclidean distance which only considers position, the Mahalanobis distance also considers the uncertainty and correlation of the prediction. The larger a distance is, the less likely an association will be. It also considers the existence of false alarms, with clutter intensity λ_c representing the number of false alarms per unit volume. After some derivations, the cost in case of detection can be represented as:

$$\ell^{i,j} = -log(\frac{P_D}{\lambda_c}) + \frac{1}{2} log(det(2\pi \mathbf{P}_z^i) + \frac{1}{2}(d^{i,j})^2 \tag{3.6.13}$$

where P_D is the probability of detection, λ_c is the clutter intensity, $\mathbf{P}_z^i$ is the covariance matrix for the predicted measurement defined in Equation 3.6.9, and $d^{i,j}$ is the Mahalanobis distance calculated in Equation 3.6.12.

In case of missed detection, the cost relates to the probability of detection P_D:

$$l^{i,0} = -log(1 - P_D) \tag{3.6.14}$$

Assuming m detections and n targets, the cost matrix $\mathbf{L}$ can be represented in Equation 3.6.15:

$$\mathbf{L} = \begin{bmatrix} \ell^{1,1} & \ell^{1,2} & \dots & \ell^{1,m} & \ell^{1,0} & \infty & \cdots & \infty \\ \ell^{2,1} & \ell^{2,2} & \dots & \ell^{2,m} & \infty & \ell^{2,0} & \cdots & \infty \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ \ell^{n,1} & \ell^{n,2} & \dots & \ell^{n,m} & \infty & \infty & \cdots & \ell^{n,0} \end{bmatrix} \tag{3.6.15}$$

In this thesis, after some empirical verifications, the clutter intensity λ_c is set to 10^{-6}, and the probability of detection P_D is set to 0.9.

After the cost matrix is formulated, the algorithm finds the best assignment by solving a linear assignment optimization problem. The assignment decision is encoded in an assignment matrix $\mathbf{A}$. As shown in Equation 3.6.16, this has the same dimensions as the cost matrix, with each element representing if the detection is assigned to the target.

$$\mathbf{A} = \begin{bmatrix} A^{1,1} & A^{1,2} & \dots & A^{1,m} & A^{1,m+1} & 0 & \cdots & 0 \\ A^{2,1} & A^{2,2} & \dots & A^{2,m} & 0 & A^{2,m+2} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ A^{n,1} & A^{n,2} & \dots & A^{n,m} & 0 & 0 & \cdots & A^{n,m+n} \end{bmatrix} \tag{3.6.16}$$

Then, an optimization problem can be formulated as in Equation 3.6.17. The optimization goal is to find the assignment matrix $\mathbf{A}$ which minimizes the total cost, or maximizes the total likelihood. There are three constraints on the assignment matrix. First, the value must be '1' (assigned) or '0' (not assigned). Second, each target must be assigned to a detection or labeled as missed detection, so each row should have only one '1'. Finally, each detection can be assigned to at most one target, so each column should have zeroes or one '1'.

$$\begin{aligned} \text{minimize} \quad & \operatorname{tr}(\mathbf{A}^T\mathbf{L}) \\ \text{subject to} \quad & A^{i,j} \in \{0,1\}, \quad i,j \in \{1,\dots,n\} \times \{1,\dots,n+m\}, \\ & \sum_{j=1}^{n+m} A^{i,j} = 1, \quad i \in \{1,\dots,n\}, \\ & \sum_{i=1}^{n} A^{i,j} \in \{0,1\}, \quad j \in \{1,\dots,n+m\}. \end{aligned} \tag{3.6.17}$$

This optimization problem can be solved using the Munkres algorithm, which is an extension of the Hungarian algorithm. The Hungarian algorithm is developed by Harold Kuhn in 1955 based on the work of two Hungarian mathematicians. It is designed to solve the linear assignment problem for square matrices. In 1957, James Munkres reviewed this algorithm and developed a version suitable for rectangular matrices. This has polynomial time complexity: $O(n^3)$. In this thesis, the Munkres algorithm implemented by [70] is used to find the best assignment matrix.

After determining the assignment matrix, the algorithm can output three variables: assigned target-detection pairs, unassigned target tracks, and unassigned detections.

Then the Kalman filter can update each assigned track with the corresponding detection. Unassigned tracks will be kept without update, while unassigned detections will lead to track initialization in the track management module.

3.6.4 Tracking management

In multiple object tracking problems, the number of targets is unknown and time varying. Approaches using the GNN without tracking management are practical only when the number of targets is known [69], whereas in real cases, it remains a problem to establish how many targets exist in the field of view of the radar sensor. Moreover, since the field of view is finite and the sensor can be moving, targets can appear and disappear from the area of interest, leading to a time varying number of tracks. To solve these problems, the tracking management module is implemented and integrated into the GNN method.

Tracking management involves the function of track initialization, confirmation, and deletion. There are two types of decision logic: history-based logic and score-based logic [71]. In the history-based logic, the tracker counts the number of detections assigned to a track in several recent updates. If enough detections are assigned, the track is confirmed. If the track is not assigned to any detection for enough updates, it is deleted. In the score-based logic, the tracker maintains a score for each track, which represents the likelihood of a real target. A high positive track score means that the track is very likely to be of a real target. A very negative track score means that the track is likely to be false. As a result, a threshold for confirming a track can be set if the score is high enough. If the score is low, the track is deleted.

For simplicity, in this thesis, history-based logic is implemented. Next, its implementation is illustrated in detail.

- Track struct array: The tracker maintains a list of tracks as time evolves. Each track is a struct variable containing the following elements:

  1. Track index: the distinctive index of the track, depending on the order of appearance;
  2. Track age: the total time steps during which the track is assigned detections to;
  3. Track state: the mean vector and covariance matrix of the estimated state;
  4. Track status: tentative or confirmed;
  5. Track logic state: a logical vector recording the recent track logical states. True (1) values indicate hits (detection), and false (0) values indicate misses (missed detection). For example, [1 0 1 1 1] represents four hits and one miss in the last five updates;
  6. Track appearance frame: the frame index when the track first appears.

- Track initialization: After data association, unassigned detections can start a new track. At first, it is unclear whether this track represents a true target or false alarm, so the track status is set to 'tentative'. Only after the confirmation logic

condition is met, the track status becomes 'confirmed'. There are two types of unassigned detections for GNN. The first type are the detections that do not fall inside any gates, also called invalid detections. The second type are the detections not assigned to any target in the optimal assignment.

After a new track is initialized, it is added to the end of the track struct array. The state mean vector sets the detection as its position, and assumes zero initial velocity. The covariance matrix is set to $100 \times \mathbf{I}_{6\times6}$, where $\mathbf{I}_{6\times6}$ is the identity matrix. The initial track logic state is a zero vector, and the initial track status is 'tentative'.

- Track confirmation: In this thesis, the confirmation logic is defined as **2** assigned detections in the recent **3** time steps. If a track is confirmed, the status will be changed into 'confirmed'. Only confirmed tracks are considered in the performance analysis.
- Track deletion: In this thesis, the confirmation logic is defined as **3** missed detections in the recent **3** time steps. Starting from the third step after track appearance, the track deletion logic becomes valid. If a track remains unassigned for 3 consecutive steps, it will be deleted from the track struct array.

3.7 Performance metrics

To quantitatively evaluate the performance of the proposed combined ego-motion estimation and multiple object tracking method, three metrics derived from the literature are adopted: Absolute Pose Error (APE), Relative Trajectory Error (RTE), and Generalized Optimal Sub-Pattern Assignment (GOSPA). The evaluation criteria for ego-motion estimation and tracking are presented in Sections 3.7.1 and 3.7.2, respectively.

3.7.1 Ego-motion estimation metrics

In this thesis, similar with [36], two metrics are used to evaluate the performance of ego-motion estimation algorithms: APE and RTE. APE measures the pose difference between the estimation and the true motion. As shown in Equation 3.7.1, it calculates the Root Mean Square Error (RMSE) between the estimated poses and the actual poses. In this thesis, the pose refers to the translational velocities in three dimensions.

$$\epsilon_{APE} = \sqrt{\frac{1}{m}\sum_{i=1}^{m} \|P_{est}(i) - P_{actual}(i)\|^2} \tag{3.7.1}$$

where m is the total number of frames, P_{est} and P_{actual} are the estimated velocities and actual velocities, respectively. L-2 norm is used to calculate the Euclidean distance.

For the results presented in Chapters 6 and 7, the APE for each frame is also calculated for better analysis on the frame level, without a very long (temporally speaking) averaging:

$$\epsilon_{APE,i} = \|P_{est}(i) - P_{actual}(i)\| \tag{3.7.2}$$

where i is the frame index.

While APE focuses on the instantaneous velocity estimation error, RTE measures the long-term localization error. Absolute Trajectory Error (ATE) is a common long-term error metric, which measures the RMSE between the estimated positions and actual positions. However, since it is computed over the entire trajectory, it is more sensitive to early errors [5]. Different from ATE, RTE measures the relative position error over short segments. This is the RMSE of the differences between the relative displacements over a small period in the estimated trajectory and the actual trajectory.

As in Equation 3.7.3, T_{est} and T_{actual} are the estimated positions and actual positions, respectively. N is the segment interval, which is a parameter of the metric. In this project, N is chosen as 10 frames for better comparison of results.

$$\epsilon_{RTE} = \sqrt{\frac{1}{m-N}\sum_{i=1}^{m-N}\left(\|T_{est}(i+N) - T_{est}(i)\| - \|T_{actual}(i+N) - T_{actual}(i)\|\right)^2} \tag{3.7.3}$$

3.7.2 Multiple object tracking metrics

In this thesis, the GOSPA [72] metric is used to evaluate the performance of the proposed multiple object tracking method quantitatively. The output of the multiple object tracking method at every time step is a list of estimated target 3D position vectors. Each element represents the tracking position of one target. The GOSPA metric measures the difference or error between the output list and the ground truth list.

As shown in Figure 3.7.1, GOSPA measures three types of error. The first error is the localization error between the estimated target position and actual target position if a target is detected correctly (e.g., the lower left target in the figure). However, considering the challenges mentioned in Section 2.2, there are other two error types caused by missed detections and false alarms. They are measured by the number of targets which are not correctly detected in the ground truth list (the upper left target) and the number of targets which do not appear in the ground truth list but appear in the output list (the lower right targets). These error types together contribute to a comprehensive evaluation for the tracking result.

Equation 3.7.4 shows the GOSPA metric between the output list $\mathbf{X}$ and the ground truth list $\mathbf{Y}$. γ is the global association hypothesis which assigns elements from $\mathbf{X}$ to $\mathbf{Y}$. Γ is the set of all possible global association hypotheses. There are three hyperparameter choices for the metric: the order of distance p, the distance metric $d(x, y)$, and the maximum allowable localization error c. In this thesis, in order to make the localization error component the same as the RMSE, p is set to 2. The Euclidean distance is chosen as the distance metric d in a conventional manner. c determines the trade-off between the localization error component, and the missed detection and false alarm number component. This can be considered as the distance where the designer wants to penalize a false or missing estimate. In this theis, c is set to 10 meters, the same choice as in some literature in automotive vehicle perception

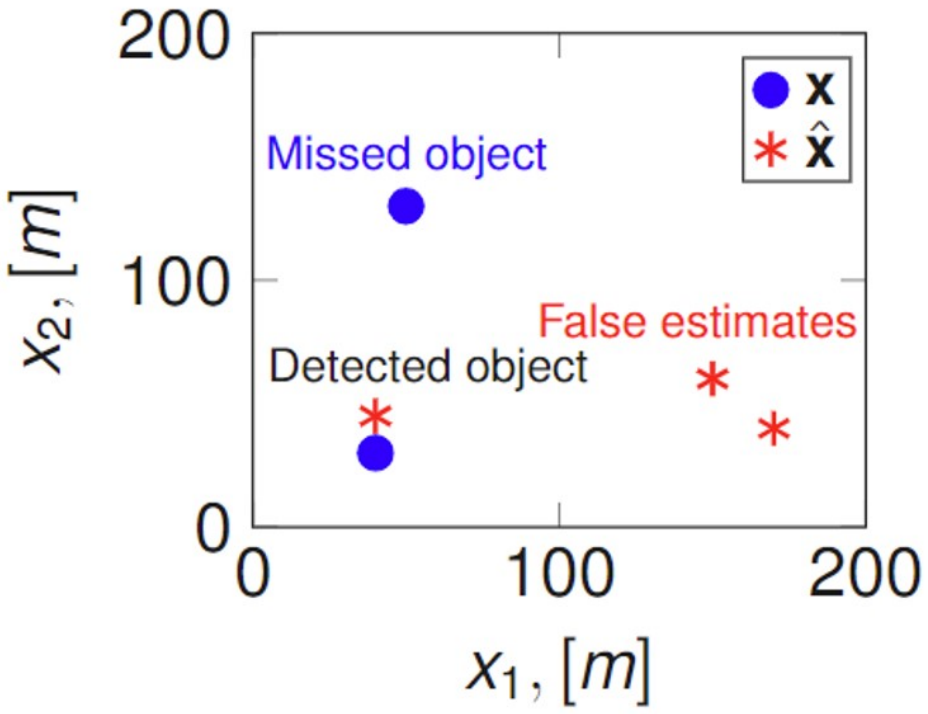


Figure 3.7.1: Concept of GOSPA metric with three cases contributing to the overall error metric.

applications [69, 73, 74].

$$GOSPA(\mathbf{X}, \mathbf{Y}) = \left[\min_{\gamma \in \Gamma} \left(\sum_{(i,j) \in \gamma} d(x_i, y_j)^p + \frac{c^p}{2} \left(|\mathbf{X}| - |\gamma| + |\mathbf{Y}| - |\gamma| \right) \right) \right]^{1/p} \tag{3.7.4}$$

The GOSPA metric measures the tracking performance for each time frame. In this thesis, the mean of GOSPA from all time frames is used to evaluate the performance across time. This evaluation metric is thus denoted by Mean GOSPA, and it is measured for multiple object tracking algorithms.

In this chapter, the raw signal based combined method is explained in detail. After the initialization phase in which tracks are established, the tracking information from the previous frame is incorporated to guide the ego-motion estimation of the current frame. The proposed design integrates ego-motion estimation and multiple-object tracking in a tightly coupled manner. In the next chapter, this method will be validated using simulated raw signals.

4 Simulation Results

To validate the performance and show the feasibility of the proposed raw signal based combined method for ego motion estimation & multiple object tracking, a series of tests based on simulations are conducted. In this chapter, results and discussions of these simulations are presented. First, in Section 4.1, the adopted automotive radar simulator and some general simulation settings are introduced. Next, the simulation results and discussions are provided in Section 4.2.

4.1 Simulation Setup

When facing a lack of relevant experimental data, simulation is always regarded as a strong supplementary data source for radar research, especially in automotive applications [75]. In this thesis, the radar simulator used to generate raw signals is implemented based on the framework proposed in a previous MSc project [75]. This is done due to the lack of well-documented and complete datasets in the literature that can also offer raw radar data and not just point clouds.

An automotive MIMO radar with eight virtual array elements for azimuth and eight for elevation estimation is considered here. An omnidirectional antenna pattern is considered for the transmitter and receiver. A typical parameter setting is used for the FMCW waveform, with 77.25 GHz centre frequency, $40\mu s$ PRI and 0.5 GHz bandwidth. The range resolution under this setting is equal to 0.3 m, and the Doppler resolution is 0.191 m/s. The time interval between two radar frames is 10.24 ms. The selected radar parameters for this simulation are listed in Table 4.1.1.

Table 4.1.1: Radar parameters in the simulations in Chapter 4

Parameters	**Value**
Center Frequency f_c (GHz)	77.25
Bandwith B (GHz)	0.5
Slope μ (MHz/μ s)	62.5
Chirp duration T_d (μ s)	16
PRI T (μ s)	40
Number of samples per chirp N_{range}	512
Number of chirps per frame $N_{Doppler}$	256
Number of virtual array elements in azimuth N_a	8
Number of virtual array elements in elevation N_e	8
Spacing between adjacent antennas in both dimensions d (m)	0.0019
Frame time T_{frame} (ms)	10.24

While radar parameter estimation algorithms estimate the range, Doppler velocity,

azimuth and elevation angle from the raw radar signals, the goal of the implemented radar simulator is to generate the raw signals assuming these measurements are known. As shown in Equation 4.1.1, given the radar measurements of point target i: (with Range: R_i, Doppler velocity: v_i, Azimuth angle: θ_i, Elevation angle: ϕ_i), the radar simulator can generate the received raw signals in the form of a 4-dimensional tensor:

$$\hat{z}_i(l,b,p,q) = \alpha_i * exp(j2\pi a(\theta_i,\phi_i)p) * exp(j2\pi e(\phi_i)q) * exp(j2\pi f_d(v_i)l) * exp(j2\pi f_r(R_i)b) \tag{4.1.1}$$

where

$$a(\theta_i,\phi_i) = f_0\frac{d}{c}sin(\theta_i)cos(\phi_i), e(\phi_i) = f_0\frac{d}{c}sin(\phi_i), f_d(v_i) = -\frac{2v_if_0}{c}T, f_r(R_i) = -\frac{\mu 2R_i}{f_sc} \tag{4.1.2}$$

f_0 is the center frequency of the radar, d is the distance between adjacent antenna elements, T is the PRI including chirp duration and settle time, μ is the slope of frequency change of the radar waveform, and f_s is the sampling frequency.

For simplification of subsequent signal processing, the 4-dimension tensor has to be reshaped to the 3-dimensional tensor by stacking azimuth and elevation dimensions together as:

$$\mathbf{Z}_i(l,b,pN_e+q) = \hat{z}_i(l,b,p,q) \tag{4.1.3}$$

In this thesis, one radar raw signal unit of information contains a single frame of simulated radar data, where its three axes correspond to the fast-time ($K = 512$), slow-time ($L = 256$), and number of channels combining azimuth and elevation ($P = 64$). In each simulation, raw radar data of 500 frames is generated in total. The total simulation time is 5.12 seconds.

The simulated scenario consists of a car equipped with a front-corner radar moving at a constant speed, alongside multiple static and moving objects. The front-corner radar is widely adopted in automotive industry to enable detection of objects in both the forward and lateral directions for functions such as Cross Traffic Alert (CTA) and Precrash [76]. It is also particularly suitable for validating the combined method of ego-motion estimation and multiple object tracking, as ego-motion estimation typically relies on roadside static targets, while multiple object tracking focuses on the detection of critical moving objects in front of the vehicle. For simplicity, the ego vehicle is modeled as a point target, and the radar is assumed to be located at the same position. The radar beam is oriented at an angle of 45 degrees to the right relative to the vehicle's forward direction in the horizontal plane, and 0 degrees in the vertical plane.

In this chapter, six parallel simulation scenarios are conducted, with the ego vehicle moving at constant speeds of 8, 9, 10, 11, 12, and 13 m/s, respectively. The ego vehicle is assumed to have zero velocity in both the cross-forward and elevation directions, an assumption that can align with real-world driving conditions on well-maintained roads.

Within the radar's field of view, 16 objects are generated in each simulation—8 static and 8 moving objects—representing a dynamic scenario with 50% of moving objects. Each object is modeled as a collection of scatterers randomly distributed in 3D space. The range of these scatterers is selected from $[0, 35]$ meters, the azimuth angle from $[-60, 60]$ degrees, and the elevation angle from $[-30, 30]$ degrees. The amplitude of all scatterers is drawn from the uniform distribution $\alpha_o \sim \mathcal{U}(0, 300)$. According

to the Swerling Model I the amplitude can be seen as constant during one coherent processing interval [63]. The scatterers are also assumed to be isotropic and provide constant amplitude and phase during the processing period. The static objects in this scenario represent common roadside elements such as trees, traffic signs, or parked vehicles. The 8 moving objects include a mixture of 2 cars (moving at 9-10 m/s, i.e., 32.4–36 km/h), 4 bicycles (moving at 3–4.5 m/s, i.e., 10.8–16.2 km/h), and 2 pedestrians (moving at 1.5–2 m/s, i.e., 5.4–7.2 km/h). These targets exhibit diverse trajectories and Doppler signatures, providing a challenging setting for ego-motion estimation and multiple object tracking algorithms.

The simulation begins by determining the 3D initial position and velocity of the ego vehicle and each object. Next, in each frame, the position of the vehicle (i.e., of the radar as well) is updated first according to the constant velocity motion model. Then, the positions of moving targets are updated in the world coordinate. With the known positions and velocities of both radar $(p_{rx}, p_{ry}, p_{rz}, v_{rx}, v_{ry}, v_{rz})$ and targets (assuming one scattering point i of the target: $p^i_{tx}, p^i_{ty}, p^i_{tz}, v^i_{tx}, v^i_{ty}, v^i_{tz}$), the range R_i, velocity v_i, azimuth angle θ_i, and elevation angle ϕ_i for each target will be measured in the radar coordinate system. The measurement equations for this are given from Equation 4.1.4 to 4.1.7:

$$R_i = \sqrt{(p^i_{tx} - p_{rx})^2 + (p^i_{ty} - p_{ry})^2 + (p^i_{tz} - p_{rz})^2} \tag{4.1.4}$$

$$v_i = (v_{rx} - v^i_{tx}) * \cos(\theta_i) * \cos(\phi_i) + (v_{ry} - v^i_{ty}) * \sin(\theta_i) * \cos(\phi_i) + (v_{rz} - v^i_{tz}) * \sin(\phi_i) \tag{4.1.5}$$

$$\theta_i = \arcsin(\frac{p^i_{ty} - p_{ry}}{R_i * \cos(\phi_i)}) \tag{4.1.6}$$

$$\phi_i = \arcsin(\frac{p^i_{tz} - p_{rz}}{R_i}) \tag{4.1.7}$$

Finally, the measurements are sent to the raw data synthesis module, which outputs the measured raw signals for that frame based on Equation 4.1.1 to 4.1.3. To make the simulation a bit closer to the reality, white Gaussian noise with zero mean and +20 dB Signal-to-Noise Ratio (SNR) is added manually to the data, generating the final simulated radar data which is the input to the combined method illustrated in Chapter 3. A relatively low noise level has been selected for better performance in the examples here. For results in noisier scenarios, please refer to the EuRAD 2025 paper in Appendix A of this thesis.

4.2 Results and Discussions

To evaluate the precision and robustness of the proposed combined method, simulations were performed in six scenarios with the ego-velocity ranging from 8 m/s to 13 m/s. Each simulation lasted 5.12 s (equivalent to 500 consecutive frames). The quantitative performance metrics defined in Section 3.7 for different ego-vehicle velocity are shown in Table 4.2.1.

