Enhancing Vehicular Navigation in Urban Environments with Radar Point Clouds: A Pre-Filtering and Sensor Fusion Approach

Amélioration de la navigation des véhicules en environnements urbains avec des nuages de points radar : une approche de pré-filtrage et de fusion de capteurs

A Thesis Submitted to the Division of Graduate Studies
of the Royal Military College of Canada

by

Lt(N) Lang, Zheng Yu

In Partial Fulfillment of the Requirements for the Degree of
Master of Applied Science in Electrical Engineering

April, 2025

Abstract

Accurate positioning is crucial for the safe operation of autonomous vehicles, particularly in urban environments where Global Navigation Satellite System (GNSS) signals are degraded due to signal blockages, multipath errors, and environmental obstructions. Inertial Navigation Systems (INS) can provide an alternative positioning solution when GNSS is unreliable by integrating measurements from Inertial Measurement Units (IMUs) to determine the vehicle position, velocity, and attitude. However, low-cost and commercial-grade IMUs found in land vehicles suffer from inherent inertial sensor errors, including bias drift and scale factor instability. These errors cause INS to be reliable only over the short term and require external corrections to remain reliable over mid- to long-term GNSS outages. Exteroceptive sensors such as cameras, lidar, and radar can enhance positioning by providing additional environmental references. Among these, automotive radar operating at 77 GHz is advantageous due to its resilience to adverse weather and varying lighting conditions, as well as its ability to provide radar cross-section (RCS) and Doppler velocity measurements. In particular, positioning systems based on registering radar scans to prior maps of the environment have shown promise as an alternative positioning solution. However, radar-based positioning presents challenges, as low-cost automotive radars are susceptible to noise, ghost detections, and a limited number of detections per scan, all factors which can potentially degrade the performance of radar-based positioning algorithms. This study develops radar point cloud filtering techniques designed to remove dynamic objects and noise from radar data, thereby improving the accuracy of radar scan-to-map registration. The point cloud preprocessing techniques developed in this work employ machine learning and classical filtering techniques, such as velocity-based filtering, geometric clustering, and support vector machine (SVM) classification, to enhance the static environment detected by the radar using Doppler, RCS, and positional information. Furthermore, the study evaluates the performance of two state estimation techniques, the Error State Extended Kalman Filter (ES-EKF) and the Unscented Kalman Filter (UKF), across multiple real-world urban driving scenarios, analyzing their robustness and accuracy. Experimental results demonstrate that the proposed filtering approach improves radar-based positioning in urban environments, improving the reliability of autonomous vehicle navigation in Urban and GNSS-denied conditions.

Résumé

Un positionnement précis est essentiel pour assurer la sécurité des véhicules autonomes, en particulier dans les environnements urbains où les signaux du Global Navigation Satellite System (GNSS) sont dégradés en raison des obstructions, des erreurs de trajets multiples et des interférences environnementales. Les systèmes de navigation inertielle (INS) offrent une solution alternative lorsque le GNSS est indisponible, en intégrant les mesures des unités de mesure inertielle (IMU) pour estimer la position, la vitesse et l'orientation du véhicule. Cependant, les IMU à bas coût et de qualité commerciale, couramment utilisées dans les véhicules terrestres, souffrent d'erreurs inhérentes aux capteurs inertiels, notamment la dérive du biais et l'instabilité du facteur d'échelle. Ces erreurs rendent les INS fiables uniquement à court terme, nécessitant des corrections externes pour maintenir leur précision lors d'interruptions du GNSS de moyenne à longue durée. Les capteurs extéroceptifs, tels que les caméras, le lidar et le radar, peuvent améliorer le positionnement en fournissant des références environnementales supplémentaires. Parmi eux, le radar automobile fonctionnant à 77 GHz présente plusieurs avantages, notamment sa résistance aux conditions météorologiques défavorables et aux variations de luminosité, ainsi que sa capacité à fournir des mesures de section efficace radar (RCS) et de vitesse Doppler. En particulier, les systèmes de positionnement basés sur l'enregistrement des scans radar avec des cartes préexistantes de l'environnement se révèlent être une solution prometteuse pour le positionnement autonome. Cependant, le positionnement basé sur le radar présente plusieurs défis, car les radars automobiles à bas coût sont sensibles au bruit, aux fausses détections (ghost detections) et à un nombre limité de détections par balayage. Ces facteurs peuvent potentiellement dégrader la performance des algorithmes de positionnement radar. Cette étude propose le développement de techniques de filtrage des nuages de points radar

conçues pour éliminer les objets dynamiques et le bruit, améliorant ainsi la précision de l'alignement des scans radar avec la carte de l'environnement. Les techniques de prétraitement des nuages de points développées dans ce travail combinent des approches classiques et de l'apprentissage automatique, notamment le filtrage basé sur la vitesse, le clustering géométrique et la classification par machine à vecteurs de support (SVM). Ces méthodes permettent d'améliorer la représentation de l'environnement statique détecté par le radar, en exploitant les informations Doppler, RCS et de position. De plus, cette étude évalue la performance de deux techniques d'estimation d'état, le filtre de Kalman étendu en état d'erreur (ES-EKF) et le filtre de Kalman non linéaire (UKF), à travers plusieurs scénarios réels de conduite urbaine, en analysant leur robustesse et leur précision. Les résultats expérimentaux montrent que l'approche de filtrage proposée améliore la précision du positionnement radar en milieu urbain, renforçant ainsi la fiabilité de la navigation des véhicules autonomes dans des environnements urbains et en l'absence de GNSS.

Contents

1 Introduction **16**

1.1 Background and Problem Statement 16

1.2 Objectives 18

1.3 Thesis Contribution 19

1.3.1 Thesis Outline 19

2 Literature Review **21**

2.1 Core Positioning Technologies for Land Vehicles 21

2.1.1 Global Navigation Satellite Systems (GNSS) 21

2.1.2 Dead Reckoning (DR) 23

2.2 Inertial Navigation Systems (INS) and Onboard Motion Sensors 25

2.2.1 Reduced Inertial Sensor System (RISS) 26

2.2.2 Common INS Error Sources 29

2.3 Exteroceptive Sensors 29

2.3.1 Camera 30

2.3.2 Light Detection and Ranging (Lidar) 30
2.3.3 Radar 31
2.4 Radar-Based Positioning 32
2.4.1 Radar Odometry Techniques 33
2.4.2 Map Registration 36
2.4.3 Challenges in Radar Based Positioning 38
2.5 Point Cloud Filtering 39
2.5.1 Neural Networks 40
2.5.2 Support Vector Machine (SVM) 41
2.5.3 Density Based Clustering 43
2.6 Sensor Fusion 44
2.6.1 Extend Kalman Filter (EKF) 45
2.6.2 Unscented Kalman Filter (UKF) 45
2.6.3 Multi-Sensor Integration with Radar 46

3 Methodology **48**

3.1 Sensor-to-Vehicle Frame Transformation 48
3.2 Radar Point Cloud Filtering 49
3.2.1 Velocity Filtering 50
3.2.2 Geometric Filtering 52
3.2.3 SVM Classification 52
3.3 Position Estimation 54

3.3.1 Dead Reckoning 55
3.3.2 Map Registration 55
3.4 Sensor Fusion 56
3.4.1 UKF 58
3.4.2 ES-EKF 59
3.5 Reference Map 60
3.5.1 Public Reference Maps 61
3.5.2 Radar-based Reference Map 62
3.5.3 Radar Point Cloud Preprocessing Tuning Parameters . 63

4 Results and Discussion **65**

4.1 Public Datasets 66
4.1.1 RadarScenes 66
4.1.2 NavINST 67
4.2 Experimental Setup 68
4.2.1 RadarScenes Setup 68
4.2.2 NavINST Setup 70
4.3 Evaluation Metrics 75
4.4 Radar Point Cloud Filter Analysis 76
4.4.1 Velocity Filter 76
4.4.2 Geometric Filter 77
4.4.3 SVM Classification 80

4.5 The Effect of Point Cloud Filtering on Trajectory Estimation . 83

4.5.1 RadarScenes 83

4.5.2 NavINST 88

4.6 Real Time Performance Analysis 97

5 Conclusion and Future Work 99

5.1 Conclusion 99

5.2 Future Work 100

List of Tables

4.1 Radar Mounting Parameters for NavINST sensor rack [56] . . 70

4.2 Trajectory Statistics 73

4.3 Classification Report for SVM Training 81

4.4 Classification Report for SVM Testing on a Balanced Trajectory 81

4.5 Classification Report for SVM Model on an Unbalanced Trajectory 81

4.6 Error Metrics for Trajectory 1 across raw, velocity-filtered, and geometric-filtered point clouds. 85

4.7 Error Metrics for Trajectory 2 across raw, velocity-filtered, and geometric-filtered point clouds. 85

4.8 Processing times for ICP computation on velocity-filtered and geometric-filtered point clouds for Trajectories 1 and 2. 86

4.9 Error Metrics for UKF and EKF for Segment 1 90

4.10 Error Metrics for UKF and EKF for Segment 2 91

4.11 Error Metrics for UKF and EKF for Segment 3 93

4.12 Error Metrics for UKF and EKF for Segment 4 95

4.13 Error Metrics for UKF and EKF for Segment 5 97

List of Figures

2.1 Dead Reckoning illustration 24

2.2 Inertial Navigation System Mechanization Flow Chart 26

2.3 Reduced Inertial Sensor System Flow Chart 27

2.4 Illustration of SVM Decision Boundary 42

3.1 Illustration of vehicle forward speed vs detected object's Doppler measurement. 51

3.2 Support Vector Machine training and testing scenarios 53

3.3 Different testing Scenarios with a combination of classical and machine-learning approaches 54

3.4 Positioning Pipeline [18, 51] 56

3.5 Radar Point Cloud Filtering to ICP [51] 57

3.6 Kingston Downtown with OSM [53] 61

3.7 Kingston Suburb with Lidar map 62

4.1 Front View of NavINST Sensor Rack [56] 67

4.2 Reference map built using downsampled, clustered, and Gaussian noise-injected static points. 71

4.3 Illustration of the radar sensors' field of view, demonstrating the 360-degree coverage provided by the four electronically scanning radars mounted on the test vehicle. 72

4.4 Five urban segments located in the City of Kingston, ON [53] . 73

4.5 Velocity filter with a threshold of 0.3 m/s applied. 78

4.6 Geometric filter applied static points returned from the velocity filter, dynamic detection and noise are also plotted for visualization. 79

4.7 SVM prediction for vehicles, the left plot is the raw point cloud, the middle is when geometric filtering is applied, and the right plot is using SVM Predicted data. 82

4.8 Comparison of Estimated and Ground Truth for Trajectory 1. . 84

4.9 Comparison of Estimated and Ground Truth for Trajectory 2. . 86

4.10 Trajectory estimation for segment 1 89

4.11 Trajectory estimation for segment 2 92

4.12 Trajectory estimation for segment 3 94

4.13 Trajectory estimation for segment 4 96

4.14 Trajectory estimation for segment 5 98

List of Acronyms

2D Two-Dimensional.

3D Three-Dimensional.

ADAS Advanced Driver Assistance Systems.

AV Autonomous Vehicle.

CNN Convolutional Neural Network.

DBScan Density-Based Spatial Clustering of Applications with Noise.

DR Dead Reckoning.

EKF Extended Kalman Filter.

ES-EKF Error-State Extended Kalman Filter.

ESR Electronically Scanning Radar.

FMCW Frequency-Modulated Continuous Wave.

GNSS Global Navigation Satellite System.

Hz Hertz.

ICP Iterative Closest Point.

IMU Inertial Measurement Unit.

INS Inertial Navigation System.

KF Kalman Filter.

Lidar Light Detection and Ranging.

m Meter.

ms Millisecond.

MSE Mean Square Error.

NavINST Navigation and Instrumentation Laboratory.

OBBs Oriented Bounding Boxes.

ODO Odometer.

OSM OpenStreetMap.

PPP Precise Point Positioning.

RANSAC Random Sample Consensus.

RCS Radar Cross Section.

RISS Reduced Inertial Sensor System.

RMC Royal Military College of Canada.

RMS Root Mean Square.

RMSE Root Mean Square Error.

RTK Real-Time Kinematic.

SD Standard Deviation.

SLAM Simultaneous Localization and Mapping.

SVM Support Vector Machine.

UKF Unscented Kalman Filter.

UTM Universal Transverse Mercator.

VO Visual Odometry.

WAAS Wide Area Augmentation System.

Chapter 1

Introduction

1.1 Background and Problem Statement

Autonomous vehicles (AVs) rely on accurate and robust positioning to ensure safe navigation, particularly in urban environments where positioning uncertainty can result in navigation errors and potential hazards. Positioning, the process of determining a vehicle's position and orientation within its environment, is fundamental to path planning, obstacle avoidance, and decision-making in AVs. Traditional positioning systems, such as those based on the Global Navigation Satellite System (GNSS), are employed due to their global coverage and ease of implementation. However, GNSS-based positioning faces significant limitations in urban environments, where high-rise buildings, tunnels, and dense foliage can block or reflect satellite signals, leading to multipath interference and degraded accuracy [1]. In such scenarios, GNSS alone cannot provide the sub-meter positioning accuracy required for safe and reliable autonomous navigation.

To address these limitations, Inertial Navigation Systems (INS) are often integrated into vehicle positioning frameworks. INS relies on Inertial Measurement Units (IMUs) encompassing three mutually orthogonal accelerometers and gyroscopes, as well as other onboard motion sensors, such as vehicle speedometers, to estimate vehicle motion through dead reckoning [1]. Although INS provides short-term positioning accuracy, it suffers from drift and bias accumulation over time, necessitating external corrections to maintain long-term reliability [2].

Exteroceptive sensors such as Lidars, cameras, and radar are increasingly integrated into positioning systems to compensate for these challenges. Lidar-based positioning offers high-resolution 3D mapping, but its performance deteriorates in adverse weather conditions such as rain and fog. Camera-based positioning is cost-effective but suffers from low visibility in extreme lighting conditions and environmental occlusions [3, 4, 5]. In contrast, automotive radar operating around the 77 GHz range provides a robust alternative for positioning estimation in urban environments. Unlike Lidar and cameras, radar is unaffected by lighting conditions and can operate in rain, fog, and dust [6, 7]. Additionally, radar provides unique measurement capabilities, such as Doppler velocity and Radar Cross-Section (RCS), which can be leveraged for motion estimation and object classification.

Despite its advantages, radar-based positioning presents several challenges that must be addressed to achieve reliable and accurate positioning. First, radar generates a sparse point cloud compared to Lidar, making feature extraction and map registration significantly more challenging [8, 9]. Second, radar signals are prone to multipath interference and ghost detections, where signals reflect off metallic surfaces, leading to false detections and degraded positioning accuracy [7, 10]. These challenges highlight the need for advanced radar point cloud filtering techniques to refine radar data before it is used in a radar-based positioning solution. Pre-processing the point cloud enhances the representation of static environments, which is critical for accurate map registration by removing dynamic objects and reducing noise. False detections and moving objects in radar scans can lead to incorrect velocity estimates and erroneous scan matching, introducing positioning errors. To mitigate these issues, filtering techniques are necessary to refine radar data, ensuring more accurate scan-to-map alignment and improving overall positioning reliability. This research explores pre-filtering techniques for removing noise and dynamic objects in urban environments to achieve robust and accurate radar-based positioning.

Multi-sensor fusion further enhances positioning estimation robustness by integrating radar with onboard motion sensors and map-based corrections.

While Iterative Closest Point (ICP) algorithms are commonly used for scan registration, their performance degrades when non-static detections introduce registration errors. This research aims to improve map-matching accuracy, reduce positioning drift, and enhance radar-based positioning reliability in GNSS-denied urban environments through the development of advanced radar filtering approaches.

1.2 Objectives

This research aims to improve radar-based positioning accuracy in urban environments by developing and evaluating the proposed radar point cloud filtering techniques. It refines radar point clouds by removing noise, dynamic objects, and non-landmark detections, thereby enhancing scan-to-map registration for vehicle positioning.

The specific objectives of this study are:

- Develop and evaluate classical and machine learning-based filtering techniques to eliminate dynamic objects and noise from radar-generated point clouds.
- Integrate filters into existing radar-based positioning algorithms and evaluate positioning performance.
- Compare the performance of the existing Error State Extended Kalman Filter (ES-EKF) algorithm with the developed Unscented Kalman Filter (UKF) for fusing radars with vehicular onboard motion sensors.
- Analyze the impact of radar filtering on vehicle positioning in urban environments utilizing vehicular road tests involving automotive radars and IMUs.

By achieving these objectives, this research aims to enhance the robustness and reliability of radar-based positioning, making it more suitable for autonomous navigation in GNSS-denied urban environments.

1.3 Thesis Contribution

This thesis contributes to addressing the challenges faced by radar-based positioning estimation for autonomous vehicles by integrating onboard motion sensors with radar map registration, developing radar point cloud filtering techniques, and comparing the performance of two sensor fusion algorithms: ES-EKF and UKF. The contributions of this research are:

- **Development of a Radar Point Cloud Filtering Framework**: Develop velocity and geometric filters to remove dynamic detections and noise and use machine learning-based classification to further refine radar point clouds for map-matching algorithms.
- **Integration of Radar Scan Pre-Processing Methods into Positioning Algorithm**: Filtering techniques are applied to an existing ICP-based positioning framework, demonstrating their impact on trajectory estimation accuracy.
- **Validation on Real-World Urban Datasets**: The proposed algorithms, including radar point cloud filtering and sensor fusion using ES-EKF and UKF, are validated on multiple urban trajectories, demonstrating significant improvements in positioning performance compared to raw radar point cloud data.

1.3.1 Thesis Outline

This thesis is structured as follows:

- **Chapter 2: Literature Review** - Discusses the challenges of GNSS-based positioning in urban environments and reviews existing radar-based positioning techniques, sensor fusion strategies, and radar point cloud filtering methods.
- **Chapter 3: Methodology** - Presents the proposed radar filtering framework, detailing the velocity filter, geometric filter, and SVM classification approach, followed by a positioning solution using both ES-EKF and UKF implementations.

- **Chapter 4: Results and Discussion** - Analyzes the experimental findings, discusses the limitations of each point cloud filter, and evaluates their effectiveness. It also provides a comparative analysis of UKF and ES-EKF positioning solutions for different urban trajectories.

- **Chapter 5: Conclusion** - Summarizes the key findings, discusses limitations of radar-based positioning, and proposes future research directions for enhancing radar-based vehicle positioning.

Chapter 2

Literature Review

2.1 Core Positioning Technologies for Land Vehicles

Accurate vehicle positioning is essential for autonomous navigation, enabling safe and reliable operation in various environments. Core positioning technologies for land vehicles include various methods, each having unique advantages and limitations. This section explores the primary techniques used for vehicle positioning, including Global Navigation Satellite Systems, and dead reckoning (DR). While GNSS provides absolute positioning, its performance degrades in urban environments due to signal blockages and multipath effects. Alternative solutions are offered by DR, to which category INS belongs, which estimate position states using onboard motion sensors but suffer from cumulative drift in positioning errors over time.

2.1.1 Global Navigation Satellite Systems (GNSS)

GNSS is utilized for a broad range of positioning applications, with vehicle positioning being one of the primary users. It is reliable for most conditions, but its performance deteriorates in urban environments and is susceptible to jamming. Jamming occurs when signals from GNSS satellites are overwhelmed by stronger signals, effectively preventing the receiver from acquiring or maintaining satellite lock, thereby degrading positioning accuracy. For civilian applications, GNSS is accurate to 10 meters for most environmental conditions [1]. However, this is considered insufficient for safe autonomous driving.

GNSS Error Sources

In urban areas, GNSS positioning can be significantly impacted by the following errors [1]:

- **Ionospheric and Tropospheric Error**: The ionospheric and tropospheric layers of the Earth's atmosphere cause delays in the GNSS signals due to refraction, changing the signal's transit time.
- **Multipath Error**: A major source of error in urban environments where GPS signals take multiple paths to the receiver due to reflection off high-rise buildings and other structures.
- **Dilution of Precision**: A measure of the geometric distribution of satellites visible to the receiver, which, if unfavourable, can lead to inaccurate position measurements.

GNSS Augmentation Methods

To improve accuracy and reliability, various augmentation techniques have been developed to mitigate GNSS errors:

- **Differential GPS**: Uses a network of well-surveyed ground stations to broadcast the error between GPS position and its known position.
- **Wide Area Augmentation System (WAAS)**: Used by the US Federal Aviation Administration to eliminate errors caused by ionospheric delay to aid aviation navigation, requires additional geostationary satellites.

- **Real Time Kinematic (RTK)**: Uses Ground stations to correct GPS errors in real-time by measuring the carrier cycles. It is capable of centimetre-level accuracy, but achieving this precision requires additional resources, like higher quality receivers and increased computational processing to handle real-time corrections. The receiver also needs to be within range of a base station.

- **Precise Point Positioning (PPP)**: Requires a network of ground stations to measure and generate a PPP solution, which depends on the satellite clock and orbital corrections, and the solution is broadcast to users via the internet.

The availability of augmented GNSS cannot be relied upon due to the service's cost and the range to a reference station. Additionally, if GNSS signals are obstructed or absent, any form of augmentation methods would also be absent.

2.1.2 Dead Reckoning (DR)

DR is a broad category of positioning methods that estimate a vehicle's current position based on a previously determined position, speed, and heading direction. Unlike GNSS, which relies on external signals, DR methods are an independent positioning systems that use onboard motion sensors such as odometers and IMUs. While DR operates independently regardless of the external environment, its accuracy deteriorates over time due to cumulative sensor errors, necessitating error mitigation techniques.

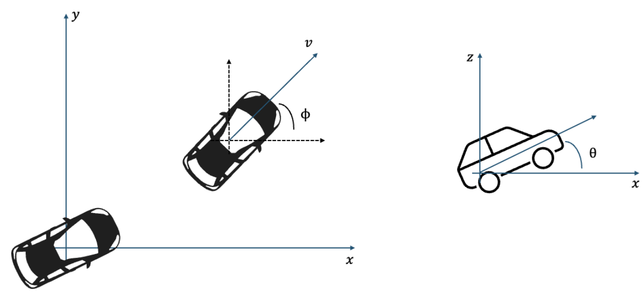


Figure 2.1: Dead Reckoning illustration

The fundamental principle of dead reckoning involves estimating a vehicle's position based on the previous location and motion parameters, shown in Figure 2.1. The positional update at time $k+1$ can be mathematically described by:

$$x_{k+1} = x_k + v\Delta t * \cos\phi_k * \cos\theta_k \tag{2.1}$$

$$y_{k+1} = y_k + v\Delta t * \sin\phi_k * \cos\theta_k \tag{2.2}$$

$$z_{k+1} = z_k + v\Delta t * \sin\theta_k \tag{2.3}$$

Where:

- x_k, y_k, z_k are the vehicle's coordinates at time k,
- ϕ_k is the heading direction,
- θ_k is the pitch angle,
- Δt is the time difference between sampling periods,
- v is the velocity of the vehicle.

For intelligent vehicle navigation, DR is a key component of local positioning systems. One approach utilizes a sensor fusion filter to correct odometer errors, while a linear error model compensates for gyroscope drift [11]. The vehicle speed estimation is refined by fusing the odometer data with IMU acceleration readings, reducing cumulative errors in position estimation.

Dead Reckoning Limitations

The main challenges faced by DR include:

- **Sensor noise and biases:** IMUs and odometers introduce errors that grow over time.
- **Wheel slip and environmental effects:** Uneven terrain and slippage will affect odometry accuracy.
- **Lack of external reference:** Unlike GNSS, DR lacks absolute positioning, requiring periodic corrections from external sources like map-matching or GNSS when available.

2.2 Inertial Navigation Systems (INS) and Onboard Motion Sensors

Onboard motion sensors, including odometry sensors, wheel encoders, and vehicle speedometers, provide valuable proprioceptive data for vehicle positioning, particularly when integrated with an INS. An INS mounted on a vehicle can estimate the position, velocity, acceleration, and heading using an IMU [1]. The measurements are gathered by monitoring the linear and angular acceleration observed by the IMU. The accelerometers and gyroscopes measure the Earth's gravitational and rotational forces, and the vehicle's own motion.

INS mechanization begins with attitude determination, where the gyroscopes measure angular rates around the roll, pitch, and yaw axes. These angular rates are integrated over time to update the vehicle's orientation. Once

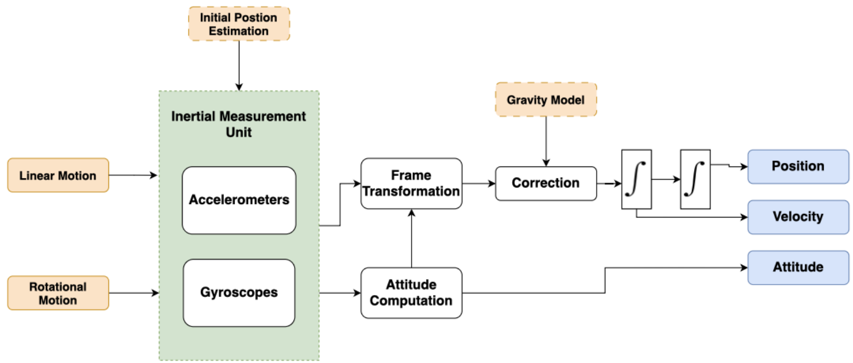


Figure 2.2: Inertial Navigation System Mechanization Flow Chart

orientation is established, the velocity is updated by transforming the accelerometer readings from the body to the navigation frame while compensating for gravitational effects. The final step involves integrating velocity over time to determine position changes, shown in Figure 2.2. This allows the vehicle to estimate its trajectory without reliance on external signals. Over time, small errors in gyroscope and accelerometer measurements accumulate, leading to position drift that increases exponentially without external corrections [1].