As shown in the table above, both ego-motion estimation and multiple object tracking achieve accurate and stable performance across different ego-velocities. Compared

Table 4.2.1: Performance evaluation with different ego-velocity in 3D space

Ego-velocity (m/s)	**APE (m/s)**	**RTE (cm)**	**Mean GOSPA**
8	0.35	2.37	12.68
9	0.39	2.18	11.20
10	0.25	1.64	11.19
11	0.27	1.66	14.49
12	0.28	1.59	12.02
13	0.29	1.73	13.51

to the results reported in [6] and [36], the APE and RTE values are similar. However, the GOSPA scores are higher than those reported in [69]. In the first set of simulations involving relatively simple scenarios in [69], typical GOSPA values are in the range of 2 to 3. As the following discussion will explain, this increase here is primarily due to missed detections and false alarms for certain moving objects.

For an example of visual evaluation, the estimated trajectories for the ego-vehicle and moving objects when the ego-velocity is 10 m/s are shown in Fig. 4.2.1. The figure only displays results in the X and Y dimensions for simplicity in visualization, while all quantitative metrics are computed in the full 3D space.

For the ego-vehicle (labeled as "Ego"), the estimated trajectory closely matches the ground truth, indicating that the ego-motion estimation method described in Section 3.4 is capable of effectively distinguishing between static and moving objects, and then accurately estimating the ego-motion based on observations of static targets.

In the right part of the scene, five moving objects (labeled as '1' to '5') are accurately tracked after ego-motion compensation and multiple object tracking. No false tracks are observed in their vicinity. Although the simulation does not incorporate object size modeling, the tracking performance remains accurate for objects with various velocities, with objects 1 and 4 representing cars, object 2 representing a bicycle, and objects 3 and 5 representing pedestrians.

However, in the left part of the scene, where three moving objects (labeled as '6' to '8') are located closer to the ego-vehicle, the tracking results show noticeable deviations from the ground truth. For objects 6 and 7, missed detections occur during the initial phase of their motion. This is because their movement directions are nearly perpendicular to the line of sight of the radar, resulting in a Doppler radial velocity component close to zero. Consequently, these objects are initially misclassified as static and not included in the tracking process. As the relative geometry between object and radar changes during motion, the Doppler component increases, enabling the algorithm to recognize them as dynamic and begin outputting their trajectories.

For object 8, tracking is accurate in the early phase, but deviates from the ground truth in the later phase. Moreover, several false alarms emerge near this object, leading to false tracks that degrade tracking performance and worsen the value of the GOSPA metric. This issue is likely caused by limitations of the radar simulator described in Section 4.1. In real-world settings, a corner radar mounted at the front-left of a vehicle has a limited field of view, and can only observe objects within this region. During the simulation, object 8 is gradually overtaken by the ego-vehicle, and eventually exits the

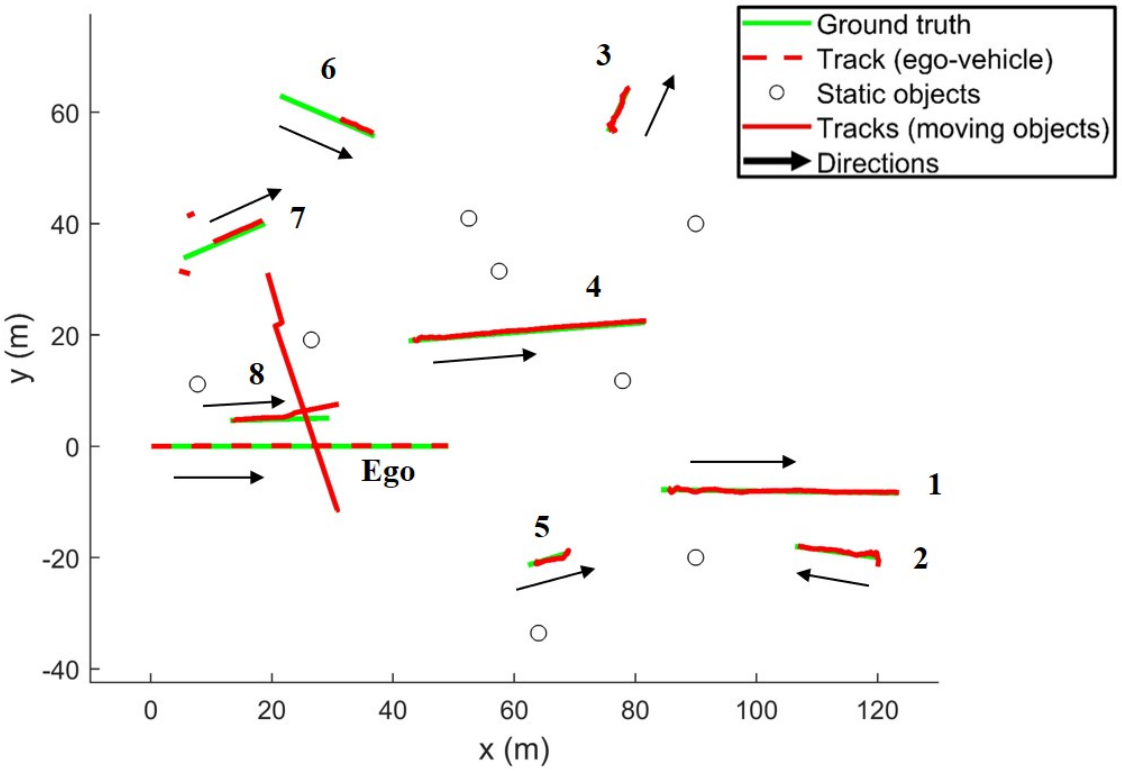


Figure 4.2.1: Ground truth and estimated trajectories for the ego-vehicle and eight moving objects from an example scenario when the ego-velocity is 10 m/s

radar's FoV. As a result, no measurements should be available for this object in the latter frames. However, the simulator in its current version lacks logic to determine whether an object lies within the radar's FoV and instead assumes all objects are always observable. This discrepancy between simulated and realistic observations leads to confusing input data and, consequently, inaccurate tracking estimates.

Overall, the simulation results demonstrate the good performance of the combined method proposed in Chapter 3 for both ego-motion estimation and multiple object tracking. As discussed in Section 1.2, methods based on radar raw signals are highly innovative and show great potential for future research. The algorithm presented in this part provides a preliminary validation of the effectiveness of a combined estimation framework.

However, the work on the raw-signal-based method developed in this part of the thesis has some limitations. It is so far theoretical and shows a noticeable gap from real-world driving scenarios. First, due to the lack of comprehensive real-world datasets, the algorithm is only evaluated using simulated data. As discussed previously, the radar simulator itself also introduces unrealistic assumptions. Second, the algorithm does not

account for the varying shapes and sizes of objects, which is not realistic for real-world applications.

Rather than focusing on improving the simulator, which would lead to an entire different research direction, and to address the discussed limitations while enhancing the practical applicability of the proposed approach, Part III introduces a novel point-cloud-based combined method. This method can be more easily validated using both simulated and real-world radar data, aiming to bridge the gap between theoretical development and real-world deployment.

Part III

Point Cloud based Combined Method

5 Methodology

In Part II of this thesis, a combined method based on radar raw signals was introduced. However, this approach faces several limitations when applied to real-world driving scenarios. To address this, Part III explains and validates another proposed method for combined ego-motion estimation & multiple object tracking that operates on radar point clouds. This approach is more practically applicable and better suited for deployment in real driving environments.

This chapter details the point cloud based combined method, with a particular focus on its differences and improvements against the method introduced in Chapter 3. Section 5.1 discusses the advantages of using point cloud as algorithm input and formulates the problem that the proposed method aims to solve. Section 5.2 provides an overview of the overall method. Subsequently, Sections 5.3 and 5.4 describe the tracking-aided ego-motion estimation and multiple extended object tracking modules, respectively, highlighting the key enhancements made over the raw signal based approach.

5.1 Problem formulation

Although the raw signal based combined method described in Part II demonstrated promising performance in both ego-motion estimation and tracking, it faces two major limitations in real-world driving scenarios. First, it neglects the extent and shape variations of different moving objects. The method neither estimates geometric properties including size, shape, and orientation, nor leverages them for dynamic object removal. This becomes particularly problematic when tracking extended targets like trucks or buses, where their spatial structure is essential for accurate data association. Second, as discussed in Section 1.2, raw radar signals are rarely included in public automotive datasets due to their high storage and bandwidth requirements. This severely limits the applicability of raw signal based algorithms on real-world data.

These limitations motivate the development of another combined approach based on preprocessed radar point clouds. While this format sacrifices some low-level signal details, it is widely used in practice and enables efficient implementation, dataset compatibility, and scalable evaluation.

As discussed in the literature review of Chapter 2, existing research on automotive radar typically treats ego-motion estimation and multiple extended object tracking as two separate tasks. Most methods use radar point cloud data as input. While ego-motion estimation focuses on using detections from static objects to infer the ego vehicle's motion, tracking assumes known ego-motion and aims to associate and track the moving objects over time for environment perception. However, these two tasks are inherently interrelated, as both rely heavily on accurate segmentation between static and dynamic points.

In current radar-based ego-motion estimation, the most widely adopted method for static and moving point segmentation is RANSAC [26], which detects static points based on the sinusoidal relationship between Doppler/velocity and azimuth angle. This approach works well in typical driving scenarios where the majority of radar reflections originate from static objects, and only a small portion come from moving targets. However, as also observed in [9], the robustness of RANSAC degrades significantly in highly dynamic environments where moving points dominate.

A representative example of such cases arises when a large vehicle, such as a truck or a bus, approaches the ego vehicle from the opposite lane. During such encounters, the large physical size of the object generates a dense cluster of moving reflections while simultaneously occluding many static structures in the scene. Figure 5.1.1 shows a camera image from scene 1 of the RadarScenes dataset [77], where a large oncoming vehicle on the left is about to pass by the ego vehicle. As illustrated in Figure 5.1.2, for this example frame, the RANSAC method fails to correctly fit the curve corresponding to static reflections, due to the high proportion of moving points. As a result, many dynamic reflections from the moving truck are mistakenly classified as static (blue), while a significant portion of actual static reflections are incorrectly labeled as moving (red). This misclassification directly leads to a distorted curve fit and a substantial degradation in ego-motion estimation accuracy during this period.

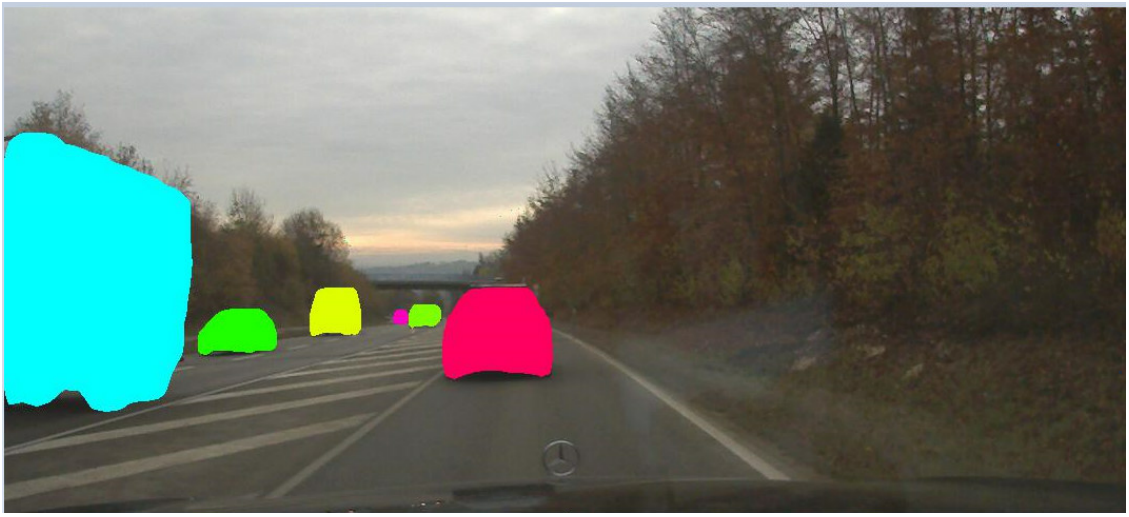


Figure 5.1.1: An image captured by camera in RadarScenes: scene 1, frame 1143, where a large vehicle on the left is about to pass by the ego vehicle

To address this issue, a tracking-aided combined method is proposed, which leverages historical object-level tracking information to enhance point cloud segmentation and ego-motion estimation. Considering the continuity and predictability of vehicle motion, prior tracking information from the previous frame can be utilized to identify potential dynamic points at the current time step, enabling a pre-filtering process. After removing the dynamic points, RANSAC can be applied to achieve accurate ego-motion estimation. This strategy significantly improves the robustness of ego-motion estimation in dynamic scenarios, while simultaneously producing accurate multiple extended

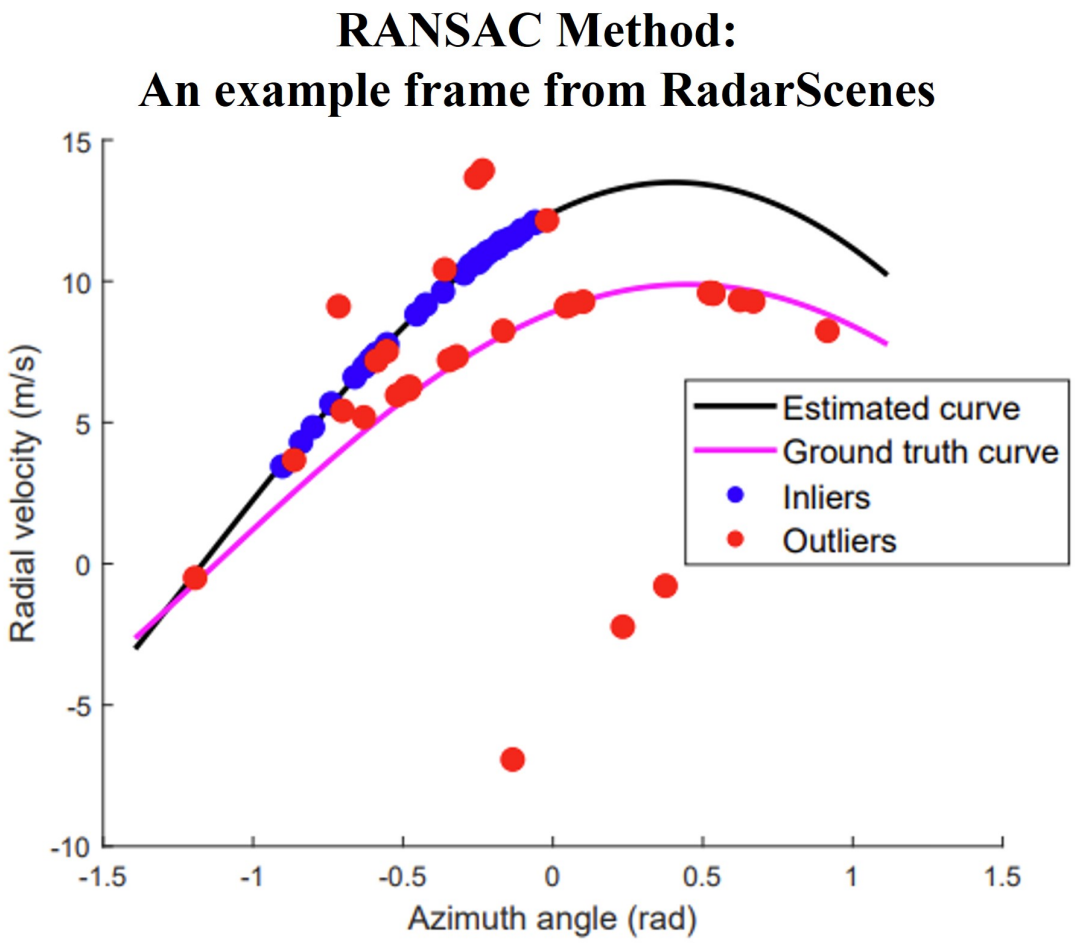


Figure 5.1.2: The problem of RANSAC algorithm for static vs dynamic points segmentation, as observed using the example frame in Figure 5.1.1

object tracking results.

In practical driving scenarios, the implementation of the combined method faces several challenges, including:

1. How to effectively use tracking information to guide the segmentation of moving and static points, thereby improving ego-motion estimation accuracy. In other words, the key question is how to exploit the point cloud from only a single radar to enhance the coupling between ego-motion estimation and multiple extended object tracking.

2. As discussed in Chapter 2, radar point cloud inherently contains a certain level of clutter due to the sensor characteristics. An essential design objective is to suppress the influence of clutter on the overall estimation performance.

3. Due to the sparsity of radar point clouds, estimating the physical extent (i.e., size and orientation) of moving targets remains a challenging and active research

problem.

Due to constraints in time and available datasets for this thesis, in contrast to the raw signal based method introduced earlier, the point cloud based method presented here in part III focuses on solving the problem in 2D plane (x- and y-axis). Specifically, it is assumed that radar point clouds do not include elevation angle measurements, and thus vertical (z-axis) motion estimation is not considered. This fits well with the data collected in the previously mentioned RadarScenes dataset, that will be used for the validation of the work in this part of the thesis. Extending the proposed method to solve the problem in a full 3D space is left for future work.

5.2 Algorithm overview

To enhance the robustness of RANSAC-based ego-motion estimation in dynamic driving scenarios while simultaneously addressing the problem of multiple extended object tracking, a frame-by-frame combined ego-motion estimation and multiple extended object tracking algorithm is proposed, which takes radar point clouds as input. The overall processing pipeline is illustrated in Figure 5.2.1. For each frame, the algorithm takes the radar point cloud as its input, and outputs the motion states of both the ego vehicle and the surrounding moving objects.

Similar to the raw signal based method overviewed in Section 3.2, the algorithm is divided into two phases in temporal order: the initialization phase and the combined estimation phase. The initialization phase ensures the establishment of initial object trajectories, which are required for assisting point cloud segmentation and ego-motion estimation in the combined estimation phase. For each frame, the combined estimation phase sequentially executes three modules: tracking-aided ego-motion estimation, ego-motion compensation, and multiple extended object tracking. While this method follows the overall structure introduced in Chapter 3, it differs in several key aspects:

1. Since this method uses radar point clouds as input, which have already undergone signal processing and target detection steps, it does not require a signal preprocessing step.

2. There are two differences with respect to the tracking-aided ego-motion estimation module of Chapter 3. In the tracking-aided segmentation step in Section 3.4.1, Euclidean gating is used to identify moving points, with each object gated using a uniform spherical threshold. However, this ignores the diversity in size and shape of moving targets in real-world road environments. Therefore, the method in this chapter improves the gating strategy by employing Mahalanobis gating with an adaptive covariance matrix, allowing the gating region to adapt to the shape and size of each tracked object. This leads to more accurate segmentation of static and dynamic points, and is better suited to real-world scenarios. Moreover, instead of the iterative ego-motion estimation based on raw signals described in Section 3.4.2, this method applies the RANSAC algorithm to estimate ego-motion from point cloud data after tracking-aided segmentation. These differences are explained in details in Section 5.3.

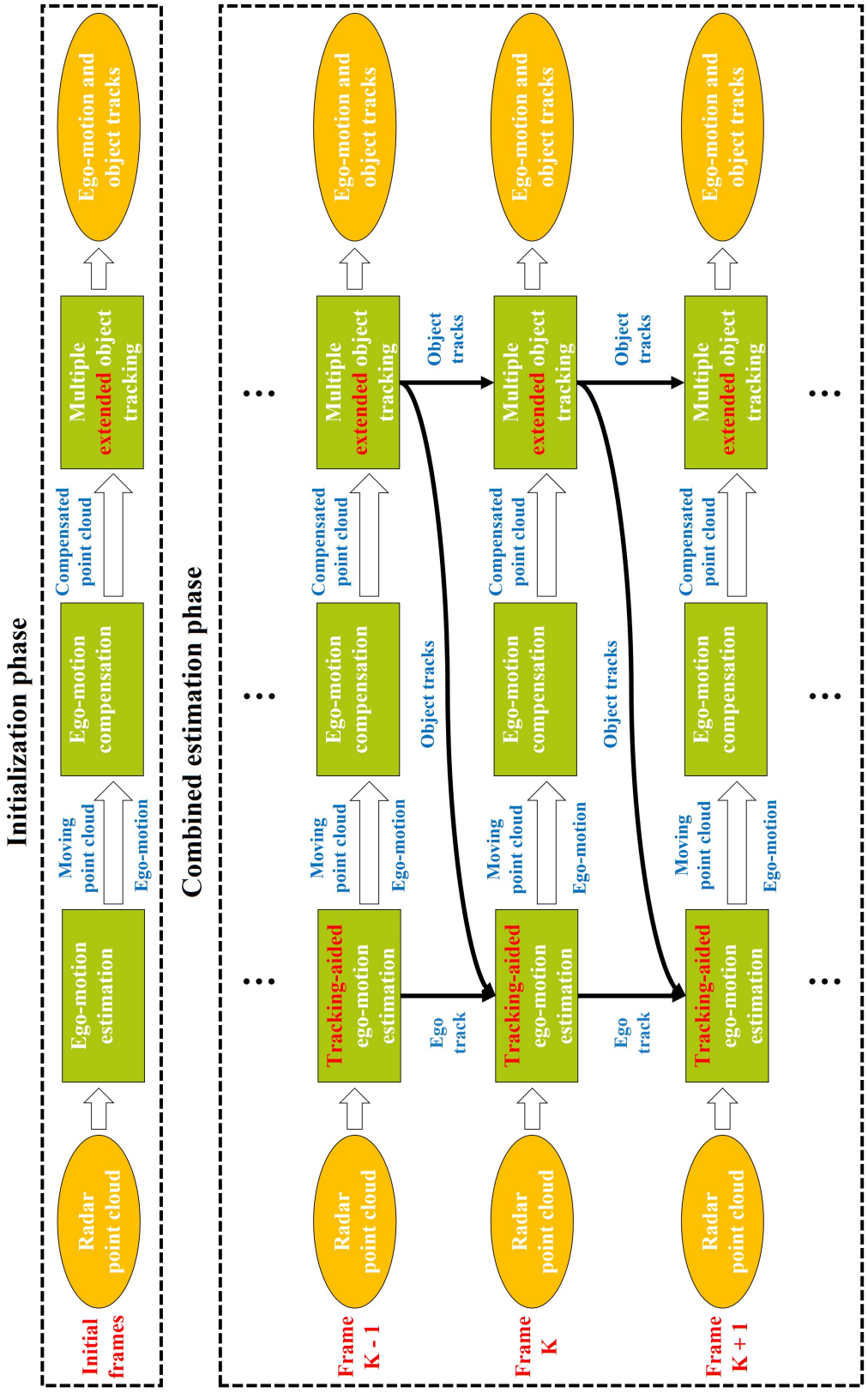


Figure 5.2.1: The overall processing pipeline of the point cloud based combined method for ego-motion estimation & multiple extended object tracking

3. In the multiple object tracking module of Chapter 3, each moving object was modeled as a point target, and only its position and velocity were estimated, without considering its spatial extent (e.g., size, shape, and orientation). To better support real-world driving applications, this method incorporates multiple extended object tracking, which additionally estimates the size and orientation of moving objects based on point cloud measurements. Size estimation not only enables a finer-grained understanding of the surrounding environment (which can be useful for later perception tasks), but also supports more precise segmentation in subsequent frames. Details of this improvements are provided in Section 5.4.