2.2.1 Reduced Inertial Sensor System (RISS)

To mitigate the high computational cost and hardware complexity of a full INS, RISS provides a more practical alternative for land vehicle navigation. Instead of using a full three-axis gyroscope setup, RISS simplifies the system by replacing two gyroscopes with an odometer sensor, which provides direct velocity measurements, shown in Figure 2.3. The system consists of three accelerometers for measuring linear acceleration, a single gyroscope along the z-axis for attitude estimation, and an odometer sensor that provides forward velocity measurements. This configuration significantly reduces the number of gyroscope error sources while maintaining sufficient accuracy for land vehicle applications.

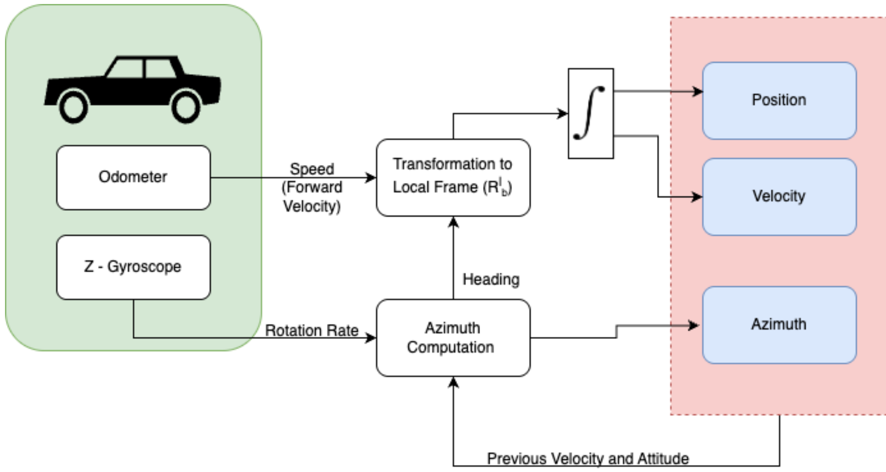


Figure 2.3: Reduced Inertial Sensor System Flow Chart

One of the key advantages of RISS over a full INS is its ability to reduce drift accumulation. In a traditional INS, errors accumulate in all three gyroscope axes, leading to rapid position degradation over time. By using only a single gyroscope for yaw measurements and relying on odometry for velocity estimation, RISS limits the rate at which angular drift grows. Additionally, instead of integrating acceleration to derive velocity, RISS directly incorporates odometry readings, which are less prone to cumulative errors. The critical difference is that instead of two integrations required to obtain position updates, only one integration is needed. This also improves computational efficiency, as a full INS requires complex sensor fusion algorithms to compensate for errors, simplifying the estimation process. The accelerometers measure the gravitational force effect along the three axes, providing pitch and roll, while the gyroscope measures the Earth's rotation rate and the vehicle's turn rate to determine the azimuth. The odometer provides forward velocity, which allows for continuous estimation of position, velocity, and acceleration when integrated with other sensor data. However, long-term reliance on dead reckoning without external corrections leads to error growth, making periodic updates from GNSS or map-based positioning necessary.

The procedure to transform the IMU measurements to obtain an updated position and attitude estimate is given in the following equations.

R^l_b provides the transformation matrix from the body frame to the local level frame. It transforms sensor data from the vehicle's coordinate system to the local navigation frame.

$$R_b^l = \begin{bmatrix} \cos(y)\cos(r) - \sin(y)\sin(p)\sin(r) & -\sin(y)\cos(p) & \cos(p)\sin(r) \\ \sin(y)\cos(r) + \cos(y)\sin(p)\sin(r) & \cos(y)\cos(p) & \sin(p) \\ -\cos(p)\sin(r) & \sin(p) & \cos(p)\cos(r) \end{bmatrix} \tag{2.4}$$

Where p is the pitch angle (rotation around the y-axis), r is the roll angle (rotation around the x-axis), and y is the yaw angle (rotation about the z-axis).

The accelerations read by the accelerometers can be described by:

$$\begin{bmatrix} f_x \\ f_y \\ f_z \end{bmatrix} = \begin{bmatrix} -\cos(p)\sin(r) \\ \sin(p) \\ \cos(p)\cos(r) \end{bmatrix} \begin{bmatrix} -g \end{bmatrix} = (R_b^l)^T \begin{bmatrix} 0 \\ 0 \\ -g \end{bmatrix} \tag{2.5}$$

Where f_x, f_y, and f_z are the specific forces measured by the acceleration along the body-frame, and g is the local gravitational acceleration. Rearranging the above equations provides the vehicle's p and r.

The Azimuth or Yaw can then be obtained using the gyroscope's measured rate of turn, the Earth's rotation, and the last known latitude:

$$Azi = -\left(\cos(p)\cos(r)\omega_z - \omega^e \sin(\phi) - \tfrac{V_e \tan(\phi)}{R+h}\right) = -Yaw \tag{2.6}$$

Where Azi is the azimuth rate in the local level frame, w_z is the yaw rate measured by gyroscope, w^e is the Earth's rotation rate, R is the Earth radius, and h is the altitude above the Earth's surface.

The vehicle's speedometer data is transformed into velocities in the local level frame below:

$$\begin{bmatrix} V_e \\ V_n \\ V_u \end{bmatrix} = R_b^l \begin{bmatrix} 0 \\ V_(od) \\ 0 \end{bmatrix} \tag{2.7}$$

Once the velocity of the vehicle in the local level frame is known, the changes in Latitude($\dot{\phi}$), Longitude($\dot{\lambda}$), and Altitude($\dot{h}$) can be determined:

$$\begin{bmatrix} \dot{\phi} \\ \dot{\lambda} \\ \dot{h} \end{bmatrix} = \begin{bmatrix} 0 & \frac{1}{R+h} & 0 \\ \frac{1}{(R+h)cos(\phi)} & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} V_e \\ V_n \\ V_u \end{bmatrix} = D^{-1}V^l \tag{2.8}$$

For conversion to the cartesian coordinate frame:

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} (R_N + h)\cos\phi\cos\lambda \\ (R_N + h)\cos\phi\sin\lambda \\ (R_N(1 - e^2) + h)\sin\phi \end{bmatrix} \tag{2.9}$$

where: R_N is the normal radius and h is the elipsoidal height [1].

2.2.2 Common INS Error Sources

Common INS errors include [1]:

- **Bias**: Variations in gyroscope and accelerometer bias that accumulate over time.
- **Noise**: Measurement noise that introduces random variations to the measurements.
- **Misalignment**: If there is a misalignment between the axes of the IMU and the body frame of the vehicle which is not properly calibrated, errors will be present in the transformation of accelerations and velocities from sensor to body frame.
- **Integration**: INS estimates position by integrating the acceleration measurement twice, with the result that small biases become large positional errors over time.

2.3 Exteroceptive Sensors

Exteroceptive sensors provide environmental perception capabilities essential for autonomous vehicle positioning and navigation. Unlike proprioceptive sensors, which measure internal motion and dynamics, exteroceptive sensors gather external information about the vehicle's surroundings. These sensors include cameras, Lidar, and radar, each offering unique advantages and limitations. Cameras capture high-resolution visual data for feature recognition and object detection, while Lidar generates detailed 3D point clouds for precise mapping. Conversely, radar excels in all-weather conditions and provides Doppler velocity measurements for motion estimation. By integrating these sensors, situational awareness and positioning accuracy can be enhanced.

2.3.1 Camera

Cameras provide a cost-effective and widely available solution for vehicle positioning by capturing visual information from the surrounding environment. Monocular and fisheye cameras are commonly used for positioning and mapping. A monocular camera-based vision system estimates vehicle position by detecting key environmental features and comparing them to pre-existing maps [12] or through Visual Odometry (VO), which tracks feature points between consecutive frames to estimate motion without requiring a prior map. However, depth estimation with a single camera is challenging and requires additional processing or stereo camera setups. Fisheye cameras are particularly useful in constrained environments such as indoor parking lots, where they offer wide-angle coverage for accurate vehicle tracking [13]. These systems use image segmentation and distortion correction techniques to improve position estimation. Despite their advantages, camera-based positioning systems are highly sensitive to lighting conditions, occlusions, and dynamic environmental changes. Vision-based positioning is often combined with Lidar or radar data to enhance accuracy, leveraging the strengths of each sensor to improve overall positioning performance.

2.3.2 Light Detection and Ranging (Lidar)

Lidar is widely recognized for its high-precision environmental mapping capabilities. Lidar sensors generate dense 3D point clouds that enable accurate positioning and obstacle detection. Unlike radar, which primarily measures range and velocity, Lidar provides detailed spatial information, allowing vehicles to map and interpret their surroundings with centimetre-level accuracy. Recent developments in Lidar-based positioning involve multi-sensor fusion with inertial sensors and high-resolution 3D digital maps to enhance positioning accuracy [14]. This approach enables precise vehicle positioning in urban environments where GNSS signals may be unreliable. Point cloud compression techniques further improve computational efficiency, making real-time processing feasible for autonomous driving applications. Despite its high accuracy, Lidar faces several challenges. The cost of Lidar sensors remains relatively high compared to other positioning technologies, limiting widespread adoption. Additionally, Lidar's performance can degrade in adverse weather conditions such as heavy rain or fog. Lidar is often used in conjunction with radar and vision-based systems to mitigate these limitations and improve robustness.

2.3.3 Radar

Radar point cloud-based positioning methods have emerged as a robust method for vehicle positioning, offering resilience in challenging environmental conditions such as low visibility, fog, rain, and darkness [15]. Unlike Lidar and camera-based approaches, radar sensors use radio waves, which penetrate through obstructions like fog and precipitation, making them highly reliable for all-weather positioning [16]. Automotive radar systems, primarily based on Frequency-Modulated Continuous Wave (FMCW) technology, generate point clouds by measuring the distance, velocity, and angle of surrounding objects [15]. The radar transmits frequency-modulated signals, which reflect off objects in the environment and return to the receiver. By analyzing the time delay, frequency shift (Doppler effect), and phase differences of these signals, the system constructs a sparse but informative point cloud representation of the vehicle's surroundings [17].

2.4 Radar-Based Positioning

Radar-based positioning has emerged as an alternative for vehicle positioning, particularly in GNSS-denied environments. Automotive radar systems measure range, azimuth, Doppler velocity, and RCS, providing valuable motion and object classification data. However, radar-based positioning presents challenges [18], including sparse point clouds, multipath reflections, and dynamic object interference, which must be addressed through filtering and sensor fusion techniques.

Radar-Based Mapping and Positioning

To utilize radar point clouds for vehicle positioning, several processing steps are applied:

1. **Preprocessing and Filtering**: The raw radar returns often contain noise due to multipath reflections [16]. Signal processing techniques such as thresholding, clustering, and outlier rejection are used to filter out irrelevant or spurious reflections.
2. **Feature Extraction**: Radar systems extract key environmental features from the point cloud, such as road boundaries, static structures, and moving objects. Unlike Lidar, which captures dense point clouds, radar generates sparser but motion-sensitive data valuable for dynamic object tracking [17].
3. **Multi-Frame Integration**: By integrating radar data over multiple frames, a more comprehensive environmental model is built, improving spatial consistency [16].
4. **Registration and Map Matching**: Radar point clouds can be matched against prebuilt high-definition radar maps or combined with Simultaneous Localization And Mapping (SLAM) algorithms [17] to provide positioning updates. Since radar is less affected by environmental variations, radar maps are highly stable over time, making them ideal for long-term positioning.

5. **Sensor Fusion for Enhanced Localization**: Radar point cloud mapping is often combined with Lidar, cameras, and inertial sensors to improve positioning accuracy [14]. The system achieves robust and redundant localization by integrating Lidar's high-resolution spatial data, camera-based feature recognition, and IMU/GNSS information.

2.4.1 Radar Odometry Techniques

RO is a method of estimating a vehicle's motion to obtain updated positioning estimates over time. The concept is to measure changes in position by analyzing radar data over consecutive scans. The changes in position can be determined through different approaches, some of which involve using Doppler velocity, and others rely on associating consecutive radar scans to identify the translation and rotation.

Saussard et al. [19] proposed a method to estimate the vehicle's 2D motion using radar odometry to enhance ego-motion estimation. The authors highlighted the limitations of wheel-based odometry with wheel slip in adverse environmental conditions. The proposed method uses two front-facing FMCW radars mounted on the left and right corners of the vehicle to detect objects. The algorithm includes data association, linear regression, and outlier rejection to determine the vehicle's motion. The radar data is fused with proprioceptive sensors to increase the vehicle's position estimation accuracy. Relative velocity must be taken from static objects to determine the vehicle's velocity.

Similar work has been done by Kellner et al. [6, 20]. The difference is how they obtained the stationary targets. The unique solution for their method is that the exact position and radar cross section (RCS) are not required to detect stationary targets. They proposed that if the sensor moves, all stationary objects have the same movement in the opposite direction. A Random Sample Consensus (RANSAC) algorithm is used to classify the stationary and non-stationary objects as inliers and outliers, respectively. In each iteration of the RANSAC algorithm, two targets are randomly chosen to determine the sensor velocity V_s and heading direction α. The V_s, α are fitted to all radar points to determine the best fit. After a certain amount of iterations, the best-fit model of V_s, α is taken to be the sensor's velocity and heading angle. In their experiment with standalone RO within a parking lot, they had an error standard deviation (SD) of 0.029m/s for velocity and 0.56 $\deg$/s. However, this method yielded a less desirable result for an urban environment, resulting in a velocity SD of 5m/s. The results show this method's efficiency in controlled conditions while showing that further enhancements must be made to improve its functionality in diverse environments.

The typical process for determining the speed of the vehicle using an automotive radar mounted on the vehicle is described below.

When the radar point clouds are received, they contain the position in cartesian coordinates relative to the sensor, angle, signal-to-noise ratio, Doppler velocity, and elevation. Only the Doppler velocity and arrival angle are used. Critically, only the information from static environmental objects contain useful information for the estimation of the velocity of the vehicle.

The speed of the sensor, v_s, mounted on the vehicle is the negative of the relative velocity V_r of the static objects around the moving sensor. Thus, the radial velocities of the static objects with respect to the sensor are equal to the negative Doppler velocities which are measured by the radar.

$$-V_r = V_D \tag{2.10}$$

For a radar scan with N object detections, Equation 2.11 demonstrates the relationship between detected radial velocities $v_{r,N}$ and the sensor's motion in the x and y directions, $v_s x, v_s y$. θ represents the azimuth angle of the detected object relative to the vehicle.

$$\begin{bmatrix} v_{r,1} \\ \vdots \\ v_{r,N} \end{bmatrix} = \begin{bmatrix} cos(\theta_1) & sin(\theta_1) \\ \vdots & \vdots \\ cos(\theta_N) & sin(\theta_N) \end{bmatrix} \begin{bmatrix} v_{sx} \\ v_{sy} \end{bmatrix} \tag{2.11}$$

RANSAC can be used to solve Eq. 2.11 for the sensor velocities v_{sx} and v_{sy}, represented in the vehicle frame. RANSAC will be successful in resolving the sensor velocity if the largest velocity group belongs to static objects. This could fail, for example, if a large percentage of the radar's returns came from a fleet of vehicles moving at the same speed. Using the Ackermann steering model, the sensor velocities can be transformed into the vehicle frame:

$$v_s = \sqrt{v_{sx}^2 + v_{sy}^2} \tag{2.12}$$

$$\alpha = \arctan\left(V_{sy}/V_{sx}\right) \tag{2.13}$$

$$v_x = -cos(\alpha)v_s \ \ and \ \ v_y = -sin(\alpha)v_s \tag{2.14}$$

$$v = (cos(\alpha + \beta) - \frac{b}{l} sin(\alpha + \beta))v_s \tag{2.15}$$

$$\omega = \frac{sin(\alpha + \beta)}{l} v_s \tag{2.16}$$

In these equations, v_x, and v_y represent the vehicle's velocity in the x and y directions in the vehicle frame. v is the forward velocity of the vehicle, and ω is its rate of turn. α is the azimuth angle of the sensor velocity in the vehicle frame, β is the sensor's mounting angle, l and b are the sensor's mounting position relative to the vehicle's rear axle [21].

Kellner et al. [6] proposed a Doppler-based RO method that estimates the vehicle's velocity and yaw rate from a single radar scan. This method leverages the radial velocities of detected objects within the radar's field of view. For accurate motion estimation, a sufficient number of detected objects must be stationary, providing reliable reference points for determining the vehicle's motion.

2.4.2 Map Registration

Iterative Closest Point (ICP), a foundational algorithm for point cloud registration, was first introduced by Besl and McKay [22]. The algorithm aligns two point clouds by iteratively minimizing the difference between their corresponding points. ICP can be used to provide position corrections by aligning a pre-existing environmental map with a current radar or Lidar scan from the vehicle as it travels through the environment. While ICP is robust in static and controlled environments, its performance degrades significantly when applied to noisy or sparse point clouds, such as those generated by automotive radar. Recent work for radar scan-to-map registration in covered parking garages proposed by Dawson [23, 18]. The proposed navigation system achieved an accuracy of under 50 centimetres 66% of the time, and 97% of the time the accuracy was within one meter. The proposed solution uses a forward-mounted radar to perform radar odometry, four electronic scanning radars (ESR) mounted on each vehicle corner to perform map matching with the generated point cloud, and a dead reckoning solution from the inertial sensors. Automotive radars provide sparse point clouds. To bypass this limitation, radar scans are aggregated over time to increase point cloud density for map registration using the ICP algorithm [24, 23].

The general Algorithm for ICP [22] is:

- **Initial Alignment**: Collect radar scans, remove dynamic objects, and take an initial pose guess.
- **Nearest Neighbour Search**: Match each point in the scan to the closest point in the reference map, setting the correspondence based on proximity.
- **Error Minimization**: Calculate the distance, or error, between each corresponding point in the radar point cloud and the reference map. Adjust the translation and rotation of the point cloud to minimize the error. Further optimization is performed at this stage, like singular value decomposition.
- **Iterate**: Repeat the process until the error falls below an acceptable threshold.

- **Position Estimation**: Interpret the transformation between the two point clouds and apply it to the vehicle's position estimation as a correction.

L. Narula et al. [24] proposed a solution that demonstrated sub-50cm accuracy at 95% of the time. The authors first created a geo-referenced radar map of the environment for positioning. During simulation, 5s batches of radar data are aggregated, creating a dense point cloud matched to the pre-existing radar map using a global optimization algorithm. Their solution is computationally efficient once the a priori maps have been generated, and it avoids local minima caused by repeating patterns in the urban radar environment.

E. Ward et al. [9] proposed a solution using an ICP algorithm for radar positioning fused with INS, showing sufficiently accurate results. In their experimentation, the Root Mean Square (RMS) lateral and longitude errors over a 5.4km long path were 7.4cm and 37.7cm, respectively. The novel aspect of their work is using a solid-state monopulse short-range radar fused with a high-precision GPS/IMU unit. The authors created their own reference map by driving the experiment route using RTK-GPS for ground truth; the radar point cloud is generated by aggregating 10 scans and converting them into Universal Transverse Mercator (UTM) coordinates.

S.H. Cen et al. [25] proposed a robust solution using a mechanically scanning radar that does not rely on an a priori map. Their unique approach is to find the largest subset between two point clouds that have overlap. This allows for comparing the overlap areas to determine the motion of the vehicles. They introduced a method that processes power-range spectra from the raw radar data to identify landmarks and reject noise. Data association between two consecutive scenes allows for the pose estimation. This process uses their novel algorithm that uses unary descriptors and pairwise compatibility scores. The output of the data association step is the largest subset of the landmarks between two scans for the computation of relative motion. During their simulation over a 10km route through Oxford, UK, the error in translation was 0.106 m/s and 0.321 deg/s in rotation [25].

2.4.3 Challenges in Radar Based Positioning

Sparse Radar Returns

The main challenge with using FMCW radars for automotive positioning is the sparsity of radar point returns, especially in low-speed scenarios. This problem is exacerbated in urban environments where map matching of the landmarks is required for positioning. In urban environments with dense buildings, static and dynamic vehicles, pedestrians, and other obstacles exist. The automotive radar returns can have an unevenly distributed point cloud because of the obstacles, making it more difficult for the map-matching algorithm to achieve reliable positioning. One approach to mitigate this issue is to aggregate successive radar scans to increase the density of detections. However, aggregating too many scans can introduce motion distortion, where errors accumulate due to the vehicle's movement between scans.

Motion Distortion

As the sensor moves, the returns from objects are distorted due to the relative motion of the vehicle, which can lead to inaccuracies in position estimation. Motion distortion is more pronounced in mechanically scanning radars, where the sensor sweeps across the scene over time, causing misalignment in the collected data. In contrast, automotive FMCW radars operate at a higher update rate, capturing more frequent scans, which helps reduce distortion. However, when multiple scans are aggregated, some distortion may still occur, especially at higher vehicle speeds, where rapid motion increases misalignment between consecutive scans. To correct for this, INS or odometry sensors can be used to compensate for motion. However, since INS is subject to drift over time, aggregating scans over an extended period can degrade accuracy.

Low Vertical Resolution

Automotive sensors typically have high horizontal range resolution but suffer from low vertical resolution. High horizontal resolution allows it to identify objects in the vehicle's path for ADAS. The low vertical resolution would hinder the ability to perform 3D map matching for positioning; this can be mitigated by using machine learning or projecting the points onto a 2D frame by removing the elevation.

Limited Field of View

The field of view of a single radar is also limited and cannot detect 360 degrees. Multiple radars are used to achieve a full field of view to mitigate this limitation. The placement of radars must be carefully selected to avoid blind spots. O. Schumann et al. [26] place the four radars to be focused on the front of the car and have sufficient overlap between each radar.

Environmental Interference

Environmental factors like rain and fog can affect radar performance by attenuating signals at certain frequency ranges. By using a higher-frequency radar, attenuation effects can be minimized, but range is reduced. In urban areas surrounded by dense metallic structures and other vehicles operating with automotive radar in the same frequency range, multipath and false detection errors can occur. This phenomenon can lead to ghost objects being detected by the radar.

2.5 Point Cloud Filtering

Point cloud filtering is a crucial preprocessing step in radar-based positioning. It enhances the quality of raw radar data by removing noise, dynamic objects, and irrelevant detections. Automotive radar generates sparse and often noisy point clouds, which can degrade the performance of scan-to-map registration techniques. The following subsections discuss several radar point cloud filtering techniques.

2.5.1 Neural Networks

With advancements in AI, mmW-radars can now be integrated with machine learning to perform detection, tracking, and classification tasks. This was demonstrated by X.Cai et al. [27], who present an algorithm that can be applied to both static and dynamic objects at all ranges for classification. Using convolutional neural networks and artificial neural networks, the authors achieved >98% detection for pedestrians and vehicles and >94% for other objects, including stationary targets. The authors presented four classification algorithms: statistical RCS, distributed RCS, range-azimuth angle radar images, and 3D radar images. The performance of the classification algorithm shows great accuracy, which is extremely beneficial to the localization problem because the dynamic and static objects can be removed from the point cloud and only leave the landmarks for map matching.

D. Niederlohner et al. [28] proposed a self-supervised method to estimate the full Cartesian velocity of detected objects in automotive radar data. Their approach extends an object detection network to predict velocity without requiring labelled velocity measurements. Pre-training is conducted using single-frame-oriented bounding box (OBB) labels without velocity annotations. The network then leverages its own OBB predictions on unlabeled data to refine velocity estimation. Specifically, the predicted OBBs from an earlier frame are updated to a later timestamp using the network's estimated velocities, and the alignment error is used as a training signal for velocity learning.

J. Rock et al. [29] used a convolutional neural network (CNN) to reduce the interference an automotive radar may face during operation in a dense urban environment. Cross-sensor interference becomes inevitable when multiple radars in the same frequency range are used in proximity. The author used the CNN on the radar spectra data and obtained the Range and Doppler information of the target after each Discrete Fourier Transform. During the experimentation, they set up two data sets, one simulated and one collected in the real world; this allowed them to first train and fine-tune the CNN with controlled data and test the robustness with real-world datasets.

K. Patel et al. [30] proposed a deep CNN that identifies the region of interest in automotive radar spectra to classify objects in a scene. Unlike traditional point cloud-based methods, their approach directly processes the radar spectra rather than working with radar detections.

Like [29], the 2D-FFT is performed on the range and velocity spectrum, followed by a constant false alarm rate detector to identify targets. With the targets identified, the region of interest is passed to the CNN for classification. The CNN Architecture consists of three layers, each with 32, 64, and 128 filters, and after each filter, it is followed by an average pooling layer. The authors successfully classified using a prediction filter over a single four-second window during their experimentation.

S. Lu et al. [31], introduced a method called 4DRO-Net, which is designed for position estimation using deep learning of sparse radar data. During their experimentation and performance evaluation, the authors compared 4DRO-Net to classical positioning algorithms like ICP and NDT; they also compared the sparse radar results with Lidar-based methods and showed their algorithm had better performance. The 4DRO-Net first takes in the 4D radar data, which includes its position in X, Y, Z, and a velocity measurement. A custom feature extractor is then used on the point cloud before passing it to the pose generation module. The algorithm will then use the extracted point features to correct the pose from the feature-extracting module and ultimately perform pose refinement from course to fine. When the authors trained their neural network, they also considered the Radar Cross Section and the Doppler Velocity of radar targets.