5.3 Tracking-aided ego-motion estimation

This section outlines two key differences compared to the raw signal based ego-motion estimation method introduced in Section 3.4. Specifically, Section 5.3.1 presents an adaptive Mahalanobis gating approach that accounts for the size and shape of each tracked object. Next, Section 5.3.2 describes the use of the RANSAC algorithm to perform ego-motion estimation based on the static points obtained from the tracking-aided segmentation.

5.3.1 Tracking-aided segmentation: adaptive gating

Motivation In radar-based ego-motion estimation, a crucial step is to distinguish between static and dynamic points in the point cloud. In the raw signal-based method introduced in Section 3.4, this is achieved using simple Euclidean distance-based gating, where each predicted object is associated with a fixed-radius spherical region to filter potential dynamic points. However, this approach assumes that all moving objects share the same size and shape, which is not consistent with real-world driving scenarios, where moving vehicles such as sedan cars, buses, and trucks vary significantly in spatial extent and orientation.

To address this limitation, an adaptive Mahalanobis gating strategy is proposed in this work, inspired by shape-aware data association and extended object tracking techniques in [38, 78, 79]. By incorporating each object's estimated size and orientation, the Mahalanobis distance allows the gating region to adapt to the object's actual geometry and uncertainty, significantly improving the accuracy of dynamic point identification.

The Mahalanobis distance is adopted over Euclidean distance for two main reasons:

- **Shape adaptation:** Mahalanobis distance defines elliptical gating regions that naturally align with the estimated shape and orientation of each object, unlike the isotropic and fixed-size spheres used in Euclidean gating.
- **Uncertainty awareness:** Mahalanobis distance incorporates the extent covariance, accounting for estimation uncertainty and enabling more robust point-to-object association under noise and partial occlusion.

Mathematical formulation Let $\mathbf{p}_i = (x_i, y_i)$ denote the Cartesian coordinates of the i-th radar detection point after ego-motion compensation. For each tracked object j

in the previous frame, its predicted centroid position in 2D $\boldsymbol{\mu}_j \in \mathbf{R}^2$ is obtained from the Kalman filter, and its 2D spatial gating region using an adaptive covariance matrix $\boldsymbol{\Sigma}_j \in \mathbf{R}^{2\times 2}$ is constructed. This matrix takes into account both the uncertainty of the estimated position and the spatial extent of the object.

The total covariance is computed as a summation of two components:

$$\boldsymbol{\Sigma}_j = \mathbf{P}_j^- + \mathbf{E}_j \tag{5.3.1}$$

where $\mathbf{P}_j^-$ is the predicted positional covariance matrix of object j from the Kalman filter, and $\mathbf{E}_j$ is the object's spatial extent matrix estimated by the method as in Section 5.4 (typically modeled as a 2D ellipse).

The squared Mahalanobis distance between the radar detection $\mathbf{p}_i$ and object j is defined as:

$$d^2_{(i,j)} = (\mathbf{p}_i - \boldsymbol{\mu}_j)^\top \boldsymbol{\Sigma}_j^{-1} (\mathbf{p}_i - \boldsymbol{\mu}_j) \tag{5.3.2}$$

This metric essentially quantifies how likely the detection $\mathbf{p}_i$ belongs to the distribution of spatial measurements associated with object j, under the Gaussian assumption.

A detection $\mathbf{p}_i$ is labeled as a moving point if it lies within the elliptical gating region of any object, i.e., there exists at least one object j for which:

$$d^2_{(i,j)} \leq \gamma \tag{5.3.3}$$

where γ is the gating threshold, usually determined based on a chi-squared distribution with 2 degrees of freedom [38].

Otherwise, if $d^2_{(i,j)} > \gamma$ for all j, the point $\mathbf{p}_i$ is classified as static. This gating process is repeated for each detection in the point cloud, resulting in two disjoint subsets: $\mathcal{P}_{\text{static}}$ and $\mathcal{P}_{\text{moving}}$. The static set provides input to the RANSAC-based ego-motion module of Section 5.3.2, while the moving set is passed to the ego-motion compensation and tracking modules.

Compared to Euclidean gating, which applies a uniform radius to all objects regardless of their geometry, Mahalanobis gating defines object-specific elliptical gating regions that are both shape-aware and uncertainty-aware. This allows for more precise segmentation, particularly in dynamic environments where vehicles may vary in size, orientation, or occlusion state.

Implementation Algorithm 2 summarizes the proposed tracking-aided segmentation approach based on adaptive Mahalanobis gating. The procedure includes three stages: prediction, compensation, and gating, which follow the structure of the raw signal-based approach. In the specific implementation, the gating threshold γ is set to 3.22, which corresponds to the gating probablity of 0.8, as determined by the inverse chi-squared distribution $\chi^2_2(0.8)$. This relatively loose threshold is selected in this work via hyperparameter tuning to balance recall (capturing valid moving points) and precision (excluding static points).

Algorithm 2 Tracking-Aided Segmentation (Point Cloud Approach)

Require: Detected point cloud at the current frame $\mathcal{P} = \{(\hat{R}_i, \hat{V}_{D,i}, \hat{\theta}_i\}_{i=1}^{N}$; ego vehicle track from the previous frame $\{\hat{\mathbf{x}}_{ego}, \mathbf{P}_{ego}\}$; object tracks from the previous frame $\{\hat{\mathbf{x}}_j, \mathbf{P}_j, \mathbf{E}_j\}_{j=1}^{M}$; the rotation matrix $\mathbf{R}_{\text{r2v}}$ and the translation vector $\mathbf{t}_{\text{r2v}}$ from radar to vehicle coordinate system; the rotation matrix $\mathbf{R}_{\text{v2w}}$ from vehicle to world coordinate system

Ensure: Static point indices $\mathcal{I}_{\text{static}}$, moving point indices $\mathcal{I}_{\text{moving}}$

Step 1: Prediction

Predict the ego vehicle state $\hat{\mathbf{x}}_{ego}^{-}$ and covariance matrix $\mathbf{P}_{ego}^{-}$ using Kalman filter
Extract predicted position of the ego vehicle $\mathbf{x}_{ego} = \hat{\mathbf{x}}_{ego}^{-}(1{:}2)$
for each object track $j = 1$ to M **do**
  Predict object state $\hat{\mathbf{x}}_j^{-}$ and covariance matrix $\mathbf{P}_j^{-}$ using Kalman filter
  Extract predicted position $\boldsymbol{\mu}_j = \hat{\mathbf{x}}_j^{-}(1{:}2)$
  Compute inflated covariance $\boldsymbol{\Sigma}_j = \mathbf{P}_j^{-} + \mathbf{E}_j$
  Compute inverse $\boldsymbol{\Sigma}_j^{-1}$
end for

Step 2: Compensation

Convert point cloud $\mathcal{P}$ to Cartesian coordinates $\mathbf{p}_{ir} = (x_i, y_i)$
Switch to the vehicle coordinate: $\mathbf{p}_{iv} \leftarrow \mathbf{R}_{\text{r2v}}(1{:}2, 1{:}2) \cdot \mathbf{p}_{ir} + \mathbf{t}_{\text{r2v}}(1{:}2)$
Apply ego-motion compensation: $\mathbf{p}_i \leftarrow \mathbf{R}_{\text{v2w}} \cdot \mathbf{p}_{iv} + \mathbf{x}_{ego}$

Step 3: Gating

Initialize $\mathcal{I}_{\text{static}} \leftarrow \{1, 2, \ldots, N\}$
for each point $\mathbf{p}_i$ **do**
  for each object $j = 1$ to M **do**
    Compute Mahalanobis distance $d_{(i,j)}^2 = (\mathbf{p}_i - \boldsymbol{\mu}_j)^\top \boldsymbol{\Sigma}_j^{-1} (\mathbf{p}_i - \boldsymbol{\mu}_j)$
    if $d^2 < \gamma$ **then**
      Point $\mathbf{p}_i$ is inside the gating region
      Mark point $\mathbf{p}_i$ as moving: remove i from $\mathcal{I}_{\text{static}}$
      break
    end if
  end for
end for
$\mathcal{I}_{\text{moving}} \leftarrow$ indices not in $\mathcal{I}_{\text{static}}$
return $\mathcal{I}_{\text{static}}, \mathcal{I}_{\text{moving}}$

5.3.2 Ego-motion estimation with RANSAC

Motivation After removing dynamic points through tracking-aided segmentation, the ego-motion estimation task is performed using the RANSAC algorithm [26], which is well-known for its robustness to outliers.

Originally developed for computer vision tasks, RANSAC is designed to handle datasets contaminated with a significant portion of outliers. In radar-based ego-motion estimation, even after segmentation, the remaining static points may still include residual clutter or misclassified points. Unlike least squares estimation, which is sensitive to such errors, RANSAC is able to identify a consensus set of inliers that best fits the

assumed motion model. This makes it especially suitable for urban or dynamic highway environments, where partial occlusion, clutter, and radar artifacts are common.

Mathematical Formulation Each static point after segmentation in frame k is represented as a triplet $(\hat{R}_i^k, \hat{\theta}_i^k, \hat{V}_{D,i}^k)$, where $\hat{R}_i^k$ is the measured range, $\hat{\theta}_i^k$ is the measured azimuth angle (in radians), and $\hat{V}_{D,i}^k$ is the observed radial velocity. The goal of instantaneous ego-motion estimation is to use these segmented static points to estimate the ego vehicle's 2D velocity vector in the radar observation coordinate system, i.e.:

$$\hat{\mathbf{v}}_k = \begin{bmatrix} \hat{v}_x^{ego,radar} \\ \hat{v}_y^{ego,radar} \end{bmatrix} \tag{5.3.4}$$

where $\hat{v}_x^{ego,radar}$ and $\hat{v}_y^{ego,radar}$ denote the velocity components along the radar's local x and y axes.

For a static target located at azimuth angle $\hat{\theta}_i^k$, the expected velocity due to ego-motion follows the radial projection:

$$\hat{V}_{D,i}^k = -\mathbf{a}_i^\top \hat{\mathbf{v}}_k + \epsilon_i, \quad \text{with} \quad \mathbf{a}_i = \begin{bmatrix} \cos(\hat{\theta}_i^k) \\ \sin(\hat{\theta}_i^k) \end{bmatrix} \tag{5.3.5}$$

where ϵ_i represents measurement noise, and the unit vector $\mathbf{a}_i$ describes the direction of the detection from the radar. The negative sign reflects the convention that Doppler is positive when the object is approaching.

By stacking N such measurements, a linear system is obtained:

$$\mathbf{d} = -\mathbf{A} \cdot \hat{\mathbf{v}}_k + \boldsymbol{\epsilon} \tag{5.3.6}$$

where

$$\mathbf{d} = \begin{bmatrix} \hat{V}_{D,1}^k \\ \hat{V}_{D,2}^k \\ \vdots \\ \hat{V}_{D,N}^k \end{bmatrix}, \quad \mathbf{A} = \begin{bmatrix} \cos(\hat{\theta}_1^k) & \sin(\hat{\theta}_1^k) \\ \cos(\hat{\theta}_2^k) & \sin(\hat{\theta}_2^k) \\ \vdots & \vdots \\ \cos(\hat{\theta}_N^k) & \sin(\hat{\theta}_N^k) \end{bmatrix} \tag{5.3.7}$$

Without outliers, the optimal velocity can be estimated using the ordinary least squares method as:

$$\hat{\mathbf{v}}_k = -(\mathbf{A}^\top \mathbf{A})^{-1} \mathbf{A}^\top \mathbf{d} \tag{5.3.8}$$

However, since outliers may exist due to clutter or misclassified dynamic points, the RANSAC algorithm is employed to robustly estimate $\hat{\mathbf{v}}_k$. In the r-th iteration, RANSAC performs the following steps:

1. Randomly select a minimal subset of M anchor points.
2. Estimate velocity $\hat{\mathbf{v}}_k^{(r)}$ using Equation 5.3.8.
3. Compute residuals $\hat{V}_{D,i}^k + \mathbf{a}_i^\top \hat{\mathbf{v}}_k^{(r)}$ and determine inliers under a threshold τ.
4. If the number of inliers exceeds the previous iterations, retain the current inlier, set as the best inlier set.

Once the best inlier set is determined, a final least-squares estimation is performed using only the inlier points to produce the final ego-motion estimate. This RANSAC-based estimation greatly improves robustness against outliers, especially in dynamic and clutter-intensive scenarios.

Compared to the iterative raw signal based method in Section 3.4.2, this RANSAC-based estimation is simpler, more efficient, and directly compatible with point cloud data. It avoids the need for Doppler FFT alignment or complex angle optimization. Moreover, thanks to the segmentation step using historical tracking, the RANSAC model receives a cleaner subset of static points, making the estimation more reliable.

Implementation In the proposed implementation, the key hyperparameters in RANSAC are configured as follows:

- The number of anchor points used in each iteration is set to $M = 5$.
- The desired inlier probability is $P_{inlier} = 0.99$, indicating a 99% confidence that at least one of the iterations yields a valid model.
- The assumed inlier ratio is $w = 0.3$, meaning at least 30% of the data is expected to be consistent with a correct model.
- The maximum allowable velocity residual error for inliers is $\tau = 0.1$ m/s.

Given the above configuration, the total number of RANSAC iterations K is computed using the standard probabilistic RANSAC formula:

$$K = \left\lceil \frac{\log(1 - P_{inlier})}{\log(1 - w^M)} \right\rceil \tag{5.3.9}$$

With the chosen hyperparameters, RANSAC can yield highly accurate velocity estimates and consistently converge to a stable estimate.

After RANSAC enables to compute the ego-vehicle velocity in the radar observation coordinate, as illustrated in Section 3.4.3, velocity transformation and state update are performed to obtain the final ego-motion estimation results.

5.4 Multiple extended object tracking

This section illustrates the implemented method for spatial extent estimation of moving objects, which enables the estimation of each moving object's size and orientation angle. For ego-motion compensation and kinematic state tracking of moving objects, the same approach as the one described for the raw signal method in Sections 3.5 and 3.6 is adopted.

Motivation Accurate estimation of the size and orientation of moving objects plays a crucial role in understanding the scene geometry and supporting perception tasks such as tracking, behavior prediction, and motion planning. In radar-based automotive systems, each object often produces multiple detection points, forming clusters that

reflect the object's spatial extent. Estimating the extent of each object allows the system to go beyond point-target models and better characterize real-world vehicles, especially in denser traffic scenarios.

Compared to simpler point-based representations, extended object models allow for more accurate data association and better gating in subsequent frames. Inspired by methods in extended object tracking in the literature [38, 44], this work proposes an efficient and robust size estimation method that relies on Minimum Volume Enclosing Ellipse (MVEE) fitting [80] and exponential smoothing. To better approximate the real shapes of objects in road scenarios, each object in this work is modeled as an ellipse. The spatial extent estimation then focuses on estimating the ellipse's major axis, minor axis, and orientation angle.

Mathematical Formulation Let $\mathcal{C}_k = \{\mathbf{z}_i = (x_i, y_i)\}_{i=1}^{N_k}$ denote the set of N_k radar detection points in the k-th cluster associated with a moving object, after DBSCAN clustering as illustrated in Section 3.6.1. The objective is to estimate a compact elliptical region that tightly encapsulates the spatial distribution of these points.

Each object is modeled as an ellipse in 2D space, defined by a center $\boldsymbol{\mu}_k \in \mathbf{R}^2$ and a symmetric positive-definite matrix $\boldsymbol{\Sigma}_k \in \mathbf{R}^{2\times 2}$, called the extent matrix. The center $\boldsymbol{\mu}_k$ is at the intersection of the major axis and minor axis of the ellipse. The ellipse can be mathematically expressed as:

$$\mathcal{E}_k = \left\{\mathbf{z} \in \mathbf{R}^2 \mid (\mathbf{z} - \boldsymbol{\mu}_k)^\top \boldsymbol{\Sigma}_k^{-1} (\mathbf{z} - \boldsymbol{\mu}_k) \leq 1\right\} \tag{5.4.1}$$

The matrix $\boldsymbol{\Sigma}_k$ encodes the size and orientation of the ellipse. Specifically, its eigenvalue decomposition is shown in Equation 5.4.2:

$$\boldsymbol{\Sigma}_k = \mathbf{R}\boldsymbol{\Lambda}\mathbf{R}^\top, \quad \text{where } \boldsymbol{\Lambda} = \begin{bmatrix} a^2 & 0 \\ 0 & b^2 \end{bmatrix}, \quad a \geq b > 0 \tag{5.4.2}$$

where:

- a: semi-major axis length
- b: semi-minor axis length
- θ: orientation angle, obtained from the rotation matrix $\mathbf{R}$. The rotation matrix $\mathbf{R}$ is constructed using the eigenvectors of $\boldsymbol{\Sigma}_k$, and θ is the angle between the x-axis and the eigenvector corresponding to a.

To estimate the extent matrix $\boldsymbol{\Sigma}_k$, the MVEE of the point set $\mathcal{C}_k$ is computed. This corresponds to the smallest-area ellipse (in 2D plane) that contains all the points. The MVEE is defined by the set:

$$\mathcal{E}_k = \left\{\mathbf{z} \in \mathbf{R}^2 \mid (\mathbf{z} - \boldsymbol{c})^\top \mathbf{A} (\mathbf{z} - \boldsymbol{c}) \leq 1\right\} \tag{5.4.3}$$

where $\mathbf{A} \in \mathbf{R}^{2\times 2}$ is positive-definite and $\boldsymbol{c}$ is the center. The relationship to the extent matrix is:

$$\boldsymbol{\Sigma}_k \approx \mathbf{A}^{-1}, \quad \boldsymbol{\mu}_k = \boldsymbol{c} \tag{5.4.4}$$

The MVEE is obtained by solving the following convex optimization problem [80]:

$$\min_{\mathbf{A}\succ 0,\, \boldsymbol{c}} \quad \log\det \mathbf{A}^{-1} \tag{5.4.5}$$

$$\text{s.t.} \quad (\mathbf{z}_i - \boldsymbol{c})^\top \mathbf{A}(\mathbf{z}_i - \boldsymbol{c}) \leq 1, \quad \forall \mathbf{z}_i \in \mathcal{C}_k \tag{5.4.6}$$

This problem can be efficiently solved using the Khachiyan's algorithm [80].

In each frame, all clusters identified as moving objects are processed to estimate the measurements, including the center position and the extent matrix. Unlike Section 3.6.1, where the center position of each cluster was estimated using the mean value of all points, this chapter uses the center of the MVEE as the center position of the clusters. This choice provides better stability in the presence of partial observations or non-Gaussian point distributions. The resulting measurements are then associated with existing tracks through the state estimation, gating and data association, and tracking management modules, as described in Section 3.6.

Considering the history estimates, for each associated track, its extent matrix is updated using exponential smoothing as:

$$\boldsymbol{\Sigma}_k^{(t)} = (1-\rho)\cdot \boldsymbol{\Sigma}_k^{(t-1)} + \rho \cdot \hat{\boldsymbol{\Sigma}}_k^{\text{MVEE}}, \quad \rho \in [0,1] \tag{5.4.7}$$

where:

- $\hat{\boldsymbol{\Sigma}}_k^{\text{MVEE}}$ is the newly estimated MVEE extent matrix from the current cluster,
- $\boldsymbol{\Sigma}_k^{(t-1)}$ is the previous extent of the track at time $t-1$,
- $\boldsymbol{\Sigma}_k^{(t)}$ is the updated extent at the current time t,
- ρ is the smoothing factor.

This update strategy enables gradual refinement of each object's size estimate over time, accounting for new observations while preserving stability against frame-to-frame noise and scintillation effects in the radar point clouds due to changes of scatterers locations on the targets.

Implementation In this work, the smoothing factor is set to $\rho = 0.5$ after hyperparameter tuning. A larger value of ρ gives more weight to the most recent observation, allowing the extent estimation to quickly adapt to shape changes (e.g., when a vehicle appears from occlusion). However, it also increases sensitivity to noisy or incomplete measurements. Conversely, a smaller ρ stabilizes the estimate but may lag behind rapid changes. Empirically, the chosen value provides a good trade-off between adaptivity and robustness in dynamic driving scenarios.

In the specific implementation, there is also a fallback strategy to avoid that there are too few points in one cluster. If a cluster contains too few points (e.g., $N_k < 5$), solving MVEE may be unstable. In such cases, the fallback strategy is to compute the sample covariance matrix of the points as:

$$\boldsymbol{\Sigma}_k = \text{cov}(\mathcal{C}_k) + \epsilon \mathbf{I} \tag{5.4.8}$$

where $\epsilon > 0$ is a small regularization constant to ensure $\mathbf{\Sigma}_k$ remains positive-definite. In this implementation, it is set to 10^{-3}.

In summary, this chapter first describes the ego-motion estimation challenge caused by large oncoming vehicles in practical driving scenarios. Next, the point cloud based combined method is presented in detail, focusing on the differences from the method presented in Chapter 3 which used raw data. Moving objects are modeled as ellipses, with their size and orientation angles estimated using the MVEE algorithm, thereby solving the full MEOT problem. With the available extent information, the tracking-aided ego-motion estimation utilizes adaptive Mahalanobis gating to precisely filter out moving points, leading to robust estimates. In the following Chapters 6 and 7, the method is validated through both simulation and a real dataset.

6 Simulation Results

To validate the feasibility and performance improvement of the point cloud based combined method for ego-motion estimation & multiple extended object tracking proposed in Chapter 5, this chapter conducts a series of tests using simulated radar point clouds. Section 6.1 introduces the simulation environment and the adopted evaluation metrics. In Sections 6.2 and 6.3, the method is tested on two representative autonomous driving simulation scenarios, with both qualitative and quantitative analyses provided. Finally, Section 6.4 extends the evaluation to a broader set of test cases to examine the generalizability of the proposed method.

6.1 Simulation Setup and Performance Metrics

To validate the performance of the combined method proposed in Chapter 5, a set of simulations were conducted based on realistic driving environments. To enhance the realism of the simulation, the Driving Scenario Designer application provided by MATLAB [81] is utilized for generating radar point cloud data under various autonomous driving scenarios.

The Driving Scenario Designer application, part of the MATLAB Automated Driving Toolbox [82], enables users to create synthetic and well-structured traffic scenes involving multiple road actors, road networks, and sensor configurations. This tool provides an intuitive interface for positioning vehicles, defining trajectories, and emulating sensor detections such as radar, LiDAR, and cameras. The underlying toolbox also supports perception, sensor fusion, motion planning, and control algorithm testing, making it a comprehensive platform for simulating and validating Advanced Driver Assistance Systems (ADAS) and autonomous driving systems. Several recent studies [83, 84] have adopted the Driving Scenario Designer application and Automated Driving Toolbox for prototyping and testing radar-based or vision-based perception pipelines, highlighting its effectiveness for algorithm development and early-stage validation in controlled conditions.

The basic simulation environment used in this Chapter is illustrated in Figure 6.1.1.

The simulated road is straight, with a total length of 150 meters and a total width of 14.55 meters. It consists of four lanes—two lanes in each direction—with a lane width of 3.6 meters. To emulate static roadside objects, ten metal guardrails (each 5 meters long, 0.43 meters wide, and 0.75 meters high) are placed symmetrically on both sides of the road with a constant interval of 30 meters.

The ego vehicle is modeled as a typical mid-size sedan with dimensions of 4.7 meters in length, 1.8 meters in width, and 1.4 meters in height. It travels in the left fast lane of the forward direction, starting from the position (1, –1.75) meters in the world

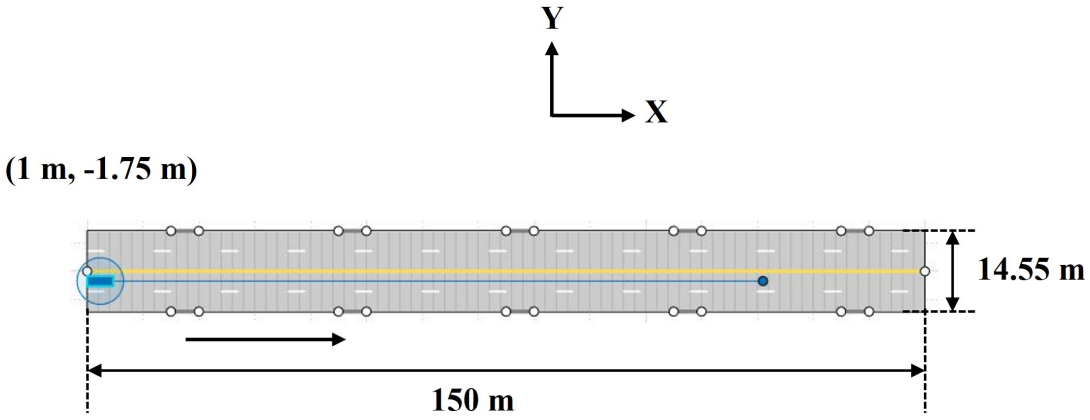


Figure 6.1.1: The basic simulation environment: road, static guardrails, and the ego vehicle in blue on the left-hand side

coordinate system. To simplify the analysis, the ego vehicle maintains a constant velocity of 12 m/s (equivalent to 43.2 km/h) along the forward direction (X-axis), with zero velocity in the lateral direction (Y-axis).

A single front corner automotive radar is installed on the front left part of the ego vehicle. To ensure realism and maintain consistency with the real-world dataset, the radar parameters are selected to be similar with those of Radar 3 in the *RadarScenes* dataset [77], which will be used in Chapter 7 for real-data validation. The detailed radar parameters used in the simulation are summarized in Table 6.1.1. As illustrated in Figure 6.1.2, the radar sensor is mounted with an azimuth angle of 25 degrees relative to the forward driving direction. The position offset from the vehicle center is 2.35 meters forward and 0.5 meters to the left. Compared to other mounting configurations such as forward-looking or side-looking radars, a front-corner radar is better suited for validating the proposed combined method, as it can simultaneously detect moving targets in front of the vehicle and static background objects along the roadside. The radar has a maximum detection range of 100 meters and an azimuth field of view of 120 degrees, providing broad angular coverage for both central and peripheral detections. To simulate the effects of missed detections, clutter, and measurement uncertainty in realistic radar point clouds, the simulation sets the detection probability to 0.9 and the false alarm rate to 1×10^{-7}. Each simulation trial covers 10 seconds, corresponding to 100 radar frames at a frame rate of 10 Hz.

To evaluate the performance of the proposed method in Chapter 5, six performance metrics are selected in total. For ego-motion estimation, the same two metrics used in the raw signal method are adopted, namely the APE and the RTE, as defined in Section 3.7.1. These metrics reflect the estimation accuracy of the instantaneous velocity and the long-term relative trajectory, respectively.

For multiple extended object tracking, a key distinction arises between the proposed raw signal method of the previous part of this thesis, and the point cloud method assessed here. While the former only tracks the kinematic states of objects, including their position and velocity, the latter is capable of estimating not only these kinematic

Table 6.1.1: Radar parameters in the 2D simulations run in Chapter 6

Parameters	**Value**
Angle with respect to the forward direction (degree)	25
Offset in the forward direction (m)	2.35
Offset in the lateral direction (m)	0.5
Maximum detectable range (m)	100
Range resolution (m)	0.15
Maximum detectable Doppler velocity (m/s)	30
Velocity resolution (m/s)	0.028
Field of view for azimuth angle (degree)	120
Azimuth angle resolution (degree)	1
Detection probability	0.9
False alarm rate	1×10^{-7}
Frame rate (Hz)	10

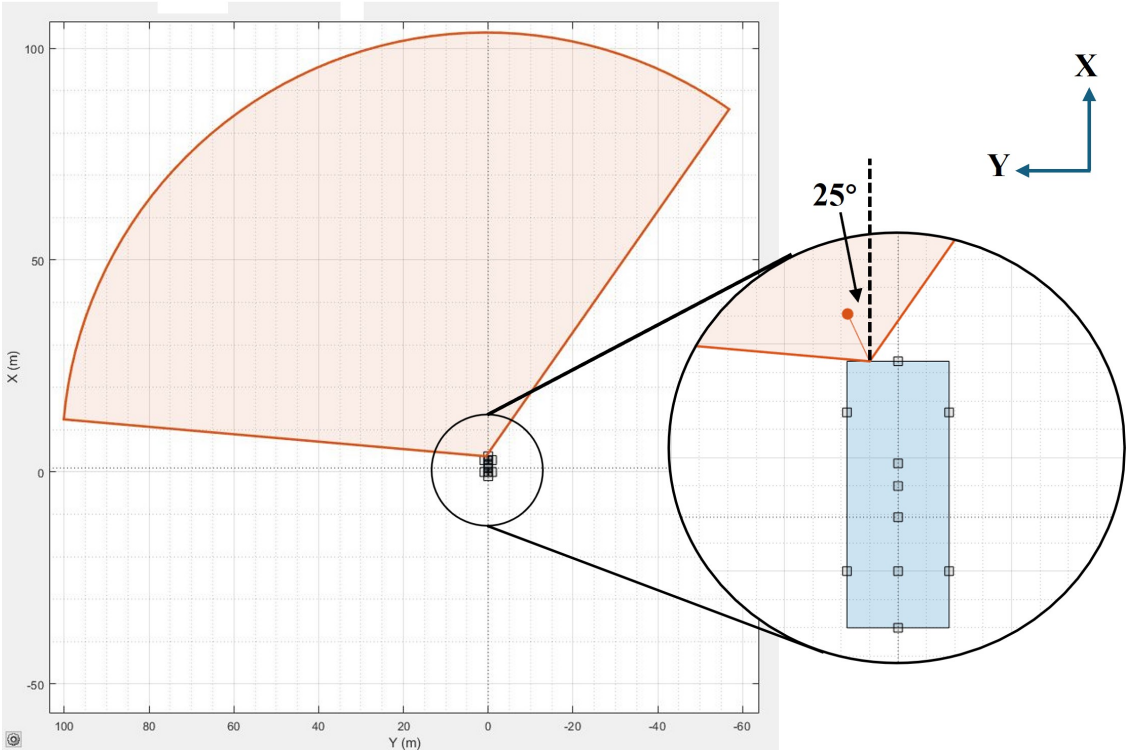


Figure 6.1.2: Installation configuration of the front-corner radar with its field of view

attributes but also the size and orientation of multiple extended targets. As a result, performance evaluation must include both kinematic tracking accuracy and size estimation accuracy. To assess the kinematic tracking accuracy, the GOSPA metric described in Section 3.7.2 is used again. Then, to assess the accuracy of shape estimation for multiple objects, each extended object is modeled as an ellipse, and three additional metrics are introduced:

1. The RMSE of the estimated semi-major axis length a

$$RMSE_a = \sqrt{\frac{1}{m}\sum_{i=1}^{m}\frac{1}{n(i)}\sum_{j=1}^{n(i)}\|\hat{a}(i,j) - a(j)\|^2} \tag{6.1.1}$$

where m denotes the total number of frames containing at least one extended object, and $n(i)$ represents the number of extended objects present in frame i. The terms $\hat{a}(i,j)$ and $a(j)$ refer to the estimated and ground truth semi-major axis lengths of object j at frame i, respectively. The Euclidean distance is computed using the L2 norm.

2. The RMSE of the estimated semi-minor axis length b

$$RMSE_b = \sqrt{\frac{1}{m}\sum_{i=1}^{m}\frac{1}{n(i)}\sum_{j=1}^{n(i)}\|\hat{b}(i,j) - b(j)\|^2} \tag{6.1.2}$$

where $\hat{b}(i,j)$ and $b(j)$ refer to the estimated and ground truth semi-minor axis lengths of object j at frame i, respectively.

3. The RMSE of the estimated orientation angle θ

$$RMSE_\theta = \sqrt{\frac{1}{m}\sum_{i=1}^{m}\frac{1}{n(i)}\sum_{j=1}^{n(i)}\|\hat{\theta}(i,j) - \theta(j)\|^2} \tag{6.1.3}$$

where $\hat{\theta}(i,j)$ and $\theta(j)$ refer to the estimated and ground truth semi-minor axis lengths of object j at frame i, respectively.

To compute these metrics, ground truth values for the three above parameters must be obtained. This is done by uniformly sampling 1000 points along the actual visible contour of each object and applying the MVEE fitting algorithm introduced in Section 5.4, to extract the semi-major axis length, semi-minor axis length, and orientation angle.

In the simulations, two types of extended targets are considered: cars and trucks. Considering the occlusion effect of the radar, the actual observed contour of a vehicle depends on its relative position to the ego vehicle. For example, as visualized in Figure 6.1.3, when a car (i.e., 4.7 m in length, 1.8 m in width) is located in front of the ego vehicle and to its right, due to partial occlusion, the radar can only receive reflections from the two exposed side edges. In this case, 1000 points are sampled along these edges and an ellipse is fitted to compute the ground truth values. The same method is applied for cars located in the left-front direction, as well as for trucks in both left-front and right-front positions. The resulting ground truth parameters for all four configurations are summarized in Table 6.1.2.

In the following simulation results, two representative scenarios are selected for detailed analysis: one involving a single oncoming truck, and another involving multiple oncoming trucks for example in a highway. As discussed in Section 5.1, the

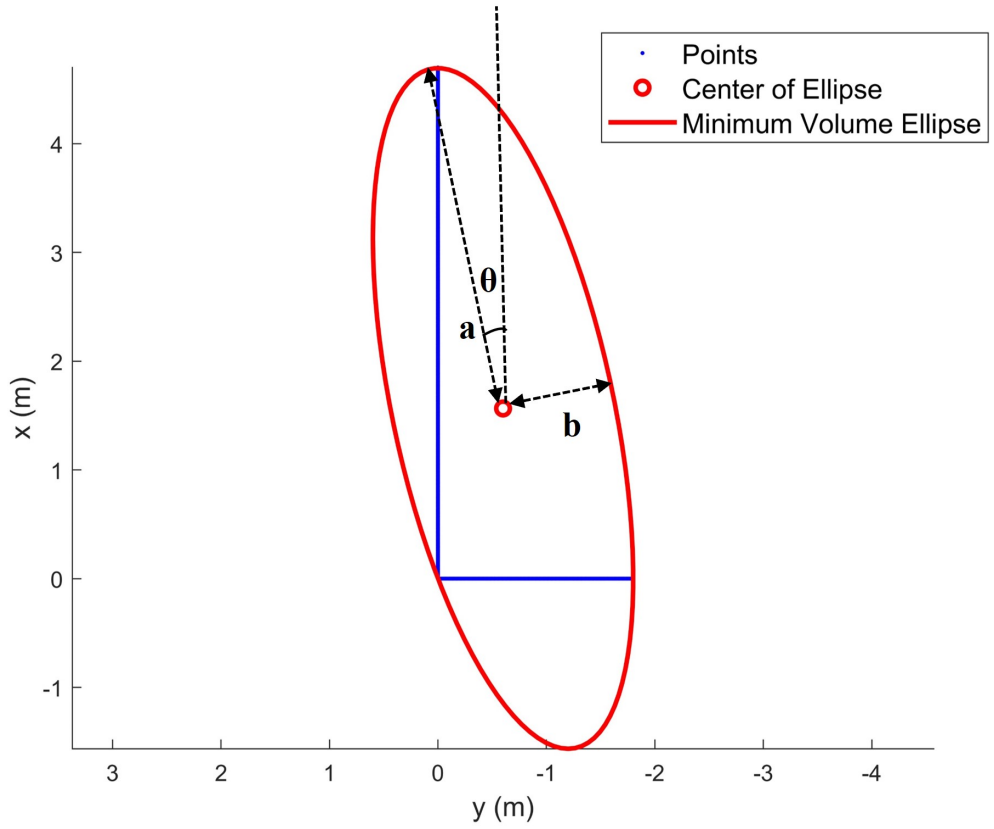


Figure 6.1.3: Example of ground truth generation for shape estimation: a right-front car, of which only two sides are visible to the radar on the ego-vehicle

Table 6.1.2: Ground truth values of shape estimation parameters

Object type	**Position**	**semi-major axis length (m)**	**semi-minor axis length (m)**	**orientation angle** (°)
Car	Right	3.19	1.02	12.11
Car	Left	3.19	1.02	-12.11
Truck	Right	5.53	1.43	9.31
Truck	Left	5.53	1.43	-9.31

RANSAC-based ego-motion estimation method is prone to failure in scenarios where large oncoming objects, such as trucks, dominate the radar measurements. In such cases, the proposed combined method is expected to demonstrate significant performance improvements with respect to conventional approaches. In addition to these two core scenarios, the proposed method is also evaluated under a wider variety of road scenarios and ego-vehicle motion patterns. The results of these extended simulations will be presented in Section 6.4.

6.2 Scene 1: A Single Truck

As shown in Figure 6.2.1, the first scenario contains four vehicles distributed across four different lanes. This setting is designed to simulate a typical driving scenario, where the ego vehicle meets oncoming vehicles, overtakes slower vehicles in the same direction, and coexists with multiple surrounding moving objects on a multi-lane road.

On the first oncoming lane closest to the ego vehicle, there is a yellow truck with dimensions of 8.2 meters in length, 2.5 meters in width, and 3.5 meters in height. It moves at a speed of 9 m/s, starting from the position (147, 1.5) m and eventually intersects with the ego vehicle. In the slow lane in the same direction as the ego vehicle, there is a red sedan with the same dimensions as the ego vehicle, moving at 8 m/s. It starts from (36, -5) m and is overtaken by the ego vehicle during the simulation. Additionally, on the second oncoming lane, there is another green sedan traveling at 6 m/s, with a starting position of (80, 5) m. It also meets the ego vehicle during the simulation. All vehicles move with constant velocities along straight-line trajectories. Their trajectories and endpoints are illustrated in the figure using lines and solid circles, respectively.

The simulation lasts for 10 seconds, corresponding to 100 radar point cloud frames. The simulated radar point cloud consists of a set of measurements including range, azimuth angle, and velocity which serve as the input to the combined method for ego-motion estimation & multiple extended object tracking:

$$\mathcal{P} = \{(R_i, \theta_i, V_{D,i})\}_{i=1}^{N} \tag{6.2.1}$$

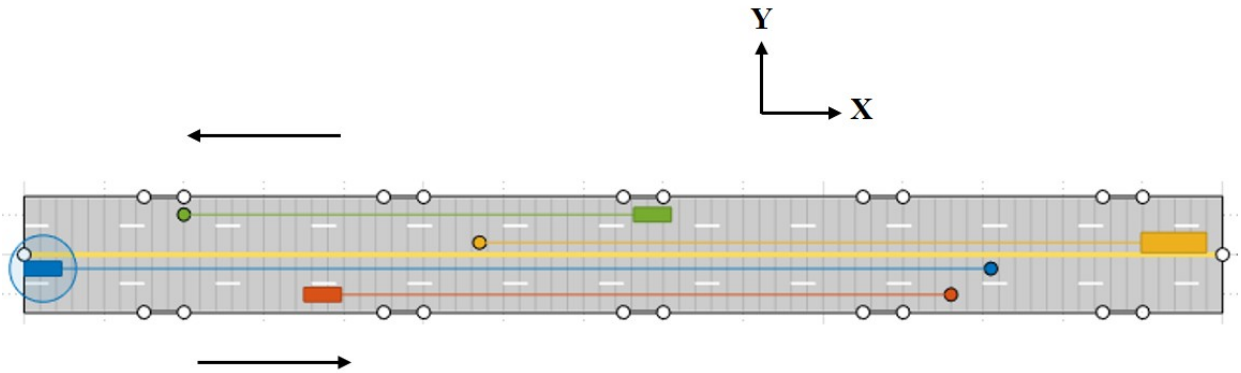


Figure 6.2.1: Simulated Scene 1: a single truck and two sedans

The results of the combined method proposed in Chapter 5 are presented next, from two perspectives: ego-motion estimation and multiple extended object tracking, in Sections 6.2.1 and 6.2.2, respectively. For ego-motion estimation, the combined method is compared with the RANSAC-based method to demonstrate the performance improvements brought by the combined approach, especially in separating static and moving targets. For multiple extended object tracking, both trajectory estimation and object size estimation are evaluated.

6.2.1 Ego-motion estimation

As discussed in Section 5.1, the performance of ego-motion estimation heavily relies on the accurate separation of static and moving targets. In highly dynamic scenes, such as when encountering an oncoming truck, a large portion of the detected radar point cloud originates from moving objects. In such cases, conventional RANSAC-based radar ego-motion estimation methods often fail to fit a correct model, resulting in significant degradation in terms of estimation accuracy.

To address this issue, the combined method proposed in Chapter 5 enhances the accuracy and robustness of ego-motion estimation by incorporating spatiotemporal information from object tracking to remove dynamic points. In the following, the segmentation results of static and moving targets obtained by the combined method are first presented, followed by a comparative evaluation of ego-motion estimation accuracy against the RANSAC-based method.

In the combined method, after the initialization phase, ego-motion estimation proceeds through three stages: tracking-aided segmentation of static and moving point clouds, ego-motion estimation via RANSAC, and coordinate transformation and state update. Figure 6.2.2 illustrates a representative frame where a large truck passes by the ego vehicle. The point cloud has been segmented using tracking information from the previous frame. Red circles indicate points identified as moving, while blue circles represent points classified as static. It can be observed that most of the points generated by the truck (located in the lower-left area) are correctly identified as moving by Mahalanobis gating. In contrast, the majority of points associated with static guardrails are accurately classified as static. It is worth noting that by the time the large truck meets the ego vehicle, the red and green sedans of this simulation have already moved behind the ego vehicle and thus fall outside the radar's field of view, resulting in no detections for these two objects in this frame. In summary, this effective separation of moving points helps eliminate their influence on subsequent model fitting, highlighting a key advantage of the proposed method.

In the second stage of ego-motion estimation, the method feeds all initially segmented static points into the RANSAC algorithm to estimate the ego vehicle's motion. Figure 6.2.3 compares the curve fitting results in the Doppler–azimuth plane obtained using the conventional RANSAC-based method (blue line) and the proposed combined method (red line). For the conventional approach that applies RANSAC directly to all detected points without tracking-aided segmentation, the curve fitting process fails when a large oncoming truck is present. In this case, more than 50% of the detections originate from dynamic objects, causing RANSAC to be biased and fit a curve that does not represent the static background. In contrast, the combined method successfully removes the majority of dynamic points through tracking-aided segmentation. As a result, RANSAC is able to fit a curve corresponding to truly static targets, leading to a more accurate ego-motion estimate.

As described in Section 6.1, ego-motion estimation performance is quantitatively evaluated using two metrics: APE and RTE. Table 6.2.1 compares the results obtained by the conventional RANSAC-based method and the proposed combined method for the entire scene. To account for the inherent randomness of the RANSAC algorithm, each result is averaged over 100 Monte Carlo trials. It can be observed that the proposed

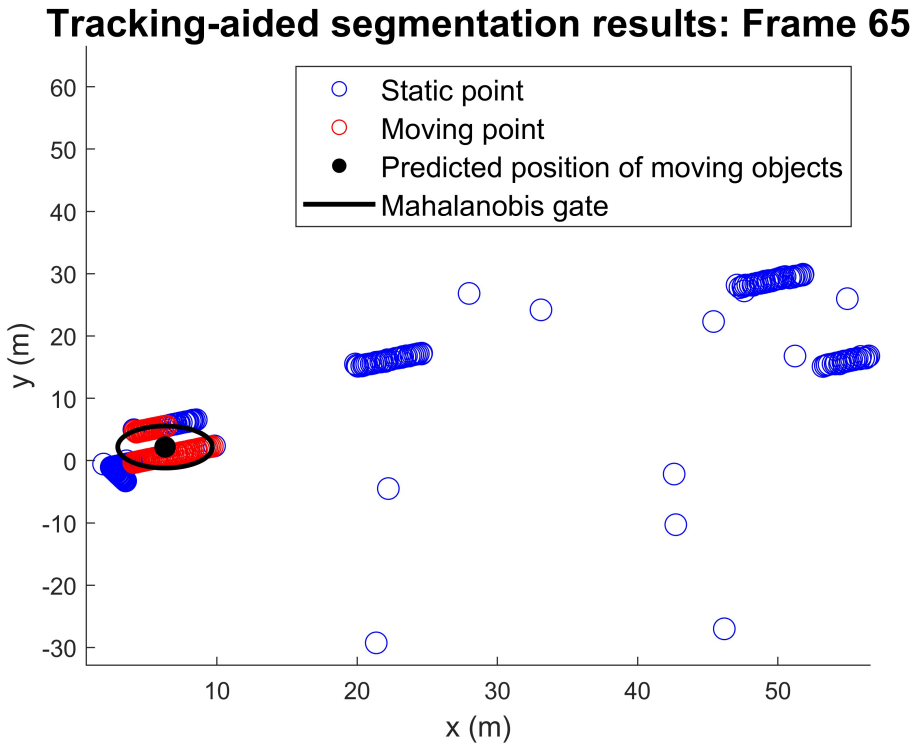


Figure 6.2.2: The tracking-aided segmentation results for an example frame: 2D positions of point cloud

combined method achieves a significant improvement over the conventional approach across the entire simulation period. Specifically, the APE is reduced from 1.79 m/s to 0.01 m/s, and the RTE is reduced from 111.58 cm to 0.37 cm. The substantial reduction in both metrics highlights the importance of correctly identifying and removing dynamic points during ego-motion estimation.