2.5.2 Support Vector Machine (SVM)

SVMs are supervised machine learning models used for complex classification tasks. SVMs operate by separating classes using a hyperplane in multiple dimensions. Figure 2.4 shows the SVM classifier separating two classes by finding the optimal decision boundary, or hyperplane, that maximizes the margin between the closest points from each class. These critical points, which directly influence the position of the hyperplane, are known as support vectors. In more complex scenarios, SVM uses different functions to map the data into higher dimensional spaces, allowing the creation of a nonlinear decision boundary that effectively separates different classes.

SVM can be applied to multi-class classification problems using differ-

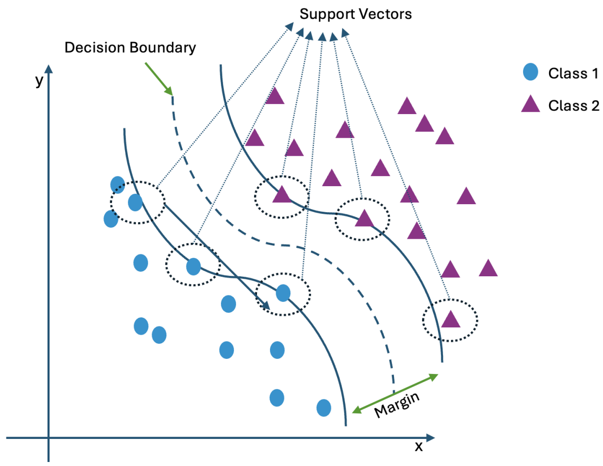


Figure 2.4: Illustration of SVM Decision Boundary

ent kernel functions, which can help optimize the decision boundary. Kernel functions such as Radial Basis Function, Polynomial, and Sigmoid handle nonlinear data by transforming the input spaces into higher dimensions, allowing the SVM to create more complex decision boundaries. When applying SVM to radar point cloud classification, parameters like Doppler velocity, range, azimuth, velocity of the ego vehicle, and RCS can aid in distinguishing points between different classes.

Zhao et al. [32] presents a method for classifying humans and vehicles using millimetre-wave radar. Zhao uses 11 training features derived from radar point clouds, focusing on shape, velocity, and echo intensity. Zhao conducted multiple tests using different SVM kernels, finding that the polynomial kernel achieved the best result. The results show that the proposed method outperforms traditional approaches, especially in complex scenarios with objects in different orientations and distances. Although Zhao's approach can effectively differentiate between pedestrians and vehicles, the data collected for the training and simulation is very controlled. The dataset detects radar signals from either a pedestrian or a vehicle directly in front of the radar without other interference.

Sasaki et al. [33] present a LiDAR-based system designed for nighttime sidewalk operations during snow removal, where poor visibility limits cameras' effectiveness. The authors use a SVM to classify pedestrians from other objects, incorporating noise removal and clustering to preprocess the LiDAR point cloud data. Although this is a LiDAR-based approach, it remains relevant to radar-based classification research because it demonstrates the effectiveness of SVM in handling noisy, cluttered environments. Similar challenges exist in automotive radar applications, where detecting and distinguishing pedestrians from other objects is crucial. The study highlights the adaptability of SVM for point cloud classification, making it a valuable reference for sensor-agnostic machine learning techniques in adverse conditions.

2.5.3 Density Based Clustering

Density Based Clustering, is a clustering technique that involves grouping points based on their spatial proximity. Clustering aims to identify and isolate distinct objects or features within a point cloud, making it easier to interpret the data and apply further analysis. Clustering a radar point cloud aims to separate meaningful objects, such as vehicles or static structures, from noise and irrelevant data points.

Zhang et al. [34] introduced Radar Elliptical Density-Based Spatial Clustering and Labeling (REDBSCAN), a variant of the DBScan algorithm designed to handle the unique challenges posed by millimetre-wave radar point clouds. The proposed method uses an elliptical instead of a traditional circle to cluster points together. REDBSCAN adjusts the clustering process based on the density and the shape of the radar returns, allowing for more accurate clustering and labelling in complex environments. However, the added complexity of REDBSCAN would require increased computational resources. It may be limited to post-processing only as it takes too many resources for real-time applications. Testing is also completed on proprietary datasets, and the authors did not show the performance of REDBSCAN on a public dataset.

Jingjie et al. [35] presented a clustering algorithm for traffic targets for radar point clouds. The proposed method combines DBScan with a Kalman Filter (KF) to refine the cluster results. The KF allows for the accumulation of predicted errors over multiple frames, which is used to improve the clustering of targets. Jingjie demonstrated that this approach enhanced the accuracy of correctly identifying traffic clusters in large and sparse datasets.

Xie et al. [36] explored the use of DBScan clustering over multiple frames to enhance the detection and tracking of targets. The developed algorithm maintains target consistency and reduces false detection due to noise and environmental factors. The authors demonstrated that their inter-frame algorithm improves the accuracy and target detection, particularly in scenarios with high noise levels.

Raj et al. [37] address the challenges of removing static clutter in automotive radar-generated point clouds prior to using the DBScan algorithm. Raj optimized the algorithm by removing static objects using a low-complexity static clutter removal method. This increases the algorithm's performance in grouping radar detection and classification. This approach is insightful as Raj et al. successfully removed static detections while clustering dynamic objects.

2.6 Sensor Fusion

Sensor fusion has become an essential component in autonomous navigation and vehicle positioning, enabling data integration from multiple sensors such as radar, Lidar, IMU, and GPS. By fusing complementary sensor modalities, positioning accuracy is significantly enhanced, particularly in GNSS-degraded environments [38, 39, 40].

For instance, Dawson et al. [38] demonstrated radar-based multi-sensor fusion, improving positioning reliability in covered parking garages. Similarly, Chen et al. [39] implemented Lidar-radar fusion, utilizing ICP-based registration to enhance environmental sensing and map-based positioning. These studies highlight the necessity of multi-sensor integration for precise vehicle positioning.

State estimation techniques such as Kalman filters play a fundamental role in sensor fusion. The Extended Kalman Filter (EKF) has been widely used for multi-sensor integration, providing real-time state estimation by linearizing system models [41, 40, 42]. However, in highly nonlinear environments, the Unscented Kalman Filter (UKF) has emerged as a more reliable alternative due to its ability to capture system dynamics without requiring linearization.

2.6.1 Extend Kalman Filter (EKF)

Traditional implementations often rely on the Extended Kalman Filter (EKF), which linearizes nonlinear system models for state estimation. In vehicle positioning, EKF has been extensively applied to integrate data from IMUs, GNSS, Lidar, and radar, compensating for the weaknesses of individual sensors. For example, Noureldin et al. [42, 40, 23] successfully applied an EKF to integrate radar and IMU data for vehicle positioning, mitigating long-term drift in GNSS-denied environments. One of the major drawbacks of the EKF is its reliance on the Jacobian matrix for linearization of the system dynamics and measurement models, which introduces approximation errors in highly nonlinear systems. To address this limitation, the Error-State Extended Kalman Filter (ES-EKF) has been developed as an alternative framework. Unlike traditional EKF, ES-EKF estimates small error perturbations around a nominal trajectory, which reduces the impact of linearization errors and improves state estimation accuracy, particularly in inertial navigation systems [41, 43]. Aravind et al. [41] demonstrated that ES-EKF provides better localization accuracy when integrating multiple sensors in GNSS-degraded environments. The ES-EKF formulation enables efficient error propagation and correction by estimating small deviations from the predicted trajectory rather than the full state vector. Marković et al. [43] further validated the effectiveness of ES-EKF for multi-sensor fusion, showing sub-meter accuracy in vehicle localization by integrating Lidar, IMU, and radar data.

2.6.2 Unscented Kalman Filter (UKF)

UKF is an alternative to the EKF that overcomes some of its limitations by avoiding linearization altogether. UKF employs a sigma point transformation, which more accurately captures the true mean and covariance of the state distribution in nonlinear systems [44, 45]. This makes the UKF particularly beneficial for applications where system dynamics are highly nonlinear and where sensor fusion involves complex state interactions. Several studies have demonstrated the superiority of UKF over EKF in sensor fusion applications. Ryu et al. [46] conducted a comparative analysis of EKF and UKF in GPS-INS sensor fusion and found that UKF consistently outperformed EKF in low-observability conditions. The UKF provided more accurate estimates when sensor updates were sparse or when rapid vehicle maneuvers introduced high system nonlinearity. Pan et al. [47] extended these findings by applying UKF in multi-sensor fusion for vehicle positioning, particularly in urban environments where GNSS signals are frequently blocked. Their results indicated that UKF provided more reliable state estimation, particularly in scenarios where the EKF struggled due to system nonlinearities and sensor noise. The sigma-point-based approach enabled UKF to handle better abrupt changes in vehicle motion, such as sharp turns or sudden acceleration.

2.6.3 Multi-Sensor Integration with Radar

Radar-based Simultaneous Localization and Mapping (SLAM) uses the vehicle's radar for positioning and map creation in adverse environments with no pre-existing map. Loop closure in radar SLAM is a crucial part of the algorithm that enables the system to recognize and correct any error that occurs over a revisited area and refine the vehicle trajectory and map over time.

J.Levinson et al. [48] proposed a map-based SLAM technique for high-accuracy localization of moving vehicles in urban areas, achieving positional accuracy of around 10 cm. Their approach integrates GPS, IMU, wheel odometry, and Lidar data to generate offline high-resolution environment maps. Although they do not use radar as their primary sensor, building and maintaining a point cloud map is similar to radar-based localization methods. The authors first defined a goal function to optimize. The algorithm is designed to be robust in dynamic environments by focusing on static objects like the road surface and disregarding moving objects. This allows the map to be reliable for overall environmental conditions. To perform localization, the authors use a particle filter that compares the real-time inferred measurement of the ground with the built map. This is then tightly coupled with the onboard IMU to determine the vehicle's position. Based on their experiments, the lateral error was around 10cm except when the vehicle turned, which had 30cm, demonstrating sub-50cm accuracy.

I.Belhajem et al. proposed a novel approach to enhance vehicle localization using low-cost sensors and machine learning [49]. The authors used SVM to increase the accuracy of the IMU for dead reckoning during a GNSS outage. This approach modifies and improves the traditional EKF for fusing the GNSS and IMU data. The EKF performance depends on the vehicle's dynamic variations and can change quickly based on environmental conditions. During the author's experimentation, using the SVM improved the positioning accuracy by 94%. When the GNSS signals are available, and the EKF is integrating its data with the IMU, an SVM is used to learn the dynamics and stochastic errors of the vehicle. When the GNSS signal is lost, the SVM uses the learnt characteristic to predict the errors based on the vehicle dynamics. The errors are used with the EKF to refine the position estimations.

Chen et al. [39] demonstrated a radar-Lidar fusion system, leveraging ICP-based registration for accurate environmental reconstruction. Beyond scan filtering improvements, sensor fusion techniques combining radar with IMU, Lidar, and GPS have significantly enhanced localization accuracy.

Overall, radar-based positioning is an evolving area of research for AV navigation. It aims to address the limitations of GNSS and INS in challenging urban environments. While advancements have been made in map registration and radar odometry, several challenges remain. The primary open research challenges include handling sparse returns, improving vertical resolution limitation, effectively managing environmental interferences, and sensor fusion with other exteroceptive sensors. Addressing these issues through advanced sensor fusion methods and machine learning techniques is essential for developing robust and accurate vehicle positioning solutions suitable for autonomous navigation applications.

Chapter 3

Methodology

This chapter details the methodology used to develop and evaluate radar-based positioning, focusing on scan filtering and sensor fusion methods for autonomous navigation. The proposed framework integrates radar point cloud filtering, map registration, and sensor fusion to improve positioning accuracy in GNSS-denied urban environments. The methodology is structured into key components, beginning with sensor-to-vehicle frame transformation, which ensures that raw radar detections are correctly aligned for subsequent processing. Next, radar point cloud filtering techniques are applied to eliminate dynamic objects, ghost detections, and noise, refining the data before scan-to-map registration. The filtered radar scans are then matched against a reference map using ICP, providing position updates. Finally, a sensor fusion framework is implemented, comparing UKF and ES-EKF for integrating radar-based positioning with onboard motion sensors.

3.1 Sensor-to-Vehicle Frame Transformation

Radar detections are recorded in the sensor frame, meaning the detected objects' positions are relative to each radar's coordinate system. To ensure an accurate point cloud representation of the surrounding 2D environment, these detections must be transformed into the vehicle's local frame. This transformation includes both rotation and translation. The transformation matrix for each radar is given by:

$$T = \begin{bmatrix} \cos\theta & -\sin\theta & x_s \\ \sin\theta & \cos\theta & y_s \\ 0 & 0 & 1 \end{bmatrix} \tag{3.1}$$

where θ is the mounting angle of the radar and x_s, y_s are the relative positions of each radar with respect to a chosen reference point on the vehicle.