Table 6.2.1: Performance metrics of ego-motion estimation: a comparison between RANSAC-based method and combined method

Method	**APE (m/s)**	**RTE (cm)**
RANSAC-based method	1.79	111.58
Combined method	0.01	0.37
Absolute difference	-1.78	-111.21

To further analyze the results in greater detail, the APE of each individual frame over time is shown in Figure 6.2.4. For clarity and better visualization of variations, the APE is plotted on a logarithmic scale. As the figure shows, between frames 60 and 70, the large oncoming truck is approaching and overtaking the ego vehicle. During this highly dynamic interaction, the performance of the RANSAC-based method deteriorates significantly, with APE reaching values on the order of 10^2 m/s. Such large estimation errors are clearly unacceptable in practical autonomous driving applications.

In contrast, the proposed combined method maintains a stable estimation perfor-

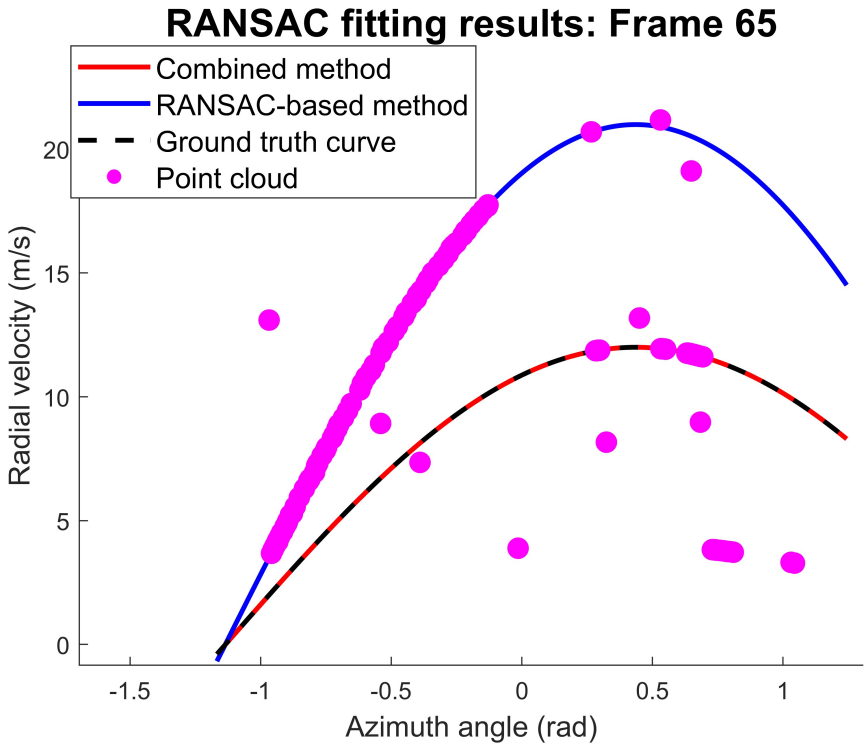


Figure 6.2.3: The RANSAC curve fitting results for an example frame (in the Doppler-azimuth plane)

mance across all frames, with APE consistently below 0.01 m/s. This demonstrates the method's robustness under challenging dynamic conditions. The key to this improvement lies in the effective removal of dynamic points using tracking-aided segmentation. Overall, the combined method can provide accurate and consistent ego-motion estimates.

6.2.2 Multiple extended object tracking

The performance of multiple extended object tracking is evaluated from two perspectives: kinematic state tracking and size estimation. Table 6.2.2 summarizes the performance metrics of the proposed combined method in the simulated Scene 1. For kinematic tracking, as introduced in Section 3.7.2, the GOSPA metric—widely used in radar tracking tasks—is employed. This metric jointly considers localization errors, false alarms, and missed detections, providing an overall performance assessment rather than individual target-level errors. For shape estimation, both the overall RMSE of the ellipse parameters and per-target RMSEs are reported. To ensure statistical reliability, all metrics represent the average over 100 Monte Carlo tests.

As shown in the table, in terms of kinematic tracking, the combined method performs accurately in this relatively simple scenario where objects are spatially well-separated and trajectories do not cross. The GOSPA score of 2.69 is consistent with results reported in [69] for the first simple scenario, where GOSPA typically lies be-

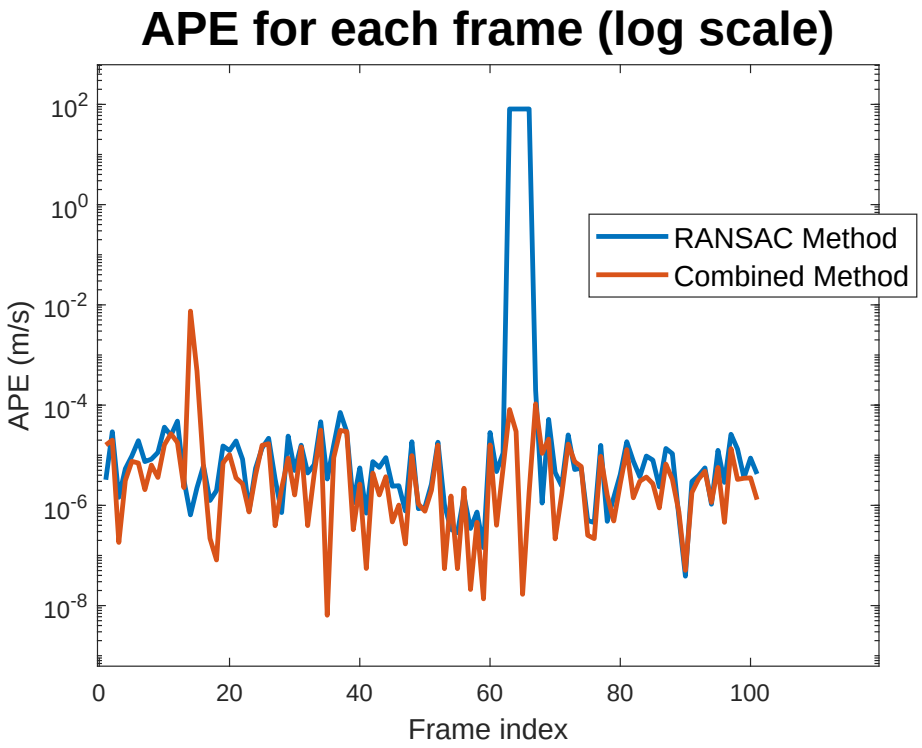


Figure 6.2.4: APE per frame for Scene 1: RANSAC-based method and proposed combined method (visualized in logarithmic scale)

Table 6.2.2: Performance metrics of multiple extended object tracking for simulated Scene 1

	Mean GOSPA	**RMSE$_a$ (m)**	**RMSE$_b$ (m)**	**RMSE$_\theta$ (°)**
All objects	2.69	1.19	0.46	6.12
Object 1 (Red sedan)		0.25	0.46	6.05
Object 2 (Yellow truck)		1.96	0.42	3.71
Object 3 (Green sedan)		1.54	0.41	6.88

tween 2 and 3. However, as also shown in [69], in more complex scenes—such as those with intersecting trajectories—the GNN-based data association used in the combined method tends to degrade, leading to higher GOSPA scores. To handle such challenges, more advanced multi-target tracking filters such as the PHD filter [73] or the PMBM filter [74] may be required, which will be considered in the future work.

For shape estimation, the RMSEs of the semi-major axis (a), semi-minor axis (b), and orientation angle (θ) are 1.19 m, 0.46 m, and 6.12°, respectively. These results indicate that the proposed method can accurately estimate the dimensions of vehicles with different sizes, orientations, and velocities within an acceptable margin of error. A closer inspection of the per-target results shows that the accuracy of b and θ estimation remains fairly consistent across all three objects. However, the estimation of a varies significantly.

The vehicle in the same direction as the ego vehicle (red sedan) exhibits much lower a error compared to the vehicles in the opposite direction (yellow truck and green sedan). This may be attributed to the fact that same-direction vehicles are generally closer to the ego vehicle, resulting in denser radar point clouds. Moreover, they remain in the radar's field of view for a longer period of time, which facilitates more stable and accurate estimation of object shape and size.

To visualize the tracking performance qualitatively, Figure 6.2.5 shows the estimated trajectories (solid red lines) and ground truth trajectories (solid green lines) for all three vehicles. The estimated ego-trajectory is also shown with a red dashed line, serving as a reference for ego-motion compensation. When the ego vehicle's motion is accurately tracked, the trajectories of other vehicles can also be consistently tracked by ego-motion compensation.

As the figure shows, all three vehicles are tracked with high accuracy during most of the frames, validating the feasibility and effectiveness of the proposed combined method. For the red sedan traveling in the same direction, two notable trajectory drifts are observed, along with a false alarm in the early phase. These errors are due to two short stationary guardrails located at 45–50 m and 75–80 m, with a lateral distance of only 2.2 m from the ego vehicle. When the red sedan passes near these structures, the combined method may mistakenly group them as one object and classify parts of the guardrail as moving, resulting in degraded tracking accuracy. For the yellow truck and green sedan, the estimated trajectories initially estimate the y-position with a slightly large error and gradually converge to the true values. This effect is more pronounced for the truck. A possible explanation is that when the objects first appear in the radar's field of view, the number of available detections is limited and their spatial distribution may be sparse. Additionally, occlusion effects can cause uncertainty in lateral localization.

6.3 Scene 2: Multiple Trucks

As shown in Figure 6.3.1, different from the first scenario, the second simulated scenario features a multi-truck setting. It includes three trucks (yellow, green, and purple) traveling in the opposite direction of the ego vehicle.

The trucks are equally spaced as in a sort of platoon and move at a constant velocity. Their initial positions are (150, 2) m, (120, 1.5) m, and (90, 1.8) m, respectively, with an inter-vehicle spacing of 30 m. All vehicles travel in straight lines at a constant speed of 6 m/s. To facilitate visualization of the tracking results, the y-coordinates of the trucks differ slightly, though they all occupy the same lane. The trajectories and endpoints of the trucks are shown as straight lines and solid circles, respectively. During the simulation, the three trucks sequentially pass by the ego vehicle. The simulation duration is 10 seconds, resulting in 100 frames of radar point cloud data.

As in the previous scenario, the results of the proposed method are presented from two perspectives: ego-motion estimation and multiple extended object tracking, in Sections 6.3.1 and 6.3.2, respectively.

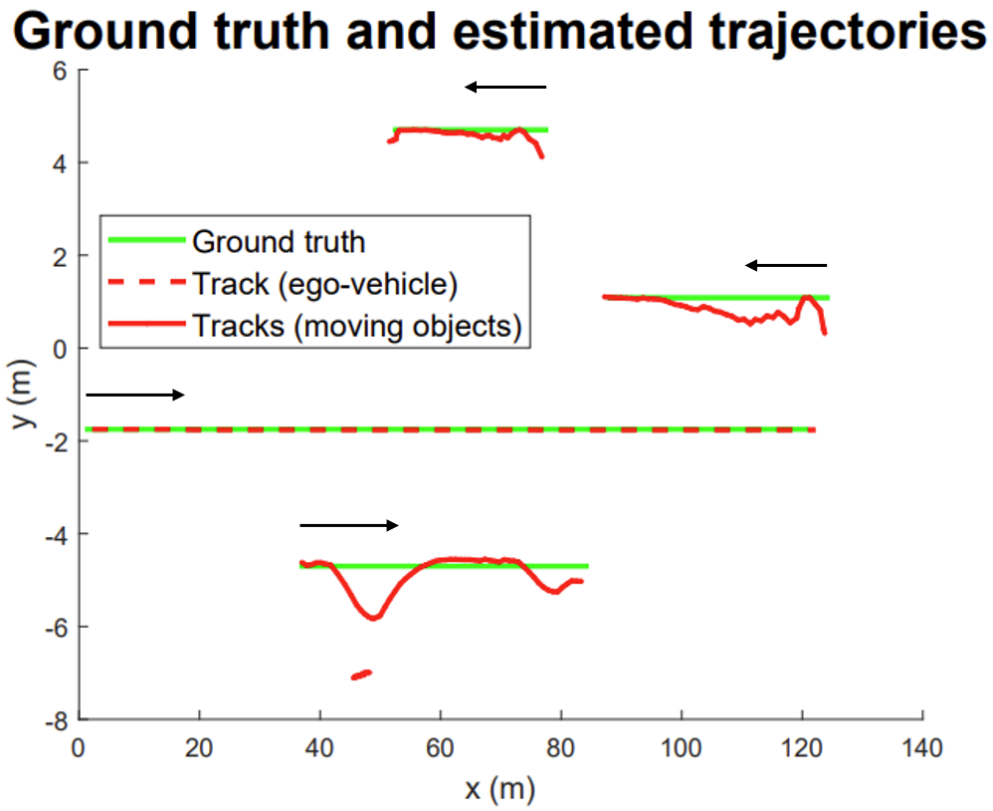


Figure 6.2.5: Tracking results for simulated Scene 1: ground truth and estimated trajectories

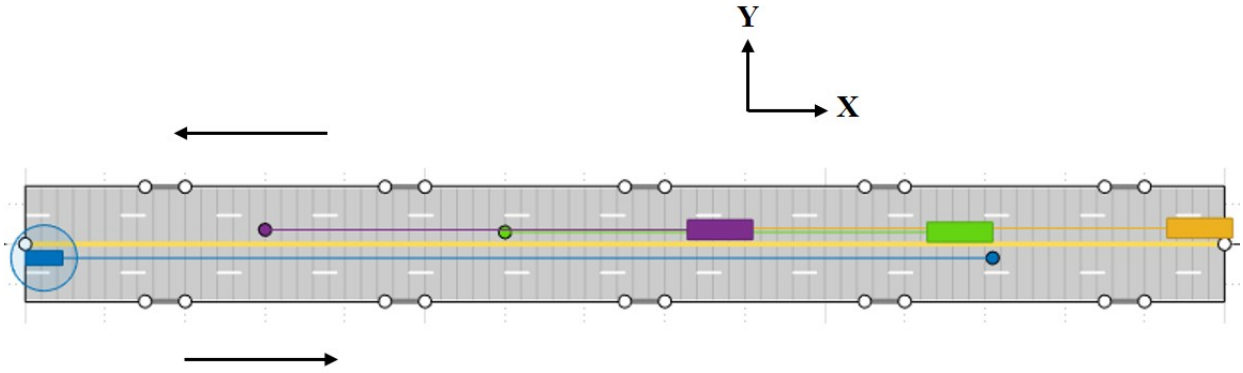


Figure 6.3.1: Simulated Scene 2: three equally-spaced trucks

6.3.1 Ego-motion estimation

In terms of ego-motion estimation, the combined method first utilizes tracking information from the previous frame to perform adaptive gating, thereby preliminarily segmenting the radar point cloud into static and moving points. Figure 6.3.2 illustrates

the segmentation result when the purple truck is passing by the ego vehicle. As the figure shows, the method is able to correctly identify the detections generated by the three trucks as dynamic, while recognizing the surrounding static guardrails as stationary. Furthermore, Mahalanobis gating adaptively generates elliptical gating regions based on each truck's estimated size. For the purple truck on the far left, although part of its body lies outside the radar's field of view due to proximity, its estimated size remains consistent. This is because the proposed size estimation employs a sliding average that incorporates historical data. For the yellow truck on the far right, the front-side edge points are missing from detection. This is due to occlusion by the purple and green trucks ahead, resulting in a reduction of the estimated elliptical gating region.

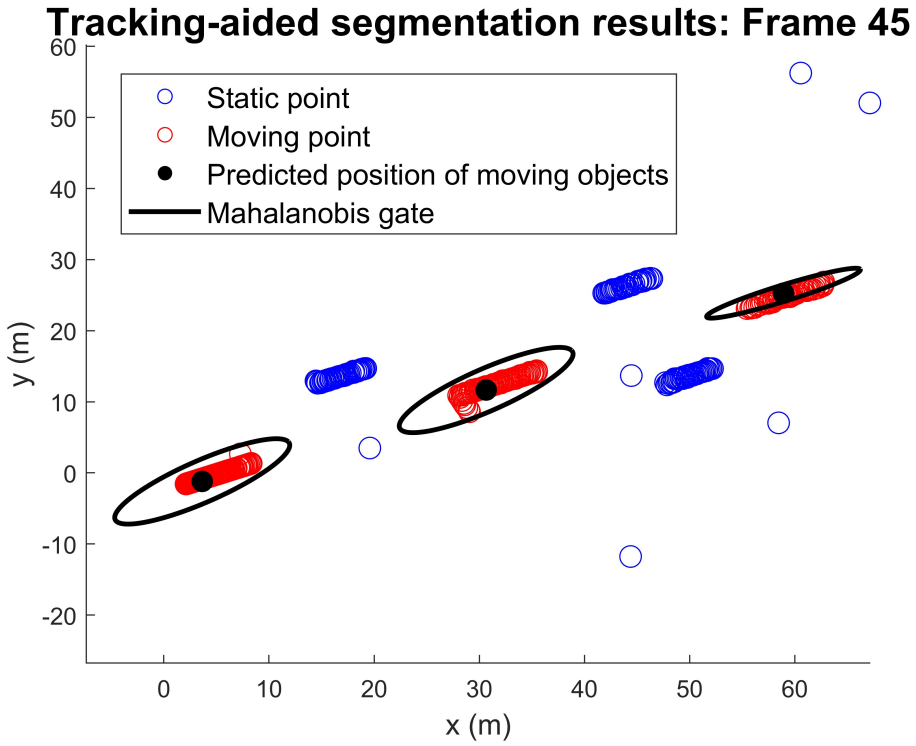


Figure 6.3.2: The tracking-aided segmentation results for an example frame: 2D positions of point cloud

Subsequently, Figure 6.3.3 presents the curve fitting results of the combined method compared with the RANSAC-based method. It can be observed that after removing most of the dynamic points using tracking-aided segmentation, RANSAC is able to correctly select the remaining static points and fit a sinusoidal curve that closely matches the ground truth.

Table 6.3.1 compares the APE and RTE metrics of the RANSAC-based method and the proposed combined method over all frames in the scenario of simulated scene 2. Due to the repeated interactions between the ego vehicle and the three oncoming trucks, the performance of the RANSAC-based method degrades, resulting in increased APE and RTE compared to scene 1. In contrast, the combined method maintains very low APE and RTE values by leveraging historical tracking information, significantly

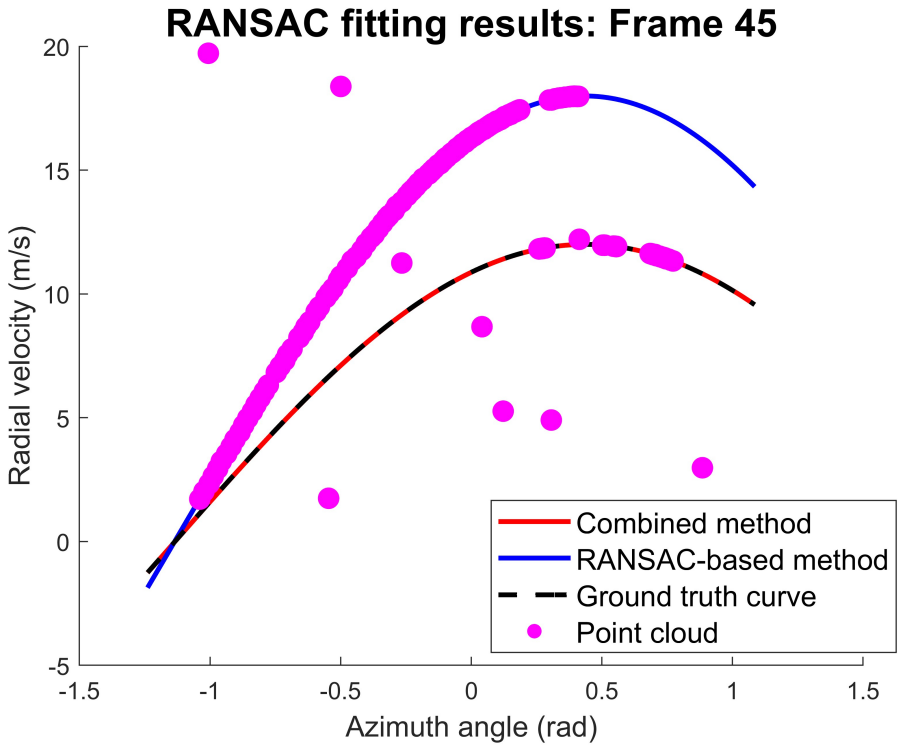


Figure 6.3.3: The RANSAC curve fitting results for an example frame (in the Doppler-azimuth plane)

improving the accuracy and robustness of ego-motion estimation.