Applying this transformation ensures that radar measurements are correctly mapped to the vehicle frame, enabling accurate sensor fusion and ensuring that the radar point cloud is correctly aligned for ICP-based map registration.

3.2 Radar Point Cloud Filtering

Both classical and machine learning-based methods are designed and evaluated for radar point cloud filtering. The goal is to eliminate dynamic objects, ghost detections, and static vehicles from the radar point cloud. Eliminating non-static environment radar returns would allow map-matching algorithms to minimize false matches and improve the positioning of the ego vehicle. Additionally, reducing the number of points may potentially lead to increased processing speed.

Two different classical filtering methods and an SVM classifier are proposed. Like RO, a velocity filter can remove dynamic objects from radar scans. The geometric filter, utilizing DBScan, is applied for clustering and removing ghost detections and noise, a common issue in radar point clouds. The SVM classifier performs the final classification of these points, distinguishing between vehicles and static environments.

3.2.1 Velocity Filtering

We implemented a velocity filter to differentiate between static and dynamic objects in the radar point cloud, similar to the approach in [6]. This filter utilizes the vehicle's odometry to estimate the radial velocity of detected objects relative to the radar sensor. The estimated radial velocity is computed using the vehicle's instantaneous forward velocity and the detected object's azimuth angle, derived from the radar measurements. By comparing this estimated radial velocity with the Doppler velocity detected directly by the radar sensor, points are classified based on their motion characteristics. This comparison helps identify objects whose motion aligns with or deviates from the vehicle's motion. Static points, such as those corresponding to buildings or other stationary objects, are isolated as they exhibit a consistent relationship between their estimated and detected radial velocities.

For a static object, the detected Doppler velocity and the estimated radial velocity depend on the direction of the vehicle's motion. For a forward-facing radar, if the vehicle is moving forward:

$$v_{\text{rad}} = -v_d. \tag{3.2}$$

If the vehicle is moving in reverse:

$$v_{\text{rad}} = v_d. \tag{3.3}$$

These relationships arise because the Doppler effect measures the relative motion between the radar and the object. The sign reversal accounts for the change in the vehicle's direction.

This relationship holds because the radar measures the velocity of an object's motion relative to its motion, Figure 3.1. A static object's Doppler return reflects the sensor's motion, while a dynamic object's Doppler return contains a combination of the sensor's and object's motion.

To account for the mounting angle of each radar sensor, we adjusted the detected azimuth angle (θ) to compute the corrected angle (α) using:

$$\alpha = \theta + \beta, \tag{3.4}$$

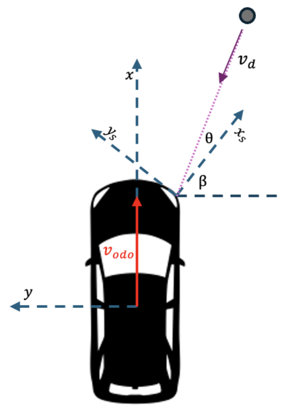


Figure 3.1: Illustration of vehicle forward speed vs detected object's Doppler measurement.

where β is the mounting angle specific to each radar sensor. Using this corrected angle, we estimated the radial velocity ($v_{\text{radial, est}}$) as:

$$v_{\text{rad,est}} = |v_{\text{odo}} * \cos(\alpha)|. \tag{3.5}$$

Next, we calculated the velocity difference (v_{diff}) by comparing the estimated radial velocity with the detected radial velocity (v_d):

$$v_{\text{diff}} = |v_{\text{rad,est}} - v_d|. \tag{3.6}$$

Points with v_{diff} below a predefined threshold were classified as static, while points exceeding the threshold were classified as dynamic.

This velocity filter forms the foundation for further processing by isolating relevant static points, contributing to an improved positioning solution.

3.2.2 Geometric Filtering

Geometric filtering was performed using a density-based clustering algorithm (DBScan). This process can remove radar ghost detections, which often appear as isolated or sparse points in the point cloud. Points that are closely packed together are grouped by DBScan, while isolated points are identified as noise. The algorithm has two parameters, ϵ and Minimum Points (MinPts). ϵ is the maximum distance between two points to be considered to be part of the same cluster. Used on radar parameters, this is the spatial proximity required for radar detections to be grouped together. MinPts is the minimum number of points required to form a cluster. If the number of points within the radius of ϵ meets or exceeds the MinPts threshold, that point is considered a core point, and a cluster is formed around it. The point is labelled as noise if it does not meet that threshold. A core point is essential for cluster growth, as it has enough neighbouring points within its radius to continue the clustering process. There are also secondary points that do not meet the qualification to be a core point but are reachable by the core point. Secondary points determine the boundary of the cluster. If a point is neither a core nor a secondary point, it will be classified as noise.

3.2.3 SVM Classification

The proposed method involves training an SVM classifier using annotated radar point clouds. Figure 3.2 shows the testing and training iterations of the proposed method. The SVM classifier is trained using features such as range, azimuth, Doppler velocity, RCS, and ego-vehicle velocity to classify detected objects into three classes. The multi-class SVM classifies the objects into vehicles, static environments, and others. However, the primary objective is to remove vehicles.

The SVM classifier can be tuned using several parameters, including C, Kernel Function, and Gamma. C controls the balance between a low misclassification error on the training data and minimizing the model's complexity. A smaller C value allows for larger margins of error between classes, allowing misclassifications in the training data. A larger C value reduces misclassification by tightening the margins and lowering the training error. However,

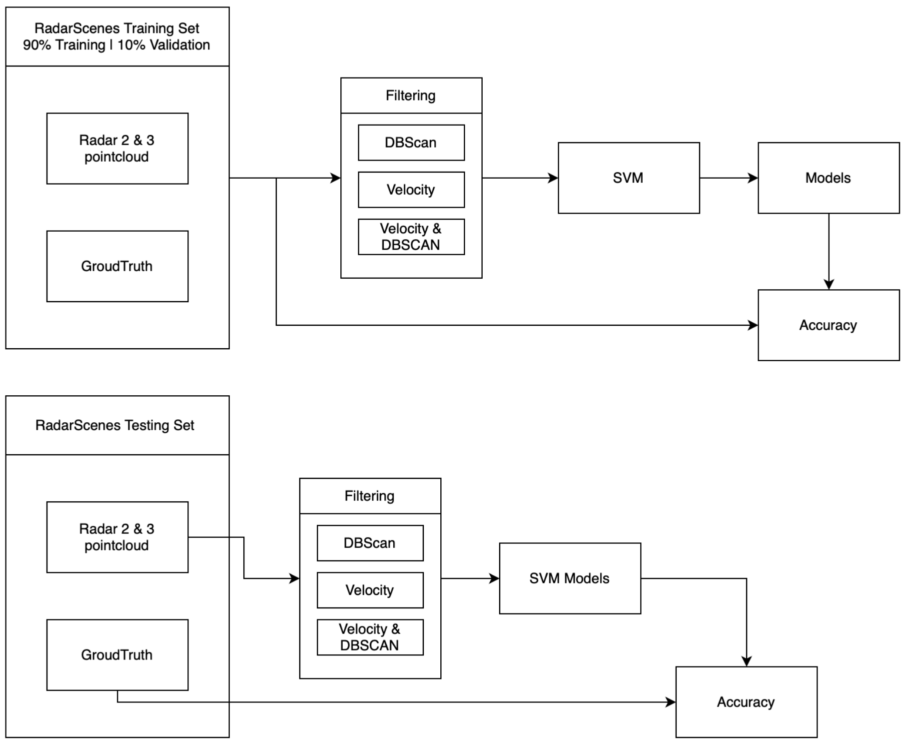


Figure 3.2: Support Vector Machine training and testing scenarios

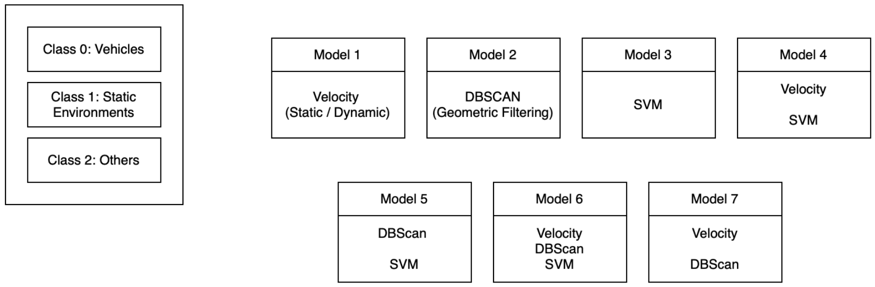


Figure 3.3: Different testing Scenarios with a combination of classical and machine-learning approaches

this comes at the cost of generalization, making the model less effective on unseen test data. This can lead to overfitting, where the model performs well on training data but struggles with using testing data. A high C value may lead to overfitting, as the model becomes excessively complex and captures noise rather than meaningful patterns [50]. Kernel functions transform the input data into higher dimensional space, allowing SVMs to handle non-linear datasets. Gamma is the last tuning coefficient, and it influences the decision boundary with the kernels [50]. Low gamma allows a smoother decision boundary. A high gamma is more sensitive to the individual data points, capturing more details. The disadvantage of high gamma is that the model becomes too sensitive to noise and anomalies in the training data. Like high C, it reduces the model's ability to generalize to the unseen data, resulting in poor performance on new test samples.

The different testing scenarios are detailed in Figure 3.3.

3.3 Position Estimation

To test the benefit of the developed radar point cloud filtering techniques to radar-based positioning, dead reckoning and map registration positioning pipelines are implemented with the point cloud filters as pre-processing steps.

3.3.1 Dead Reckoning

In these experiments, DR is employed to estimate the vehicle's trajectory, serving as a baseline for comparison against radar-based positioning methods. The computation of DR states is updated as follows:

$$States = \begin{bmatrix} x_k & y_k & \theta_k \end{bmatrix} \tag{3.7}$$

The time interval between the current and previous epoch is calculated:

$$\Delta t = t - t_{prev} \tag{3.8}$$

State update:

$$\begin{aligned} \theta_{k+1} &= \theta_k + w_z \Delta t \\ x_{k+1} &= x_k - v\sin(\theta)\Delta t, \\ y_{k+1} &= y_k + v\cos(\theta)\Delta t. \end{aligned} \tag{3.9}$$

The DR approach is efficient and operates independently without external sensors. However, without external corrections, the solution drifts and suffers from cumulative errors over time.

3.3.2 Map Registration

Radar-based positioning using scan-to-map registration relies on matching real-time radar scans with a previously generated reference map [52, 22]. For testing, we have two separate reference maps: one generated using OpenStreetMaps [53] and a Lidar scan map publicly generated by the city of Kingston. The reference maps will be discussed in the following section.

The processing of radar scan-to-map registration is detailed in Figure 3.4. Due to the sparsity of radar returns, multiple scans are aggregated to create a sufficiently dense point cloud for map registration. These scans are then compared against a pre-existing reference map using an ICP algorithm, which estimates the transformation needed to align the scan to the reference map [22]. This transformation provides a relative position estimate that can be fused with other sensor data for improved accuracy.

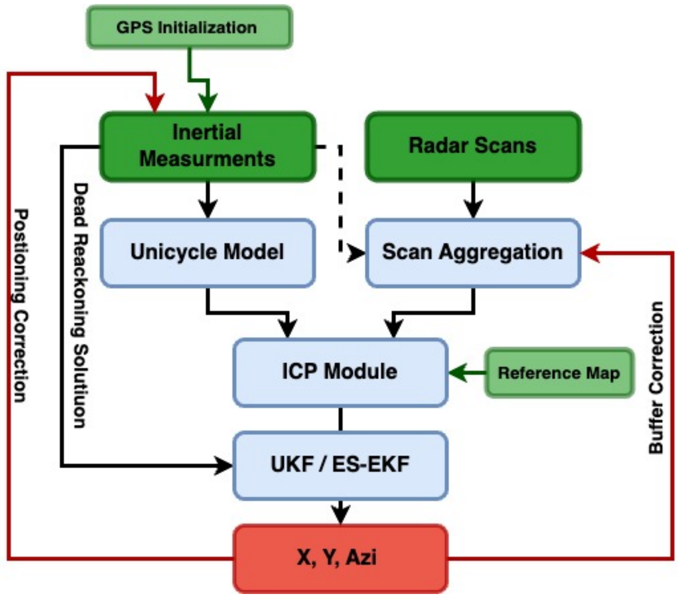


Figure 3.4: Positioning Pipeline [18, 51]

Figure 3.5 illustrates the filtering steps applied to radar point cloud data before map registration. The velocity filter removes stationary objects or detections outside the vehicle's expected movement range. The scan aggregation process collects multiple radar scans to enhance robustness. The geometric filter refines the point cloud by filtering out outliers before feeding the processed data into the ICP module, which aligns the radar scan with the reference map.