Table 6.3.1: Ego-motion estimation performance comparison between RANSAC-based method and combined method for simulated scene 2

Method	**APE (m/s)**	**RTE (cm)**
RANSAC-based method	3.16	270.90
Combined method	0.03	1.29
Absolute difference	-3.13	-269.61

Next, the frame-wise APE and RTE are plotted in Figure 6.3.4 using logarithmic scale for better visualization. Due to the sequential encounters with three oncoming trucks, the RANSAC-based method produces three distinct peaks in APE, each corresponding to a large estimation error, with peak values approaching 10^2 m/s. The width of these peaks gradually decreases, because during the former truck encounters, subsequent trucks still far in space also contribute dynamic points, increasing the number of frames dominated by moving objects. In contrast, during the later encounters, the previous trucks have already passed, resulting in less interference. When using the proposed combined method, the ego-motion errors induced by all three trucks are effectively suppressed, maintaining the velocity estimation error below 0.1 m/s across all frames. This further demonstrates the improved effectiveness of the combined method in highly dynamic driving scenarios.

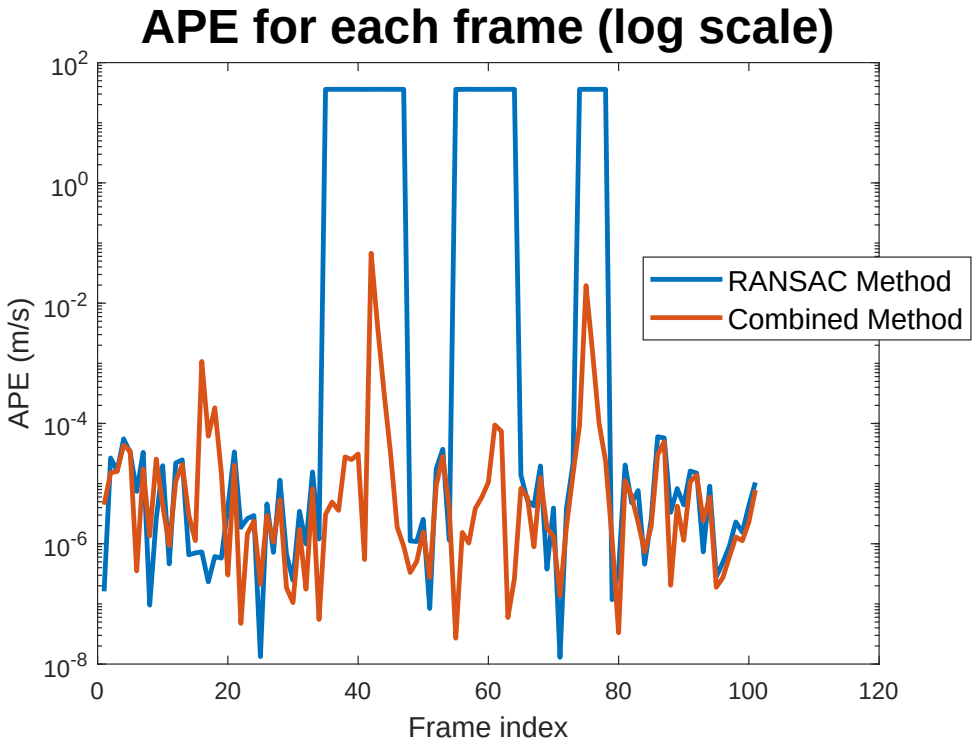


Figure 6.3.4: APE per frame for simulated Scene 2: RANSAC-based method and proposed combined method (visualized in logarithmic scale)

6.3.2 Multiple extended object tracking

In terms of tracking performance, Table 6.3.2 presents the GOSPA score and the RMSE values of the three size estimation parameters. The trajectory tracking results for the three trucks are illustrated in Figure 6.3.5. It can be observed that the trajectories along the x-axis are accurately estimated. However, in the y-axis, the estimated positions tend to be initially biased toward lower values, with the estimation errors gradually reducing over time. This phenomenon is more pronounced for the rear trucks compared to the front ones, which is attributed to the occlusion effect. Due to the large physical size of the trucks and their relatively close spacing, the front truck can block the radar's view of the front surface of the rear trucks. As a result, only the left side surface of the rear trucks is visible to the radar in the early frames, leading to underestimated y positions. As time progresses and the front trucks move out of the radar's field of view, the occlusion is reduced, and the tracking accuracy of the rear trucks improves accordingly.

This occlusion-induced bias contributes to a relatively larger GOSPA score of 3.46 compared to scene 1 and also affects the accuracy of size estimation. The per-object RMSE analysis shows that the estimation errors in all three parameters (semi-major

axis, semi-minor axis, and orientation angle) are higher for the rear trucks than for the front ones. This is because partial occlusion in the early stage limits the accuracy of size estimation for the rear targets.

Overall, for Scene 2, the proposed combined method achieves robust performance in both trajectory tracking and shape estimation.

Table 6.3.2: Performance metrics of multiple extended object tracking for simulated scene 2

	Mean GOSPA	**RMSE$_a$ (m)**	**RMSE$_b$ (m)**	**RMSE$_\theta$ (°)**
All objects	3.46	1.31	0.76	6.30
Object 1 (Purple truck)		0.45	0.20	1.63
Object 2 (Green truck)		0.70	0.70	6.03
Object 3 (Yellow truck)		2.06	1.05	8.60

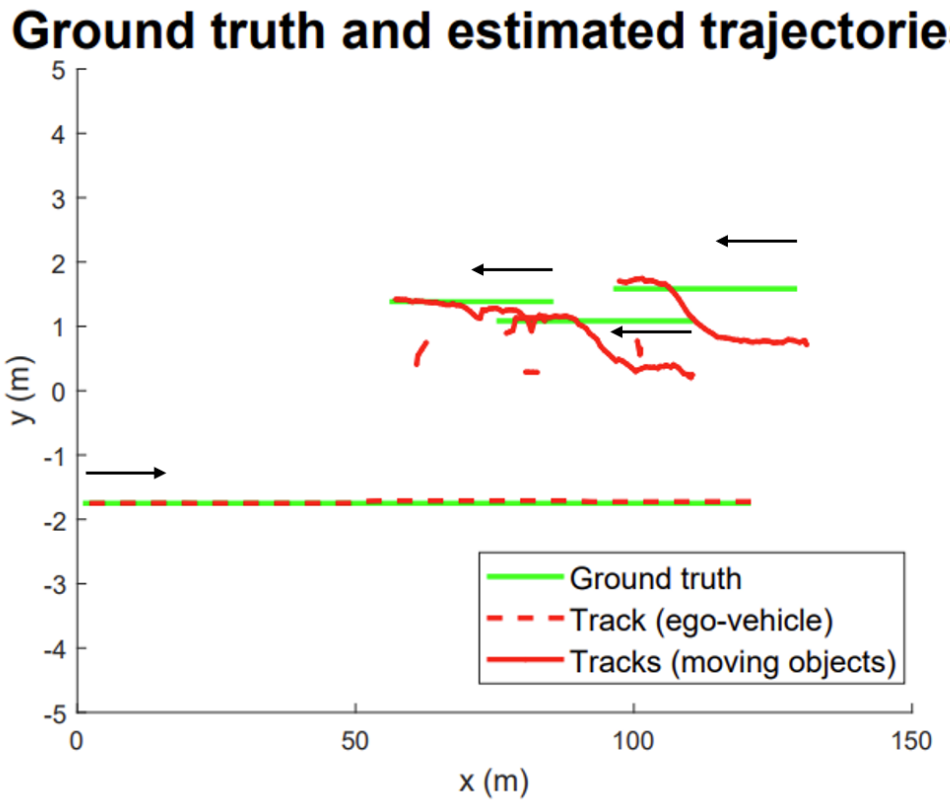


Figure 6.3.5: Tracking results for simulated Scene 2: ground truth and estimated trajectories

6.4 Evaluation on More Scenes

To further evaluate the adaptability of the proposed method in more diverse scenarios, in this section the configurations of Scene 1 and Scene 2 are extended to generate

a total of 20 simulated scenes, each lasting 10 seconds. The following variations were applied across the different scenes:

- Varying the ego-velocity within the range of 8 to 13 m/s.
- Changing the position of the ego-vehicle from the fast lane to the slow lane in a 2-lanes road as those considered so far.
- Adjusting the positions of moving objects by moving them from the fast lane to the slow lane or vice versa.

Table 6.4.1 and 6.4.2 summarize the average performance of ego-motion estimation and multiple extended object tracking across these 20 scenarios. The results indicate that the proposed method achieves consistently accurate and robust performance, and exhibits strong generalization capability under various scene configurations.

Table 6.4.1: Ego-motion estimation performance: averaged among 20 diverse scenes

Method	**APE (m/s)**	**RTE (cm)**
RANSAC-based method	2.02	158.14
Combined method	0.02	0.53
Absolute difference	-2.00	-157.61

Table 6.4.2: Multiple extended object tracking performance: averaged among 20 diverse scenes

	Mean GOSPA	**RMSE$_\mathbf{a}$ (m)**	**RMSE$_\mathbf{b}$ (m)**	**RMSE$_\theta$ (°)**
All objects	3.70	1.04	0.55	8.40

This chapter validates the performance of ego-motion estimation and multiple extended object tracking using simulated point cloud generated by a dedicated MATLAB tool. Various simulation results demonstrate that the proposed combined method can significantly improve the ego-motion estimation robustness in the presence of large oncoming vehicles, while also yielding good tracking results. After addressing the problem with well-controlled simulation data, the next chapter moves on to test the performance on a real-world dataset.

7 Real Dataset Results

In addition to the simulation-based evaluations presented in Chapter 6, this chapter validates the proposed combined method for ego-motion estimation & multiple extended object tracking using real-world radar data. The RadarScenes dataset [77], a publicly available benchmark containing annotated automotive radar measurements, is utilized for this purpose. Since the ground-truth trajectories and sizes of moving objects are not directly accessible from this dataset, the evaluation focuses primarily on the performance of ego-motion estimation. Both quantitative comparisons and qualitative visualizations are provided to demonstrate the effectiveness of the proposed approach under real driving conditions.

In this chapter, Section 7.1 first introduces the RadarScenes dataset and describes the selected scenarios for evaluation. In Section 7.2, the ego-motion estimation results of the proposed method are presented and compared with the conventional RANSAC-based approach.

7.1 RadarScenes Dataset

The RadarScenes dataset [77] is a publicly available large-scale dataset designed to support research on automotive radar perception tasks. It consists of over 10,000 labeled frames captured across various driving environments, including urban streets, highways, and suburban roads. The dataset was collected using a production vehicle equipped with four automotive-grade Continental radars, and it includes rich metadata such as ego-vehicle odometry, semantic annotations, and sensor calibration parameters. RadarScenes emphasizes real-world challenges such as multipath reflections, clutter, occlusion, and sparse detections, making it well-suited for evaluating the robustness and generalizability of radar-based perception and ego-motion estimation algorithms.

As illustrated in Figure 7.1.1, the RadarScenes dataset provides measurements from four 77 GHz automotive radars mounted on the test vehicle. In this work, data from corner Radar 3 is utilized to evaluate the proposed method, as it can observe both the front region (typically occupied by moving objects) and the side region (where static objects often appear). This radar is mounted at the front-right corner of the vehicle and oriented with a 25° angular offset from the forward driving direction. Radar 3 operates at 77 GHz with a maximum range of 100 m, offering a range resolution of 0.15 m and Doppler resolution of 0.3 m/s. Its azimuth field of view is ±60° with 0.5° resolution, and it records frames at 17 Hz. These specifications enable accurate and temporally dense sensing of surrounding dynamic objects. The detailed parameters are listed in Table 6.1.1.

To comprehensively evaluate the robustness of the proposed combined method, two categories of real-world scenarios are selected from the RadarScenes dataset. The first

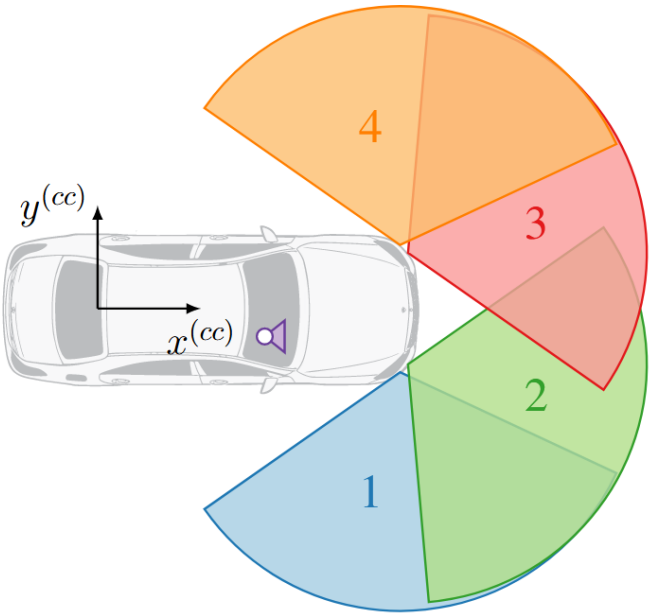


Figure 7.1.1: Radar configuration and field of view on the test vehicle [77]

Table 7.1.1: Parameters of corner Radar 3 in *RadarScenes* dataset

Parameters	**Value**
Center frequency (GHz)	77
Maximum detectable range (m)	100
Range resolution (m)	0.15
Doppler resolution (m/s)	0.36
Field of view for azimuth angle (°)	±60
Azimuth angle resolution (°)	0.5
Angle with respect to the forward direction (°)	25
Frame rate (Hz)	17

category contains dynamic scenes that reflect the challenges described in Section 5.1, where one or more large vehicles approach the ego vehicle from the opposite lane. A total of seven representative scenarios are selected, namely: Scene 1, 67, 76, 104, 125, 127, and 146. These scenes cover a variety of road types including urban and suburban roads, as well as diverse weather conditions including clear, cloudy, and rainy environments. The second category consists of five static scenes, where no large vehicles approach the ego vehicle in the opposite direction. These scenarios are used to verify that the combined method can maintain consistent and accurate performance also in ordinary environments, where conventional RANSAC-based approaches typically operate reliably.

7.2 Ego-motion Estimation

Since obtaining the ground truth trajectories and dimensions of surrounding moving objects from RadarScenes is complex due to the lack of provided labelled information in the dataset, this chapter focuses on evaluating the performance of ego-motion estimation. To provide a quantitative assessment, APE as illustrated in Section 3.7.1 is adopted as the evaluation metric. In order to highlight the performance improvement during the short interval when large vehicles pass the ego vehicle, a 10-second segment is extracted for additional evaluation.

For the selected seven dynamic scenes, Table 7.2.1 reports the APE over the full scene, while Table 7.2.2 shows the APE 'zooming-in' during the 10-second short intervals. A comparison against the baseline RANSAC-based method is conducted. To account for the stochastic nature of RANSAC, each experiment is repeated 100 times for each method and scene.

Across all seven dynamic scenes, the proposed combined method consistently outperforms the conventional RANSAC-based method in terms of ego-motion estimation accuracy. As shown in Table 7.2.1, the APE over the full duration of each scene is significantly reduced by the proposed method. On average, a performance improvement of **25.9%** is observed. Furthermore, when focusing on the 10-second short window centered around the large vehicle encounter event, the advantage of the combined method becomes more pronounced. As reported in Table 7.2.2, the average APE reduction reaches **56.9%**, demonstrating the method's robustness under challenging dynamic conditions. These results confirm that the integration of tracking-aided segmentation and adaptive gating effectively mitigates the failure cases commonly encountered by RANSAC.

Table 7.2.1: APE (m/s) over the full scene: comparison between RANSAC-based method and combined method

Method	**Scene 1**	**Scene 67**	**Scene 76**	**Scene 104**	**Scene 125**	**Scene 127**	**Scene 146**	**Mean**
RANSAC-based method	0.55	0.60	0.55	0.39	0.80	0.32	0.54	**0.54**
Combined method	0.50	0.57	0.42	0.34	0.28	0.19	0.51	**0.40**
Absolute difference	-0.05	-0.03	-0.13	-0.05	-0.52	-0.13	-0.03	**-0.14**

The improvement in ego-motion estimation accuracy can be primarily attributed to the effective removal of moving points at the current frame, guided by historical tracking information. Recall from Section 5.1 that in the scenario shown in Figure 5.1.2, the RANSAC-based method failed to fit an accurate sinusoidal curve due to heavy interference from moving objects. For the same scenario, Figure 7.2.1 illustrates the curve fitting result of the combined method after tracking-aided segmentation has been applied to filter out dynamic points. It can be observed that the static points retained after segmentation are well aligned with the ideal sinusoidal pattern. As a result, the RANSAC algorithm is able to robustly fit a curve that closely matches the expected theoretical relationship between velocity and azimuth angle for static scatterers. This significantly reduces the estimation error of ego-motion.

Table 7.2.2: APE (m/s) during the 10-second encounter window with large vehicles: comparison between RANSAC-based method and combined method

Method	**Scene 1**	**Scene 67**	**Scene 76**	**Scene 104**	**Scene 125**	**Scene 127**	**Scene 146**	**Mean**
RANSAC-based method	1.72	1.03	1.20	0.90	1.33	1.25	0.67	**1.16**
Combined method	1.56	0.51	0.46	0.25	0.17	0.26	0.32	**0.50**
Absolute difference	-0.16	-0.52	-0.74	-0.65	-1.16	-0.99	-0.35	**-0.66**

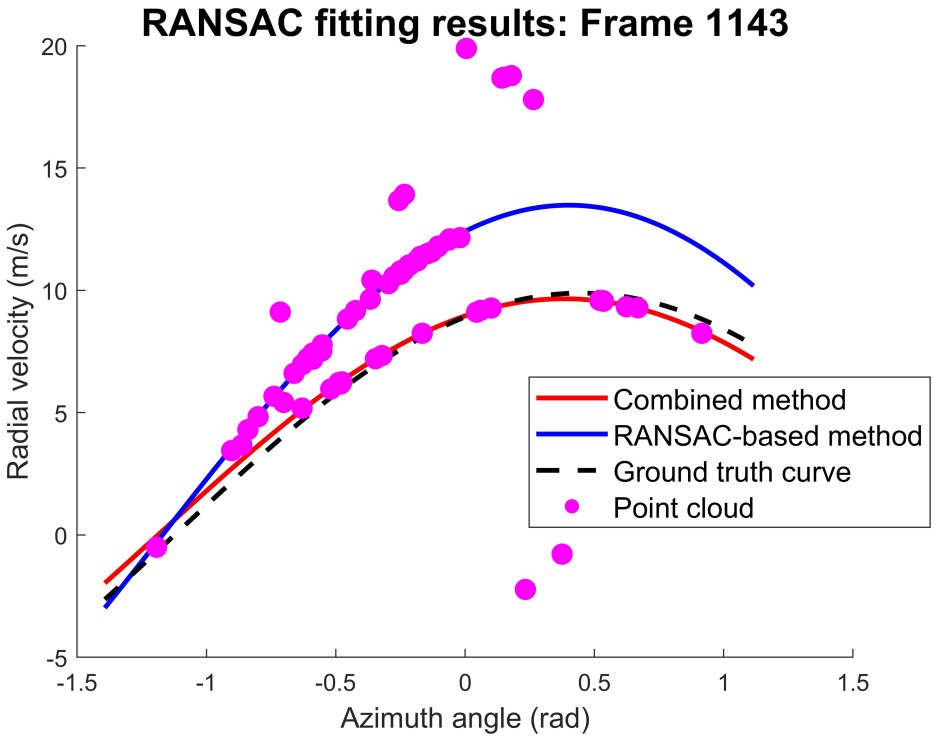


Figure 7.2.1: RANSAC fitting results after tracking-aided segmentation for the same frame analyzed in Figure 5.1.2

To further evaluate the generalizability of the proposed method, Table 7.2.3 reports the ego-motion estimation results for five selected static scenes where no large oncoming vehicles are present. In these scenes, ego-motion estimation mainly relies on static roadside targets, and the interference from moving objects is minimal.

The results show that in static scenes, the combined method yields comparable performance to the baseline RANSAC-based method, with a slight increase in average APE. This observation suggests that while the combined method is specifically designed to mitigate the influence of large moving objects, other factors can also affect ego-motion estimation accuracy in real-world driving scenarios. For instance, in cases of strong ego acceleration or deceleration, or during turning maneuvers with significant rotational

Table 7.2.3: APE (m/s) for static scenes: comparison between RANSAC-based method and combined method

Method	**Scene 77**	**Scene 87**	**Scene 92**	**Scene 101**	**Scene 130**	**Mean**
RANSAC-based method	0.30	0.34	0.13	0.25	0.28	**0.26**
Combined method	0.31	0.33	0.16	0.26	0.29	**0.27**
Absolute difference	+0.01	-0.01	+0.03	+0.01	+0.01	**+0.01**

velocity, the underlying motion model may deviate from the assumptions used in the algorithm [8]. As the combined method is a model-based approach, such deviations can reduce its effectiveness. In contrast, the RANSAC method may occasionally offer better results since it does not link the ego-motion from consecutive time steps. To address these limitations, future work may explore more expressive motion models that capture non-linear vehicle dynamics, or leverage data-driven approaches such as deep learning models that can learn complex motion patterns from large-scale radar datasets.

In summary, the proposed combined method has been evaluated on multiple real-world driving scenarios containing significant incoming large vehicles, using data from the public *RadarScenes* dataset. These scenarios include highway environments, where large trucks often appear one after another in platoon formations, and urban settings, where combinations of buses, trucks, and vans are frequently encountered. Experimental results demonstrate clear improvements in ego-motion estimation accuracy when compared to the baseline RANSAC-based approach. This highlights the method's practical value in handling complex dynamic environments in real-life driving situations.

Part IV
Closing Remarks

8 Conclusion and Future Work

This chapter summarizes the main conclusions of the thesis and outlines possible future research directions. Section 8.1 presents the conclusions and key findings, while Section 8.2 discusses limitations and potential improvements in the future research.

8.1 Conclusion

This thesis explores a novel integration of ego-motion estimation, a localization task, and multiple extended object tracking, a perception task, within a unified framework, using only automotive radar as the sensor input. Looking at the open literature in the field, establishing such connection is relatively innovative, especially when relying solely on a single automotive radar. Two combined estimation methods were proposed and validated throughout this thesis, differing primarily in the form of radar input data.

The first method, developed in the early stage of the thesis, operates directly on raw radar signals, aiming to leverage low-level information and validate the theoretical feasibility of the framework. The second method, developed in the later stage, is based on radar point cloud data, a more common data format in publicly available datasets, and focuses on addressing practical challenges in dynamic driving scenarios.

In Part II, the radar raw signals-based method demonstrated the feasibility of performing combined ego-motion estimation and object tracking directly on low-level radar data. Simulation results under various ego-velocity showed that the method can effectively distinguish between static and moving targets, enabling accurate motion state estimation of both the ego-vehicle and surrounding moving objects. However, the method did not estimate object size or orientation, and was not practical yet for deployment in real-world scenarios.