3.4 Sensor Fusion

A basic ES-EKF was implemented using a unicycle model. The best results were achieved using a radar buffer size of four seconds and an ICP update rate of two seconds. To ensure a fair comparison, identical model parameters were applied across all test scenarios for both the ES-EKF and the UKF. Specifically, the radar buffer size and ICP update rate were initially tuned for the ES-EKF and then transferred to the UKF without further modification. This approach ensures that any differences in performance are attributed solely to

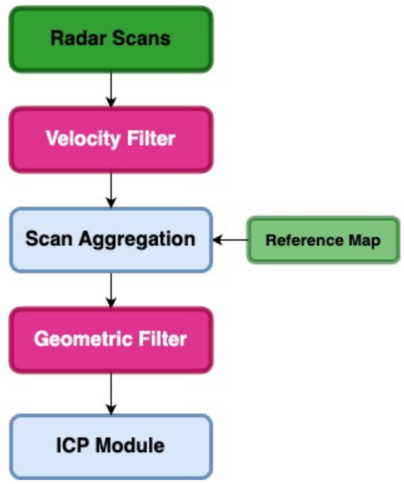


Figure 3.5: Radar Point Cloud Filtering to ICP [51]

the filtering methodologies rather than to differences in parameter optimization.

Tuning the UKF was challenging due to the unscented transform's sensitivity to sigma point selection. Poorly chosen sigma point weights could negatively impact the filter's stability and accuracy. To ensure the UKF functioned correctly, the predicted state trajectory needed to closely match the dead reckoning solution before any external corrections were applied. This was achieved by carefully adjusting the sigma point distribution weights, the state covariance matrix, and the process noise covariance matrix. The specific values for these matrices are detailed in the following sections.

To evaluate the performance of ES-EKF and UKF in urban segments, both filters were executed using the same dataset, ensuring identical start and end indexes for a fair comparison. The filters were applied separately to analyze their performance before assessing their combined impact on sensor fusion accuracy. Additionally, we evaluated the filters with fewer ICP iterations to examine the trade-off between positioning accuracy and computational efficiency.

3.4.1 UKF

UKF for these experiments is set up to estimate the vehicle's position, heading, and velocity by integrating the odometer, ω_z sensor from the IMU, and radar point cloud based ICP measurements.

The UKF state vector is defined as:

$$X = \begin{bmatrix} x & y & \theta & v & b_{\omega_z} \end{bmatrix} \tag{3.10}$$

where:

- x, y: represent the vehicle's position,
- θ: is the heading angle or azimuth,
- v: is the velocity from the odometer,
- b_{ω_z}: gyroscope bias for heading rate correction.

The state transition function propagates the vehicle state forward in time using IMU and odometer measurements:

$$\begin{aligned}
&\theta_{k+1} = \theta_k + (w_z + b_{wz})\Delta t \quad \text{(Heading-position update)} \\
&x_{k+1} = x_k - v\sin(\theta)\Delta t \quad \text{(X-position update)} \\
&y_{k+1} = y_k + v\cos(\theta)\Delta t \quad \text{(Y-position update)} \\
&v_{k+1} = v_k \quad \text{(Velocity update)}
\end{aligned} \tag{3.11}$$

The UKF measurement model incorporates radar-based ICP positioning measurements, which provide position and heading corrections:

$$\mathbf{z} = h(\mathbf{x}) = \begin{bmatrix} x & y & \theta \end{bmatrix} \tag{3.12}$$

The UKF uses the following covariance matrixes to propagate noise, errors, and sigma points. State covariance matrix P:

$$P_U = \begin{bmatrix} 0.001^2 & 0 & 0 & 0 & 0 \\ 0 & 0.001^2 & 0 & 0 & 0 \\ 0 & 0 & (0.01^\circ)^2 & 0 & 0 \\ 0 & 0 & 0 & 0.1^2 & 0 \\ 0 & 0 & 0 & 0 & 0.005^2 \end{bmatrix}$$

Process noise covariance matrix Q:

$$Q_U = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & (0.1^\circ)^2 & 0 & 0 \\ 0 & 0 & 0 & 5.8e^{-5^2} & 0 \\ 0 & 0 & 0 & 0 & 2.9e^{-9^2} \end{bmatrix}$$

Measurement noise covariance matrix R:

$$R_U = \begin{bmatrix} 0.2^2 & 0 & 0 \\ 0 & 0.2^2 & 0 \\ 0 & 0 & (2^\circ)^2 \end{bmatrix}$$

3.4.2 ES-EKF

The ES-EKF used in this research is a basic Extended Kalman Filter implementation, relying on a unicycle model to propagate the system states. The error states are:

$$\delta X = \begin{bmatrix} \delta x & \delta y & \delta\theta \end{bmatrix} \tag{3.13}$$

The ES-EKF also uses the same state transition functions as the UKF. However, it differs in the measurement model on how the measurement model is applied:

$$\begin{aligned} z = h(x) &= \begin{bmatrix} x & y & \theta \end{bmatrix} \\ \delta X &= K(z - x) \\ X_{k+1} &= X + \delta X \end{aligned} \tag{3.14}$$

where:

- K: Kalman Gain
- X: is the nominal state, containing the position and heading

The ES-EKF propagates noise, errors, and gains using the following covariance matrixes: State covariance matrix P:

$$P_E = \begin{bmatrix} 0.1^2 & 0 & 0 \\ 0 & 0.1^2 & 0 \\ 0 & 0 & 0.1^2 \end{bmatrix}$$

Process noise covariance matrix Q:

$$Q_E = \begin{bmatrix} 0.001 & 0 & 0 \\ 0 & 0.001 & 0 \\ 0 & 0 & 0.001 \end{bmatrix}$$

Measurement noise covariance matrix R:

$$R_E = \begin{bmatrix} 0.01 & 0 & 0 \\ 0 & 0.01 & 0 \\ 0 & 0 & 0.01 \end{bmatrix}$$

3.5 Reference Map

A prior map of the environment is essential for evaluating radar-based positioning within a scan-to-map registration framework. Accurate reference maps provide a static representation of the surroundings, allowing radar scans to be aligned with known landmarks using ICP algorithms. This study utilizes multiple reference maps, including publicly available maps such as OpenStreetMap, high-resolution Lidar maps, and custom radar-generated maps for scenarios where no pre-built maps exist. These maps serve as the baseline for positioning evaluation, ensuring that the radar scan alignment process is robust across different urban environments.

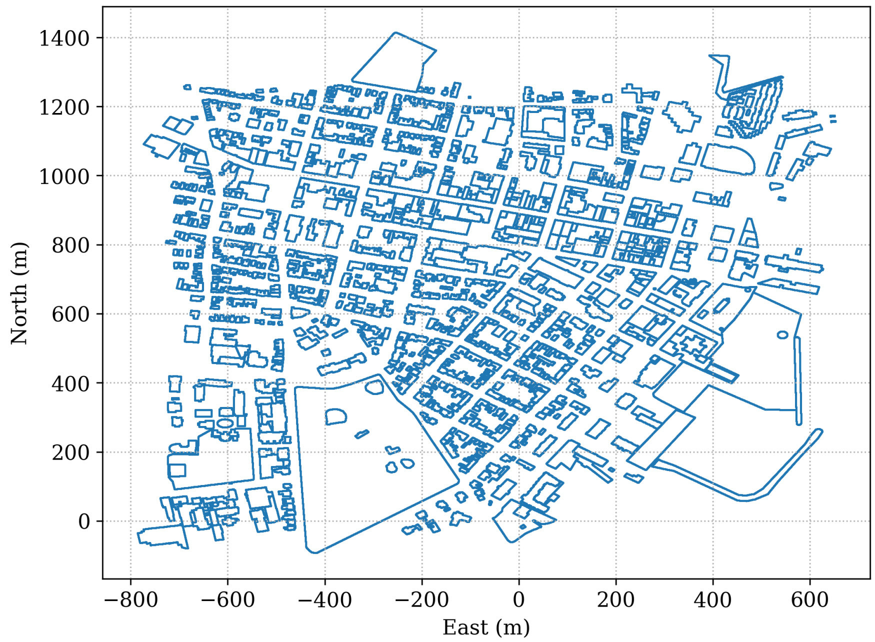


Figure 3.6: Kingston Downtown with OSM [53]

3.5.1 Public Reference Maps

The reference maps utilized in this study were sourced from two open datasets: OSM Figure 3.6 and Kingston Lidar Maps Figure 3.7. These publicly available datasets serve as a baseline for radar scan positioning and provide valuable structural details for ICP-based positioning. The choice of which reference map to use for each segment depends on the environmental characteristics, which will be further analyzed in the results and discussion section.

OSM [53] is a widely used vector-based open-source mapping service that provides geographic information on road networks, building footprints, and static infrastructure. The OSM dataset used in this study was extracted for the Kingston downtown region, where the street network and building outlines serve as static landmarks for positioning.

The Kingston Lidar map provides high-resolution 3D point cloud data, which offers precise structural details of the environment. Lidar maps cap-

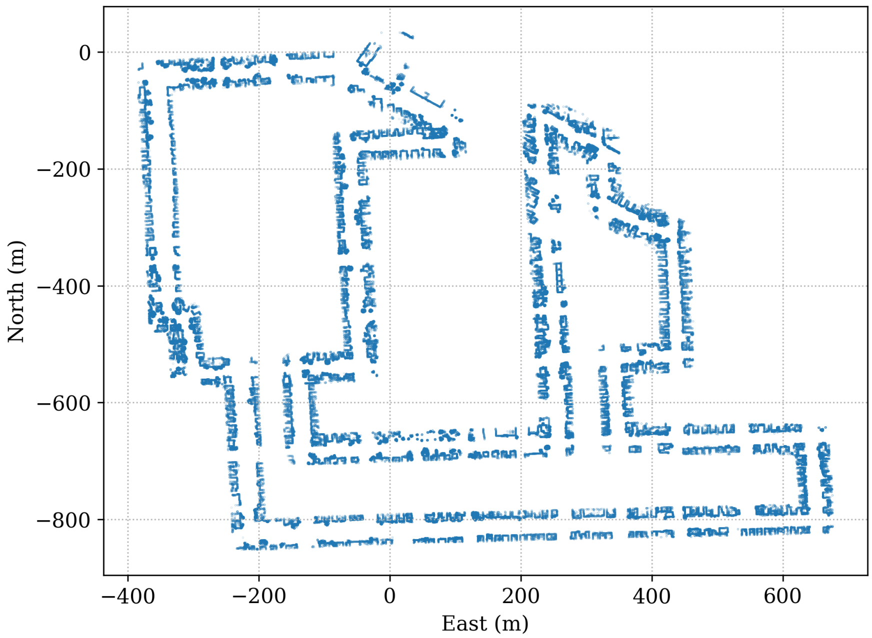


Figure 3.7: Kingston Suburb with Lidar map

ture fine-grained surface features, including elevation, building facades, and obstacles. To ensure the Lidar reference map was suitable for radar-based positioning, we performed preprocessing using CloudCompare [54] to remove all detected vehicles, pedestrians and erroneous detections determined to be noise. The resulting point cloud represented static environment features only.

3.5.2 Radar-based Reference Map

A limitation of some open-source datasets aimed towards research in positioning and perception is the lack of maps and trajectory location data for privacy reasons. This limitation prevents the direct use of open-source maps for testing, because the actual location is not known. To address this limitation and still benefit from the publicly available dataset, its possible to create a custom reference map using the radar point cloud data from the dataset and aligning it using a ground truth solution. However, this approach introduces a challenge: the same radar points used to build the reference map are also part of the testing process. This overlap could potentially bias the results, which would pose a risk to the testing data, resulting in an inflated performance metric. To minimize the impact of this limitation, we implemented several strategies to ensure the integrity of the evaluation process. The generation of a radar-based reference map is described in the Experimental setup section in the following chapter.

3.5.3 Radar Point Cloud Preprocessing Tuning Parameters

Each point cloud filtering method contains several tuneable parameters, which can be adjusted to improve the performance of the algorithm. The implementation details for each filtering method are described in the following subsections.

Velocity Filter

The threshold for classifying objects as static or dynamic was set to 0.29 m/s, which corresponds to the expected accuracy of the radar Doppler velocity measurements as given by the sensor datasheet. This value was selected to ensure that minimal sensor noise would incorrectly classify static objects as dynamic. Additionally, through visual inspection, this threshold proved to be strict enough to effectively filter out moving objects while retaining sufficient static detections for accurate map matching.