In Part III, to bridge the gap toward practical applications, a second method based on radar point cloud data was proposed. This framework was designed to address key challenges in dynamic driving environments, particularly the presence of oncoming large vehicles, which is common in real-life scenarios like highways with platoons of trucks or urban roads with buses, trucks, and vans. In simulations generated with a dedicated MATLAB tool, the proposed method outperformed the state-of-the-art RANSAC-based baseline in terms of ego-motion estimation [15], reducing the APE metric by 2.00 m/s and the RTE metric by 1.58 m. In terms of multiple extended object tracking, it also achieved a GOSPA score of 3.70 and provided accurate size estimates of tracked objects, with RMSEs of 1.04 m (major axis), 0.55 m (minor axis), and 8.40° (orientation).

Validation on the real-world *RadarScenes* dataset demonstrated further improvements in real traffic scenarios. On average across seven dynamic driving scenes, the proposed method reduced APE by 25.9% over the entire scene and by 56.9% in 10-

second critical segments compared to the RANSAC-based baseline [15]. By leveraging prior tracking information to filter out moving object detections, the method enhanced static point selection and improved the robustness of ego-motion estimation.

In summary, both methods demonstrate the effectiveness of radar-only systems for achieving accurate and robust localization and perception in complex, dynamic traffic scenarios.

8.2 Future work

Despite the effectiveness of the proposed methods, several limitations remain and suggest promising directions for future research:

1. **Extension to 3D space using 4D imaging radar:**
   In part III, due to time constraints and the limitation of the RadarScenes dataset whose radar sensor lacks elevation angle measurements, the proposed method was only evaluated on the 2D plane. Future work could extend the current pipeline to 3D space, incorporating the position and velocity estimation in the z (elevation) dimension. Moreover, advanced automotive radar systems such as 4D imaging radars, which offer higher angular resolution and denser point clouds, are becoming increasingly available in industry compared to the radars used to record RadarScenes [85, 86]. Investigating and evaluating the proposed method on such datasets with elevation angle information and improved detection quality will be critical for validating its performance in realistic driving scenarios.

2. **Adoption of more advanced tracking algorithms:**
   In both the raw signal-based and point cloud-based methods, the tracking module relies on the computationally efficient but relatively simple data association algorithm GNN. As discussed in Section 6.2.2, GNN performs adequately in moderately dynamic scenes but may struggle in more complex situations, such as when multiple targets are spatially close or their trajectories intersect. To overcome this limitation, future studies could investigate more sophisticated multi-target tracking frameworks, particularly those based on RFS theory. Promising candidates include the PHD filter [55], the PMBM filter [56], and the sum-product algorithm [39]. These approaches are expected to provide improved robustness and accuracy under challenging tracking situations.

3. **Development of more realistic simulators and comprehensive radar datasets for raw signals:**
   For radar algorithm research, realistic simulators and comprehensive datasets are crucial. Currently, most radar datasets only provide point cloud data [1, 14]. The algorithm and data ecosystem based on raw radar signals is still in its early stage. An important future task is to design simulators for raw radar signals that more accurately reflect real-world scenarios. Equally important is overcoming storage limitations and collecting more comprehensive datasets that include raw radar signals, which can be widely adopted by the academic community.

4. **Design of joint ego-motion estimation and tracking algorithms:**
   Within each frame, the combined method proposed in this thesis solves the two tasks sequentially, in which ego-motion is estimated first, followed by tracking. While this decoupled design simplifies the implementation, it may limit the exploitation of mutual information between ego-state and dynamic objects. As suggested in [12, 13], jointly estimating ego-motion and object states in a tightly coupled framework can improve the overall estimation accuracy and robustness, especially in highly dynamic environments. Challenges remain in capturing the motion dynamics of various road participants and effectively utilizing sparse and noisy radar point clouds. Future research could focus on developing frameworks to simultaneously infer ego-motion and surrounding object states in a coherent manner.

Bibliography

[1] L. Fan, J. Wang, Y. Chang, Y. Li, Y. Wang, and D. Cao, "4d mmwave radar for autonomous driving perception: a comprehensive survey," *IEEE Transactions on Intelligent Vehicles*, 2024.

[2] Z. Han, J. Wang, Z. Xu, S. Yang, L. He, S. Xu, J. Wang, and K. Li, "4d millimeter-wave radar in autonomous driving: A survey," *arXiv preprint arXiv:2306.04242*, 2023.

[3] N. J. Abu-Alrub and N. A. Rawashdeh, "Radar odometry for autonomous ground vehicles: A survey of methods and datasets," *IEEE Transactions on Intelligent Vehicles*, vol. 9, no. 3, pp. 4275–4291, 2023.

[4] F. Engels, P. Heidenreich, M. Wintermantel, L. Stäcker, M. Al Kadi, and A. M. Zoubir, "Automotive radar signal processing: Research directions and practical challenges," *IEEE Journal of Selected Topics in Signal Processing*, vol. 15, no. 4, pp. 865–878, 2021.

[5] S. Zhu, A. Yarovoy, and F. Fioranelli, "Deepego: Deep instantaneous ego-motion estimation using automotive radar," *IEEE Transactions on Radar Systems*, vol. 1, pp. 166–180, 2023.

[6] S. Yuan, S. Zhu, F. Fioranelli, and A. G. Yarovoy, "3-d ego-motion estimation using multi-channel fmcw radar," *IEEE Transactions on Radar Systems*, vol. 1, pp. 368–381, 2023.

[7] H. Chen, Y. Liu, and Y. Cheng, "Drio: Robust radar-inertial odometry in dynamic environments," *IEEE Robotics and Automation Letters*, vol. 8, no. 9, pp. 5918–5925, 2023.

[8] S. Zhu, S. Ravindran, L. Chen, A. Yarovoy, and F. Fioranelli, "Deepego+: Unsynchronized radar sensor fusion for robust vehicle ego-motion estimation," *IEEE Transactions on Radar Systems*, 2025.

[9] D. C. Herraez, M. Zeller, L. Chang, I. Vizzo, M. Heidingsfeld, and C. Stachniss, "Radar-only odometry and mapping for autonomous vehicles," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2024, pp. 10 275–10 282.

[10] K. Granström and M. Baum, "A tutorial on multiple extended object tracking," *Authorea Preprints*, 2022.

[11] C.-C. Wang, C. Thorpe, S. Thrun, M. Hebert, and H. Durrant-Whyte, "Simultaneous localization, mapping and moving object tracking," *The International Journal of Robotics Research*, vol. 26, no. 9, pp. 889–916, 2007.

[12] J. Zhang, M. Henein, R. Mahony, and V. Ila, "Vdo-slam: A visual dynamic object-aware slam system," *arXiv preprint arXiv:2005.11052*, 2020.

[13] X. Tian, Z. Zhu, J. Zhao, G. Tian, and C. Ye, "Dl-slot: Tightly-coupled dynamic lidar slam and 3d object tracking based on collaborative graph optimization," *IEEE Transactions on Intelligent Vehicles*, vol. 9, no. 1, pp. 1017–1027, 2023.

[14] S. Yao, R. Guan, Z. Peng, C. Xu, Y. Shi, W. Ding, E. G. Lim, Y. Yue, H. Seo, K. L. Man *et al.*, "Exploring radar data representations in autonomous driving: A comprehensive review," *IEEE Transactions on Intelligent Transportation Systems*, 2025.

[15] D. Kellner, M. Barjenbruch, J. Klappstein, J. Dickmann, and K. Dietmayer, "Instantaneous ego-motion estimation using doppler radar," in *16th International IEEE Conference on Intelligent Transportation Systems (ITSC 2013)*. IEEE, 2013, pp. 869–874.

[16] M. Barjenbruch, D. Kellner, J. Klappstein, J. Dickmann, and K. Dietmayer, "Joint spatial-and doppler-based ego-motion estimation for automotive radars," in *2015 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2015, pp. 839–844.

[17] B. Jian and B. C. Vemuri, "A robust algorithm for point set registration using mixture of gaussians," in *Tenth IEEE International Conference on Computer Vision (ICCV'05) Volume 1*, vol. 2. IEEE, 2005, pp. 1246–1251.

[18] M. Rapp, M. Barjenbruch, M. Hahn, J. Dickmann, and K. Dietmayer, "Probabilistic ego-motion estimation using multiple automotive radar sensors," *Robotics and Autonomous Systems*, vol. 89, pp. 136–146, 2017.

[19] P. Biber and W. Straßer, "The normal distributions transform: A new approach to laser scan matching," in *Proceedings 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2003)(Cat. No. 03CH37453)*, vol. 3. IEEE, 2003, pp. 2743–2748.

[20] K. Haggag, S. Lange, T. Pfeifer, and P. Protzel, "A credible and robust approach to ego-motion estimation using an automotive radar," *IEEE Robotics and Automation Letters*, vol. 7, no. 3, pp. 6020–6027, 2022.

[21] C. X. Lu, M. R. U. Saputra, P. Zhao, Y. Almalioglu, P. P. De Gusmao, C. Chen, K. Sun, N. Trigoni, and A. Markham, "milliego: single-chip mmwave radar aided egomotion estimation via deep sensor fusion," in *Proceedings of the 18th Conference on Embedded Networked Sensor Systems*, 2020, pp. 109–122.

[22] C. Doer and G. F. Trommer, "Yaw aided radar inertial odometry using manhattan world assumptions," in *2021 28th Saint Petersburg International Conference on Integrated Navigation Systems (ICINS)*. IEEE, 2021, pp. 1–9.

[23] Y. Zhuang, B. Wang, J. Huai, and M. Li, "4d iriom: 4d imaging radar inertial odometry and mapping," *IEEE Robotics and Automation Letters*, vol. 8, no. 6, pp. 3246–3253, 2023.

[24] H. Griffiths, L. Cohen, S. Watts, E. Mokole, C. Baker, M. Wicks, and S. Blunt, "Radar spectrum engineering and management: Technical and regulatory issues," *Proceedings of the IEEE*, vol. 103, no. 1, pp. 85–102, 2014.

[25] Y. Zhou, L. Liu, H. Zhao, M. López-Benítez, L. Yu, and Y. Yue, "Towards deep radar perception for autonomous driving: Datasets, methods, and challenges," *Sensors*, vol. 22, no. 11, p. 4208, 2022.

[26] M. A. Fischler and R. C. Bolles, "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography," *Communications of the ACM*, vol. 24, no. 6, pp. 381–395, 1981.

[27] D. Kellner, M. Barjenbruch, J. Klappstein, J. Dickmann, and K. Dietmayer, "Instantaneous ego-motion estimation using multiple doppler radars," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2014, pp. 1592–1597.

[28] Y. S. Park, Y.-S. Shin, J. Kim, and A. Kim, "3d ego-motion estimation using low-cost mmwave radars via radar velocity factor for pose-graph slam," *IEEE Robotics and Automation Letters*, vol. 6, no. 4, pp. 7691–7698, 2021.

[29] S. Lovett, K. MacWilliams, S. Rajan, and C. Rossa, "Indoor temporally constrained instantaneous ego-motion estimation using 4d doppler radar," *IEEE Transactions on Instrumentation and Measurement*, 2025.

[30] A. Kramer, C. Stahoviak, A. Santamaria-Navarro, A.-A. Agha-Mohammadi, and C. Heckman, "Radar-inertial ego-velocity estimation for visually degraded environments," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 5739–5746.

[31] J.-T. Huang, R. Xu, A. Hinduja, and M. Kaess, "Multi-radar inertial odometry for 3d state estimation using mmwave imaging radar," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2024, pp. 12 006–12 012.

[32] H. Yang, P. Antonante, V. Tzoumas, and L. Carlone, "Graduated non-convexity for robust spatial perception: From non-minimal solvers to global outlier rejection," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 1127–1134, 2020.

[33] S. Kim, J. Seok, J. Lee, and K. Jo, "Radar4motion: Imu-free 4d radar odometry with robust dynamic filtering and rcs-weighted matching," *IEEE Transactions on Intelligent Vehicles*, 2024.

[34] C. Doer and G. F. Trommer, "An ekf based approach to radar inertial odometry," in *2020 IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI)*. IEEE, 2020, pp. 152–159.

[35] A. Galeote-Luque, V. Kubelka, M. Magnusson, J.-R. Ruiz-Sarmiento, and J. Gonzalez-Jimenez, "Doppler-only single-scan 3d vehicle odometry," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2024, pp. 13 703–13 709.

[36] S. Yuan, D. Wang, F. Fioranelli, and A. Yarovoy, "Improved accuracy for 3d ego-motion estimation using automotive fmcw mimo radar," in *2024 IEEE Radar Conference (RadarConf24)*. IEEE, 2024, pp. 1–6.

[37] J. W. Tukey *et al.*, *Exploratory data analysis*. Springer, 1977, vol. 2.

[38] K. Granstrom, M. Baum, and S. Reuter, "Extended object tracking: Introduction, overview and applications," *arXiv preprint arXiv:1604.00970*, 2016.

[39] F. Meyer and J. L. Williams, "Scalable detection and tracking of geometric extended objects," *IEEE Transactions on Signal Processing*, vol. 69, pp. 6283–6298, 2021.

[40] S. Batool, F. Frezza, F. Mangini, and P. Simeoni, "Introduction to radar scattering application in remote sensing and diagnostics," *Atmosphere*, vol. 11, no. 5, p. 517, 2020.

[41] M. Ester, H.-P. Kriegel, J. Sander, X. Xu *et al.*, "A density-based algorithm for discovering clusters in large spatial databases with noise," in *kdd*, vol. 96, no. 34, 1996, pp. 226–231.

[42] S. Lim, S. Lee, and S.-C. Kim, "Clustering of detected targets using dbscan in automotive radar systems," in *2018 19th international radar symposium (IRS)*. IEEE, 2018, pp. 1–7.

[43] J. W. Koch, "Bayesian approach to extended object and cluster tracking using random matrices," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 44, no. 3, pp. 1042–1059, 2008.

[44] M. Feldmann, D. Franken, and W. Koch, "Tracking of extended objects and group targets using random matrices," *IEEE Transactions on Signal Processing*, vol. 59, no. 4, pp. 1409–1420, 2011.

[45] K. Granstrom, "A random matrix approach to radar-based extended object tracking," *Sensors*, vol. 19, no. 16, p. 3584, 2019.

[46] X. Weng, J. Wang, D. Held, and K. Kitani, "3d multi-object tracking: A baseline and new evaluation metrics," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2020, pp. 10 359–10 366.

[47] M. Hassan, F. Fioranelli, A. Yarovoy, and S. Ravindran, "Radar multi object tracking using dnn features," in *2023 IEEE International Radar Conference (RADAR)*. IEEE, 2023, pp. 1–6.

[48] A. Palffy, E. Pool, S. Baratam, J. F. Kooij, and D. M. Gavrila, "Multi-class road user detection with 3+ 1d radar in the view-of-delft dataset," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 4961–4968, 2022.

[49] J. e. a. Scheel, "Radar-based mot: A hybrid radar-camera tracking pipeline," in *2022 IEEE Intelligent Vehicles Symposium (IV)*, 2022, pp. 1082–1089.

[50] H. W. Kuhn, "The hungarian method for the assignment problem," *Naval research logistics quarterly*, vol. 2, no. 1-2, pp. 83–97, 1955.

[51] K. G. Murty, "An algorithm for ranking all the assignments in order of increasing cost," *Operations research*, vol. 16, no. 3, pp. 682–687, 1968.

[52] P. D. Konstantinova, A. Udvarev, and T. Semerdjiev, "A study of a target tracking algorithm using global nearest neighbor approach." in *Compsystech*, vol. 3, 2003, pp. 290–295.

[53] Y. Bar-Shalom, F. Daum, and J. Huang, "The probabilistic data association filter," *IEEE Control Systems Magazine*, vol. 29, no. 6, pp. 82–100, 2009.

[54] S. S. Blackman, "Multiple hypothesis tracking for multiple target tracking," *IEEE Aerospace and Electronic Systems Magazine*, vol. 19, no. 1, pp. 5–18, 2004.

[55] K. Granstrom, C. Lundquist, and O. Orguner, "Extended target tracking using a gaussian-mixture phd filter," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 48, no. 4, pp. 3268–3286, 2012.

[56] K. Granström, M. Fatemi, and L. Svensson, "Poisson multi-bernoulli mixture conjugate prior for multiple extended target filtering," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 56, no. 1, pp. 208–225, 2019.

[57] S. Fang and H. Li, "Lidar slammot based on confidence-guided data association," *arXiv preprint arXiv:2412.01041*, 2024.

[58] B. Bescos, C. Campos, J. D. Tardós, and J. Neira, "Dynaslam ii: Tightly-coupled multi-object tracking and slam," *IEEE robotics and automation letters*, vol. 6, no. 3, pp. 5191–5198, 2021.

[59] Z. Ying and H. Li, "Imm-slammot: Tightly-coupled slam and imm-based multi-object tracking," *IEEE Transactions on Intelligent Vehicles*, vol. 9, no. 2, pp. 3964–3974, 2023.

[60] Y.-K. Lin, W.-C. Lin, and C.-C. Wang, "Asynchronous state estimation of simultaneous ego-motion estimation and multiple object tracking for lidar-inertial odometry," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2023, pp. 10 616–10 622.

[61] S. Shi, X. Wang, and H. Li, "Pointrcnn: 3d object proposal generation and detection from point cloud," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 770–779.

[62] J. Liu, W. Xiong, L. Bai, Y. Xia, T. Huang, W. Ouyang, and B. Zhu, "Deep instance segmentation with automotive radar detection points," *IEEE Transactions on Intelligent Vehicles*, vol. 8, no. 1, pp. 84–94, 2022.

[63] M. A. Richards, J. A. Scheer, and W. A. Holm, *Principles of Modern Radar: Basic Principles*. Raleigh, NC, USA: SciTech Publishing, 2010.

[64] C. Iovescu and S. Rao, "The fundamentals of millimeter wave radar sensors," *Texas Instruments*, pp. 1–7, 2020.

[65] C. Audet and J. E. Dennis Jr, "Analysis of generalized pattern searches," *SIAM Journal on optimization*, vol. 13, no. 3, pp. 889–903, 2002.

[66] MathWorks, *Global Optimization Toolbox™ User's Guide*, 2024, accessed: June 2025. [Online]. Available: https://www.mathworks.com/products/global-optimization.html

[67] ——, *Coordinate Systems in Automated Driving Toolbox*, 2024, accessed: June 2025. [Online]. Available: https://www.mathworks.com/help/driving/ug/coordinate-systems.html

[68] International Organization for Standardization, "ISO 8855:2011 - Road vehicles – Vehicle dynamics and road-holding ability – Vocabulary," https://www.iso.org/standard/51180.html, 2011, accessed: 2025-06-30.

[69] J. Smith, F. Particke, M. Hiller, and J. Thielecke, "Systematic analysis of the pmbm, phd, jpda and gnn multi-target tracking filters," in *2019 22th International Conference on Information Fusion (FUSION)*. IEEE, 2019, pp. 1–8.

[70] D. F. Crouse, "The tracker component library: free routines for rapid prototyping," *IEEE Aerospace and Electronic Systems Magazine*, vol. 32, no. 5, pp. 18–27, 2017.

[71] MathWorks, "Sensor fusion and tracking toolbox," Natick, Massachusetts, United States, 2024. [Online]. Available: https://www.mathworks.com/help/fusion/ug/introduction-to-track-logic.html

[72] A. S. Rahmathullah, Á. F. García-Fernández, and L. Svensson, "Generalized optimal sub-pattern assignment metric," in *2017 20th International Conference on Information Fusion (Fusion)*. IEEE, 2017, pp. 1–8.

[73] Á. F. García-Fernández, J. L. Williams, K. Granström, and L. Svensson, "Poisson multi-bernoulli mixture filter: Direct derivation and implementation," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 54, no. 4, pp. 1883–1901, 2018.

[74] L. Lindenmaier, S. Aradi, T. Bécsi, O. Törő, and P. Gáspár, "Gm-phd filter based sensor data fusion for automotive frontal perception system," *IEEE Transactions on Vehicular Technology*, vol. 71, no. 7, pp. 7215–7229, 2022.

[75] Y. Sun, "An fmcw mimo automotive radar signal simulator for realistic extended targets," Master's thesis, Delft University of Technology, Delft, the Netherlands, 2023. [Online]. Available: http://resolver.tudelft.nl/uuid:294cb65b-7ca4-4770-8679-5e84d855962d

[76] S. M. Patole, M. Torlak, D. Wang, and M. Ali, "Automotive radars: A review of signal processing techniques," *IEEE Signal Processing Magazine*, vol. 34, no. 2, pp. 22–35, 2017.

[77] O. Schumann, M. Hahn, N. Scheiner, F. Weishaupt, J. F. Tilly, J. Dickmann, and C. Wöhler, "Radarscenes: A real-world radar point cloud data set for automotive applications," in *2021 IEEE 24th International Conference on Information Fusion (FUSION)*. IEEE, 2021, pp. 1–8.

[78] R. Altendorfer and S. Wirkert, "A complete derivation of the association log-likelihood distance for multi-object tracking," *arXiv preprint arXiv:1508.04124*, 2015.

[79] V. D. Nguyen and T. Claussen, "Individual-gating-by-sorting in mht," in *17th International Conference on Information Fusion (FUSION)*. IEEE, 2014, pp. 1–8.

[80] M. J. Todd and E. A. Yıldırım, "On khachiyan's algorithm for the computation of minimum-volume enclosing ellipsoids," *Discrete Applied Mathematics*, vol. 155, no. 13, pp. 1731–1744, 2007.

[81] MathWorks, *Driving Scenario Designer*, 2024, accessed: June 2025. [Online]. Available: https://www.mathworks.com/help/driving/ref/drivingscenariodesigner-app.html

[82] ——, *Automated Driving Toolbox™ User's Guide*, 2024, accessed: June 2025. [Online]. Available: https://www.mathworks.com/products/automated-driving.html

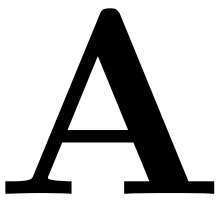
[83] S. H. Juttiga, "Development and evaluation of sensor fusion with multiple radars and cameras," Master's thesis, Technische Hochschule Ingolstadt, Ingolstadt, Germany, 2023. [Online]. Available: https://opus4.kobv.de/opus4-haw/frontdoor/index/index/docId/3794

[84] J. Espinoza Gonzalez and E. González, "Simulation of autonomous driving systems for the city of macas using matlab driving scenario design app," in *World Conference on Information Systems and Technologies*. Springer, 2024, pp. 73–81.

[85] I. Roldan, A. Palffy, J. F. Kooij, D. M. Gavrila, F. Fioranelli, and A. Yarovoy, "A deep automotive radar detector using the radelft dataset," *IEEE Transactions on Radar Systems*, 2024.

[86] D.-H. Paek, S.-H. Kong, and K. T. Wijaya, "K-radar: 4d radar object detection for autonomous driving in various weather conditions," *Advances in Neural Information Processing Systems*, vol. 35, pp. 3819–3829, 2022.

Appendix



This appendix presents the submitted paper:

- S. Yuan, T. Wang, A. Yarovoy and F. Fioranelli, *"Joint ego-motion estimation and multiple object tracking using automotive radar"*, 2025 22nd European Radar Conference (EuRAD), Utrecht, the Netherlands.

This paper has been accepted for oral presentation at the EuRAD 2025 conference in September 2025.

Joint ego-motion estimation and multiple object tracking using automotive radar

Sen Yuan[#1], Taoyue Wang[#2], Alexander Yarovoy[#3], Francesco Fioranelli[#4]

[#]MS3 Group, Department of Microelectronics, Faculty of EEMCS, Delft University of Technology, The Netherlands

{[1]s.yuan-3, [3]a.yarovoy,[4]f.fioranelli} [2]t.wang

***Abstract* — The problem of joint ego-motion estimation and multiple object tracking (MOT) in automotive multiple-input and multiple-output (MIMO) radar has been studied. The 3D ego-motion estimation is performed based on phase changes of the raw signal caused by relative movement between objects and the radar, and the ego-motion-induced velocities are compared with the detected ones to label static vs moving objects. The static objects are used for ego-motion estimation again to improve the accuracy, while the moving objects are used for MOT. The performance of the algorithm has been studied on simulated data and evaluated using different tracking algorithms, proving the feasibility of this approach.**

***Keywords* — Ego-motion estimation, Multiple object tracking, MIMO radar, signal processing.**

I. Introduction

RADAR can provide accurate and direct measurements of the range, relative velocity, and angle of multiple objects, as well as a long-range coverage of over 200 meters even in challenging weather or lighting conditions, outperforming other sensors, namely, camera and Lidar. Thus, radar has attracted significant importance for autonomous driving.

To measure the vehicle's own motion with radar, namely 'ego-motion estimation', state-of-the-art methods can be mainly divided into detection point cloud-based [1], [2] and intermediate frequency signal-based approaches [3], [4]. Performing ego-motion estimation starting from the lower signal level (i.e., the radar base-band signal before range-Doppler processing) can be beneficial in automotive scenarios. Firstly, the ego-motion estimation can be performed fast, within one frame. Secondly, using algorithms implemented directly at the signal level, it is easier to combine them with other high-resolution imaging algorithms.

For self-driving vehicles, the problem of multiple object tracking (MOT) for moving objects in proximity is a critical component to ensure situational awareness and safe planning and control [5]. Various MOT algorithms are used in different tracking applications, the most popular ones being GNN (Global Nearest-Neighbor) [6], JPDA (Joint Probabilistic Data Association) [7], MHT (Multiple Hypotheses Tracking) [8], as well as deep learning methods [9]. GNN, JPDA, and MHT are conventionally used in MOT systems. These methods utilize pruning and merging techniques to manage the growing complexity of tracking multiple objects and to ensure computational efficiency.

However, jointly estimating ego-motion and the movement of objects into a single process can be beneficial for performance improvement and efficiency. Extensive research has been conducted on Camera and LIDAR [10], [11]. However, to our knowledge, not much literature is available on radar data, especially those operating at a raw signal level in the automotive context. One of the challenges is that the estimation of the ego-motion is based on a single frame, while the MOT is done based on the processing of multiple frames. Meanwhile, ideal ego velocity estimation considers static objects, while MOT focuses on only moving objects. This paper addresses this joint problem by tackling the challenges associated with the proposed processing pipeline. The proposed pipeline is verified with numerical simulations and related metrics, showing the feasibility of this approach.

The rest of the paper is organized as follows. In Section II, the fundamental of ego-motion estimation and MOT is discussed. The proposed method and evaluation metrics are given in Section III. The simulation results are provided in Section IV. Finally, conclusions are drawn in Section V.

II. Fundamentals

A. Radar-based ego-motion estimation

The ego-motion estimation is implemented based on the approach in [3], as shown in Fig. 1. The whole process is divided into two steps, initial estimation and iterative estimation. For initial estimation, 2D FFT (Fast Fourier Transform) and 2D CA-CFAR (Cell Averaging Constant False Alarm Rate) detection are performed for the first group of chirps, locating the detected objects. Then, an optimization is performed to find the azimuth and elevation angles of such objects. The same operations are performed for another chirp group, and an initial, coarse value of ego velocity is estimated based on the information of all detected objects, including static and moving ones, as detailed in [4]. However, the moving objects introduce extra Doppler components, degrading the estimation of the ego velocity. Thus, an iterative ego-motion estimation is proposed by comparing the value of object velocity derived from the ego-motion estimation with each detected velocity, thus improving the distinction between static vs moving objects. The static objects are used in the next step of ego-motion estimation until a certain threshold is met and a final value of ego velocity is obtained.

B. Multiple object tracking (MOT)

Given a radar point cloud after detection, the goal of MOT algorithms is to determine the number of moving objects

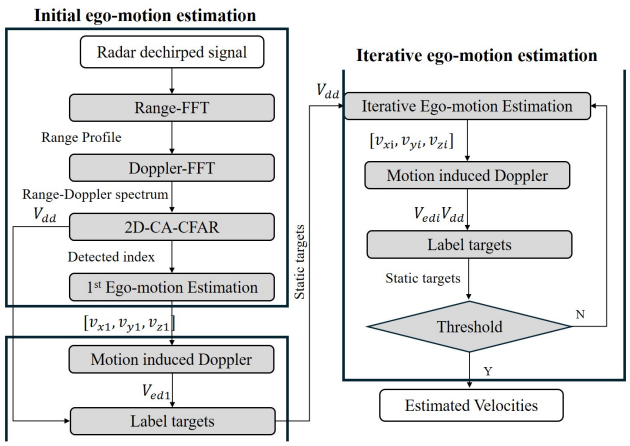


Fig. 1. Block diagram of the ego-motion estimation algorithm inspired by [3].

and their state over time. The state variable of each object includes its 3D position and velocity. The output is a set of tracking trajectories for all moving objects, each associated with a unique object identity. Fig. 2 shows the functional modules within MOT algorithms. First, clutter removal is typically implemented to reduce the contributions caused by static objects, together with a conversion of the detections from the radar coordinate to the world coordinate. Next, clustering is performed to group the detections of the same object. For each time step, gating rejects invalid detections that are too far from the existing object tracks. Only valid detections enter the data association module, which assigns detections to each track. With the prediction step, assigned tracks are updated with the corresponding detections inside the state estimation module. Unassigned tracks are kept, only performing prediction but without updates. Unassigned detections are sent to the track management module for track initialization. Tracks after state estimation are also sent to the track management module for track confirmation and deletion. Finally, the estimated states for each confirmed object track are provided in the output.

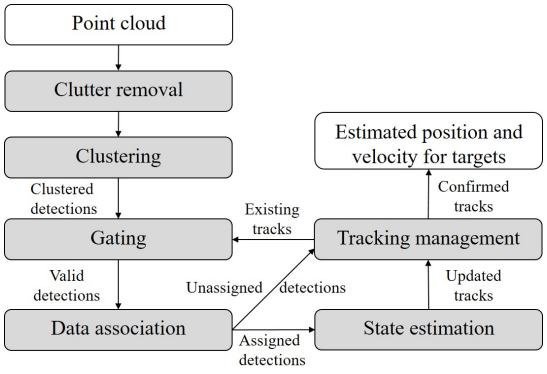


Fig. 2. Block diagram of a typical MOT algorithm

III. Proposed method & Evaluation metrics

As discussed in Section II, ego-motion estimation considers only static objects, while MOT considers moving objects. Thus, there is an advantage in formulating a joint approach for both problems, using all objects wisely and providing the trajectory of both the ego-vehicle and the moving objects at the output.

A. Proposed method

This paper proposes a systematic processing pipeline to jointly approach ego-motion estimation and MOT in automotive radar. The proposed pipeline is shown in Fig. 3. An initial ego-motion estimation is first implemented to estimate coarse ego velocities for each time step and generate the point cloud for the MOT algorithm.

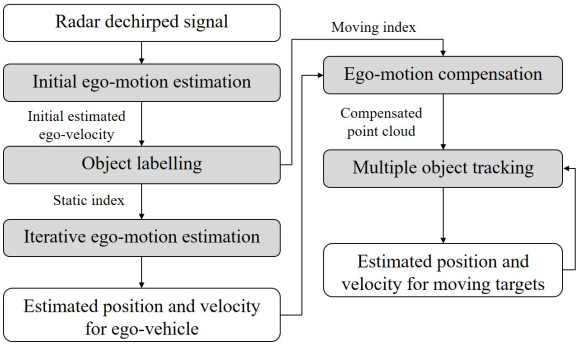


Fig. 3. Block diagram of the proposed joint algorithm for ego-motion estimation & MOT

For this joint formulation where multiple moving & static objects exist in the environment and tracking moving objects is also a part of the main task, static objects act as a major clutter source. To remove them, an object labelling step is used to distinguish moving objects from static objects, which is also used in the iterative ego-motion estimation. The decision logic is to compare the velocity difference between the detected velocity and the ego-motion-induced velocity with a fixed threshold, as follows:

$$\begin{cases} \text{Label moving object} & \text{if } |V - V_{induced}| > \text{Threshold} \\ \text{Label static object} & \text{if } |V - V_{induced}| \leq \text{Threshold} \end{cases} \tag{1}$$

where the ego-motion-induced velocity $V_{induced}$ is calculated by projecting the ego-motion estimation results to the objects' directions, and V denotes the velocity directly obtained after Doppler processing. The threshold is chosen as 1.25 m/s after empirical verifications. Using only the detections of static objects, an iterative ego-motion estimation is performed to obtain the final ego-motion estimation results.

After extracting all detections labelled as moving objects, the clutter caused by static objects is removed. Before using these point clouds in the MOT algorithm, ego-motion compensation is needed. Since the point cloud measures objects in the moving radar coordinates, the detections corresponding to moving objects should be compensated for the radar ego-motion to convert the detections into the world coordinates. In Equation (2), detections after ego-motion compensation are obtained with known ego-motion estimation results in $[V_x^{ego}, V_y^{ego}, V_z^{ego}]$.

$$\begin{bmatrix} dx \\ dy \\ dz \\ v_d \\ p_o \end{bmatrix} = \begin{bmatrix} R * \cos(\theta)\cos(\phi) + V_x^{ego} * \text{Time} \\ R * \sin(\theta)\cos(\phi) + V_y^{ego} * \text{Time} \\ R * \sin(\phi) + V_z^{ego} * \text{Time} \\ |V - V_{induced}| \\ p \end{bmatrix} \quad (2)$$

Specifically, each detection consists of the 3D position $[dx, dy, dz]$, the Doppler velocity v_d and the reflected power p_o. It is worth noting that object labelling and ego-motion compensation link jointly the ego-motion estimation algorithm and MOT. The performance of ego-motion estimation will directly affect the performance of MOT. In this paper, two different MOT algorithms are implemented for comparisons: GNN [6] based on hypotheses pruning, and JPDA [7] based on hypotheses merging.

B. Evaluation metrics

Two metrics are used to evaluate the performance of ego-motion estimation algorithms in 3D space: Absolute Pose Error (APE) and Relative Trajectory Error (RTE).

APE measures the pose difference between the estimation and the true motion for each frame. As shown in (3), it calculates the RMSE between the estimated poses and the actual poses.

$$\epsilon_{APE} = \sqrt{\frac{1}{m}\sum_{i=1}^{m} \|P_{est}(i) - P_{actual}(i)\|^2} \quad (3)$$

where m is the total number of frames, P_{est} and P_{actual} are the estimated velocities and actual velocities, respectively. L-2 norm is used to calculate the Euclidean distance.

While APE focuses on the instantaneous velocity estimation error, RTE measures the long-term localization error. It measures the relative position error over short segments. This is the RMSE of the differences between the relative displacements over a small period in the estimated trajectory and the actual trajectory. As in Equation (4), T_{est} and T_{actual} are the estimated positions and actual positions, respectively. N is the segment interval, which is set to 10 frames (0.05 seconds) in this study.

$$\epsilon_{RTE} = \sqrt{\frac{1}{m-N}\sum_{i=1}^{m-N} \begin{pmatrix} T_{est}(i+N) - T_{est}(i) \\ -T_{actual}(i+N) - T_{actual}(i) \end{pmatrix}^2} \quad (4)$$

To evaluate the MOT results, the Generalized Optimal Sub-Pattern Assignment (GOSPA) [12] metric is used. A list of estimated object 3D position vectors is obtained from the MOT method. Each element represents the tracking position of one object. The GOSPA metric measures the difference or error between the output list and the ground truth list, as:

$$GOSPA(\mathbf{X}, \mathbf{Y}) = \left[\min_{\gamma \in \Gamma} \left(\sum_{(i,j)\in\gamma} \begin{matrix} d(x_i, y_j)^p + \\ \frac{c^p}{2}\left(|\mathbf{X}| - |\gamma| + |\mathbf{Y}| - |\gamma|\right) \end{matrix} \right) \right]^{1/p} \quad (5)$$

where X and Y are two data lists; γ is the global association hypothesis which assigns elements from X to Y. Γ is the set of all possible global association hypotheses. There are three hyperparameter choices for the metric: the order of distance p, the distance metric $d(x, y)$, and the maximum allowable localization error c. The hyperparameter p is set to 2 to make the localization error component the same as the RMSE. The Euclidean distance is chosen as distance metric d. The hyperparameter c determines the trade-off between the localization error component and the missed detection and false alarm number component, and can be considered as the distance where the designer wants to penalize a false or missing estimate. In automotive vehicle perception applications, c is commonly set to 10 meters [13].

IV. Results and Discussion

To validate the performance and show the feasibility of the proposed method, results based on simulations are presented.

An automotive MIMO radar with eight virtual array elements for azimuth and eight for elevation estimation is considered here. The omnidirectional antenna pattern is considered for the transmitter and receiver. A typical parameter setting is used for the FMCW waveform, with 77.5 GHz centre frequency, $20\mu s$ pulse repetition time and 1 GHz bandwidth. The range resolution under this setting is equal to 0.15 m, and the Doppler resolution is 0.378 m/s.

The simulated scenario consists of a car equipped with a side-looking radar moving at a constant speed, alongside multiple static and moving objects. The ego-vehicle is assumed to travel forward at a constant velocity of 12 m/s (43.2 km/h), with zero velocity in both the cross-forward and elevation directions, an assumption that aligns with real-world driving conditions on well-maintained roads. Within the radar's field of view, 15 static objects and 3 moving objects are generated. Each object is modeled as a collection of scatterers randomly distributed in 3D space. The range of these scatterers is selected from $[0, 35]$ meters, the azimuth angle from $[-60, 60]$ degrees, and the elevation angle from $[-30, 30]$ degrees. The amplitude of all scatterers is drawn from the uniform distribution $\alpha_o \sim \mathcal{U}(0, 300)$. According to the Swerling Model I the amplitude can be seen as constant during one coherent processing interval. The scatterers are also assumed to be isotropic and provide constant amplitude and phase during the processing period. The static objects in this scenario represent common roadside elements such as trees, traffic signs, or parked vehicles. Additionally, three moving objects are simulated, representing a car, a bicycle, and a pedestrian. These objects move at constant speeds of 9.11 m/s (33 km/h), 4.12 m/s (15 km/h), and 1.41 m/s (5 km/h), respectively. Their trajectories are designed to capture diverse interactions with the ego-vehicle: the car moves in a nearly parallel direction, the bicycle approaches from the opposite direction, and the pedestrian crosses perpendicularly to the vehicle's path.

To evaluate the precision and robustness of the proposed joint algorithm, simulations were performed in four independent scenarios at SNR levels ranging from +20 dB

to -5 dB, and the results were averaged at each level. In each scenario, the positions of static and moving objects were randomly generated and each simulation lasted 2.56 s (equivalent to 500 consecutive frames). The quantitative performance metrics defined in Section III-B for different SNR levels are shown in Table 1. With the proposed joint method, both the estimation of ego-motion and the tracking of multiple objects can perform a decent estimation even under high noise condition, i.e., SNR = 0 dB. It can be observed that while both the estimation of ego-motion and the performance of MOT decrease with decreasing SNR as expected, the estimation of ego-motion is more stable compared with MOT. The GNN algorithm outperforms JPDA in high-SNR scenarios, but performs worse in low-SNR scenarios. Since the ego-motion estimation remains consistent over consecutive frames and the clutter removal step effectively identifies most static objects, the data association problem is simplified, and both GNN and JPDA MOT algorithms yield good performance.

Table 1. Performance evaluation with different SNR in 3D space

SNR (dB)	APE (m/s)	RTE (m)	Mean GOSPA (GNN)	Mean GOSPA (JPDA)
+20	0.3058	0.0090	2.868	2.995
+15	0.3204	0.0094	3.239	3.681
+10	0.3349	0.0102	3.413	3.283
+5	0.3557	0.0102	3.765	4.698
0	0.4740	0.0123	4.582	5.024
-5	0.8418	0.0213	15.474	13.303

For visual evaluation, the estimated trajectories for the ego-vehicle and moving objects from an example scenario at the 20 dB SNR level are shown in Fig. 4. The figure only displays results in the X and Y dimensions for simplicity in visualization, while all quantitative metrics are computed in the full 3D space. Since the GNN and JPDA algorithms provide similar results, only the results of GNN are visualized here. One can observe the trajectories of the ego-vehicle and three moving objects are all accurately estimated, with small deviations from the ground truth and no false tracks.

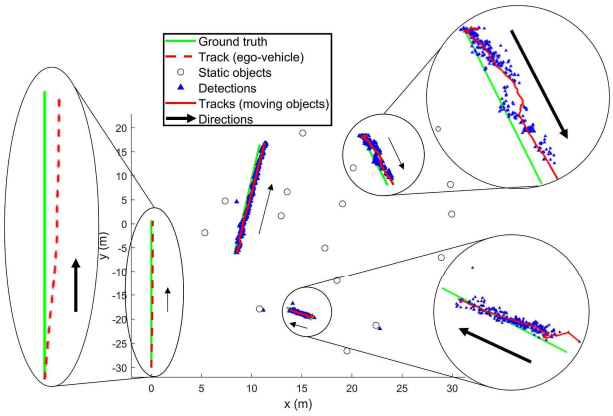


Fig. 4. Ground truth, detections and estimated trajectories for the ego-vehicle and moving objects from an example scenario at the +20 dB SNR level

V. Conclusion

A processing pipeline is proposed to solve the problem of joint ego-motion estimation and MOT in automotive MIMO radar. This pipeline links ego-motion estimation with MOT by labeling static and moving objects to perform ego-motion compensation. Static objects are used for ego-motion estimation, while moving ones are used for MOT algorithms. The performance of the proposed approach has been studied in simulations and the evaluation has been implemented on GNN and JPDA algorithms for feasibility. Promising results are shown for the feasibility of the proposed method.More detailed results can be found in [14]. Future work will involve adopting more realistic motion models, as well as expanding the pipeline to support multiple extended object tracking.

References

[1] D. Kellner, M. Barjenbruch, J. Klappstein, J. Dickmann, and K. Dietmayer, "Instantaneous ego-motion estimation using doppler radar," in *16th International IEEE Conference on Intelligent Transportation Systems (ITSC 2013)*. IEEE, 2013, pp. 869–874.

[2] M. Rapp, M. Barjenbruch, M. Hahn, J. Dickmann, and K. Dietmayer, "Probabilistic ego-motion estimation using multiple automotive radar sensors," *Robotics and Autonomous Systems*, vol. 89, pp. 136–146, 2017.

[3] S. Yuan, D. Wang, F. Fioranelli, and A. Yarovoy, "Improved accuracy for 3d ego-motion estimation using automotive fmcw mimo radar," in *2024 IEEE Radar Conference (RadarConf24)*. IEEE, 2024, pp. 1–6.

[4] S. Yuan, S. Zhu, F. Fioranelli, and A. G. Yarovoy, "3-d ego-motion estimation using multi-channel fmcw radar," *IEEE Transactions on Radar Systems*, vol. 1, pp. 368–381, 2023.

[5] K. Granström and M. Baum, "A tutorial on multiple extended object tracking," *Authorea Preprints*, 2023.

[6] P. D. Konstantinova, A. Udvarev, and T. Semerdjiev, "A study of a target tracking algorithm using global nearest neighbor approach." in *Compsystech*, vol. 3, 2003, pp. 290–295.

[7] Y. Bar-Shalom, F. Daum, and J. Huang, "The probabilistic data association filter," *IEEE Control Systems Magazine*, vol. 29, no. 6, pp. 82–100, 2009.

[8] S. S. Blackman, "Multiple hypothesis tracking for multiple target tracking," *IEEE Aerospace and Electronic Systems Magazine*, vol. 19, no. 1, pp. 5–18, 2004.

[9] Z. Pan, F. Ding, H. Zhong, and C. X. Lu, "Ratrack: Moving object detection and tracking with 4d radar point cloud," *arXiv preprint arXiv:2309.09737*, 2023.

[10] J. Zhang, M. Henein, R. Mahony, and V. Ila, "Robust ego and object 6-dof motion estimation and tracking," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, pp. 5017–5023.

[11] X. Tian, Z. Zhu, J. Zhao, G. Tian, and C. Ye, "Dl-slot: Tightly-coupled dynamic lidar slam and 3d object tracking based on collaborative graph optimization," *IEEE Transactions on Intelligent Vehicles*, vol. 9, no. 1, pp. 1017–1027, 2023.

[12] A. S. Rahmathullah, Á. F. García-Fernández, and L. Svensson, "Generalized optimal sub-pattern assignment metric," in *2017 20th International Conference on Information Fusion*. IEEE, pp. 1–8.

[13] Á. F. García-Fernández, J. L. Williams, K. Granström, and L. Svensson, "Poisson multi-bernoulli mixture filter: Direct derivation and implementation," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 54, no. 4, pp. 1883–1901, 2018.

[14] T. Wang, "Combined ego-motion estimation and multiple extended object tracking with automotive radar," Master's thesis, Delft University of Technology, 2025.