Geometric Filter

The primary setting for the geometric filter is 0.5m for ϵ and 5 for $MinPts$. These values were chosen to account for the sparsity of radar point clouds after the initial filtering stage. A more restrictive setting could risk removing too many static points, which are crucial for improving the accuracy of the ICP process. An additional validation step was implemented to prevent excessive filtering. If the geometric filter removed more than 30% of the points, the filtered result was discarded, and the original point cloud was used instead. This safeguard ensured that the filtering process did not overly reduce the number of useful detections.

SVM Classification

Five features—range, azimuth, radar cross-section, measured Doppler, and vehicle speed—were sent to the SVM for training. The SVM was trained to classify three classes: vehicles, static environments, and others. From the RadarScenes annotated data, vehicles (0) include passenger cars, large vehicles, trucks, buses, and trains. Static environments (1) retain their original labels, and others (2) contain bicycles, pedestrians, animals, and other dynamic objects encountered while driving.

Chapter 4

Results and Discussion

This chapter discusses the results and analysis of the proposed radar point cloud filtering methods within a radar-based positioning framework. It assesses the effectiveness of radar point cloud filtering, scan-to-map registration, and sensor fusion in enhancing positioning accuracy in urban and GNSS-denied environments.

This chapter is broken down into the following sections:

1. **Public Datasets**: Describes the datasets used for evaluation, including RadarScenes and NavINST.
2. **Experimental Setup**: Details the selected test sequences, reference map construction, and preprocessing steps applied to radar point clouds.
3. **Evaluation Metrics**: Defines the error metrics used to assess the performance of different filtering and positioning techniques.
4. **Radar Processing Analysis**: Examines the impact of velocity filtering, geometric filtering, and SVM classification on improving radar data quality.
5. **Trajectory Estimation**: Compares estimated vehicle trajectories against ground truth data, highlighting improvements from the proposed filter-

ing methods. Analyzes the performance of UKF and ES-EKF sensor fusion frameworks, evaluating their accuracy and robustness.

6. **Real-Time Performance Analysis**: Investigates the computational efficiency of the proposed approach and its feasibility for real-time implementation.

4.1 Public Datasets

The performance of the proposed radar-based positioning framework was evaluated using publicly available datasets. Real-world radar sensor measurements from these datasets were analyzed to investigate various driving conditions, thereby facilitating a comprehensive assessment of point cloud filtering, scan-to-map registration, and sensor fusion techniques. The datasets selected for this study include RadarScenes, which was developed with an emphasis on radar perception and object detection, and the NavINST dataset, which offers multi-sensor data for high-precision positioning.

4.1.1 RadarScenes

The RadarScenes dataset is a large-scale dataset designed for automotive radar perception and positioning [55]. It includes data from four automotive radar sensors, enabling research in ego-motion estimation, object detection, and radar odometry. Key features of RadarScenes include:

- **158 driving sequences**: in urban, suburban, and highway environments;
- **Automotive-grade radar sensors**: capturing range, azimuth, elevation, and Doppler velocity;
- **IMU sensor data**: included for ego-motion estimation and odometry correction;

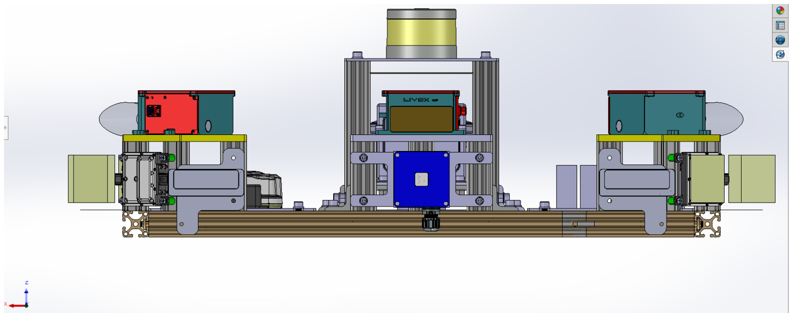


Figure 4.1: Front View of NavINST Sensor Rack [56]

- **Radar-based object detection annotations**: point-wise labelling for training and evaluating perception models.

Maps of the environment are not available for the RadarScenes dataset, and nor is its geographic location known.

4.1.2 NavINST

The second dataset used for testing was provided by the Navigation and Instrumentation (NavInst) Laboratory at the Royal Military College of Canada [56]. The test vehicle was equipped with a comprehensive suite of sensors, including Lidar, radar, cameras, IMUs, and GNSS receivers, as shown in Figure 4.1, enabling multi-modal data fusion for high-precision positioning and mapping. The NavINST dataset contains a diverse set of sensors, each contributing to different aspects of vehicle positioning and navigation:

- **Camera**: includes both monocular and stereo cameras, which provide visual perception for autonomous navigation
- **Lidar** : solid-state and mechanical Lidar sensors, generating high-resolution 3D point clouds for mapping and positioning
- **Radar**: dataset includes four electronically scanning radars (ESRs), offering 360-degree coverage. These radars provide Doppler velocity measurements, range estimation, and object detection, enabling radar-based positioning in GNSS-denied environments.

- **IMU**: dataset incorporates multiple IMUs, including both tactical-grade and commercial-grade sensors, allowing for inertial navigation and sensor comparison.
- **GNSS**: provides high-precision GNSS data with RTK corrections, allowing for centimeter-level ground truth positioning. GNSS data is crucial for benchmarking sensor fusion algorithms, validating positioning performance, and providing a reference for map-based positioning systems.

Additionally, the NavINST dataset includes trajectories for both indoor and outdoor environments, recorded under various lighting conditions. These trajectories span urban streets, parking garages, and complex GNSS-challenged environments.

The geographic locations of the NavINST trajectories are known, enabling the use of publically available maps of the corresponding regions.

4.2 Experimental Setup

The experimental setup defines the testing framework used to evaluate radar-based positioning performance. This includes selecting test segments, preprocessing radar data, and constructing reference maps for scan-to-map registration. The proposed filtering techniques and sensor fusion algorithms are rigorously evaluated in diverse urban environments.

4.2.1 RadarScenes Setup

Sequences 112–117, 125, 131, 138, 139, and 142–145 were selected from the RadarScenes dataset for trajectory testing. These sequences were selected based on manual inspection of corresponding camera images, confirming that the trajectory is within urban environments. This setting provided a challenging and realistic test environment for evaluating our radar-based positioning and filtering methodologies.

Because the RadarScenes dataset is in an unknown geographic lcoation, a reference map had to be constructed to test the radar to amp registration pipeline. The map creation is described in the following subsection.

RadarScenes Map Creation

The reference map was built exclusively using radar detections labelled as the static environment (label ID = 11), stitched together using the ground truth solution. To reduce the bias effect of using the test data in reference map construction, we also implemented the following measures:

1. **Downsampling**: The static landmarks were randomly downsampled by 80%, significantly reducing the overlap between the reference map and radar data used for registration.
2. **Geometric Filtering**: The remaining reference points were processed through an additional geometric filter to remove noise and clusters. This step ensures that only points corresponding to significant static landmarks, such as buildings, are preserved. Noise points, along with smaller clusters, were filtered out as they are unlikely to represent meaningful landmarks. By reducing the number of overall points, the resulting dataset is cleaner and more representative of the static environment.
3. **Gaussian Noise Injection**: Random Gaussian noise with a standard deviation of 15 cm was added to the x and y coordinates of the remaining reference points. This value was chosen based on the radar sensor's range error. This noise was introduced to add uncertainty to the reference map, preventing the ICP algorithm from achieving perfect overlap between radar data and the reference map. Ensuring the points in the reference map do not align exactly helps to reduce the bias in the testing process and provides a more realistic representation of real-world segments.

Figure 4.2 illustrates two reference maps constructed using the aforementioned approach. The reference maps demonstrate the effectiveness of filtering and isolating meaningful static landmarks. By removing small clusters and noise, the geometric filter retains the most prominent static structures, which are then used to construct the reference map. This map will be evaluated for its reliability during the ICP alignment process. Although some smaller clusters remain along the outer edges of the maps, limiting the radar point cloud range minimizes their impact on the registration process.

To focus on relevant points for positioning, radar detections from all four sensors within a range of 30 meters were included. The resulting point cloud underwent two stages of filtering to improve the data quality: velocity filtering and geometric filtering. These filters effectively isolate static objects and eliminate noise, improving radar data quality for accurate map registration.

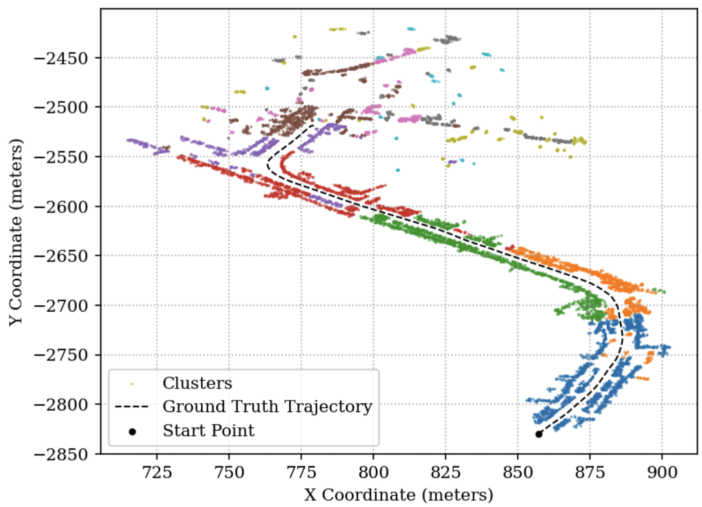
Filtered radar data was then aligned with the pre-constructed reference map using the ICP algorithm. To achieve accurate alignment, ICP employed a search window with a 50-meter radius centred around the vehicle's estimated position. This was chosen to ensure the algorithm focuses on the relevant sections of the point cloud and prevents any over-corrections due to irrelevant points. By constraining the search to a localized area, ICP could identify corresponding points between the radar and reference map more effectively, even in dense urban environments. The radar point cloud was used to iteratively refine the vehicle's pose, improving its position and orientation estimates. Finally, we compared the estimated trajectory to the ground truth to determine the proposed filtering methods' performance in enhancing the quality of the radar point cloud for ICP-based positioning.

4.2.2 NavINST Setup

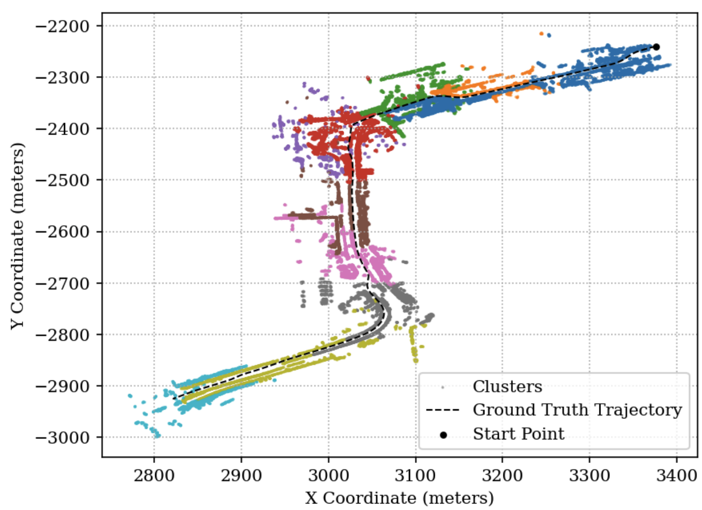
Sensor ID	X(mm)	Y(mm)	Mounting Angle(degree)
Radar 1: Front Left	-413.8	308.3	306.7
Radar 2: Front Right	413.8	308.3	53.3
Radar 3: Rear Left	413.7	-328.7	216.7
Radar 4: Rear Right	-413.7	-328.7	126.7

Table 4.1: Radar Mounting Parameters for NavINST sensor rack [56]

The vehicle was fitted with four electronically scanning radars (UMRR-



(a) RadarScenes Sequence 139 Reference Map.



(b) RadarScenes Sequence 113 Reference Map.

Figure 4.2: Reference map built using downsampled, clustered, and Gaussian noise-injected static points.

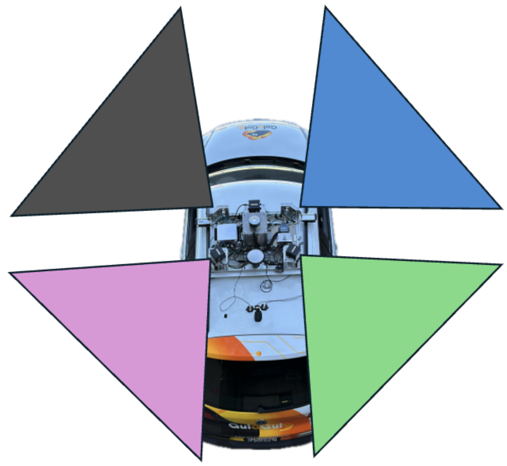


Figure 4.3: Illustration of the radar sensors' field of view, demonstrating the 360-degree coverage provided by the four electronically scanning radars mounted on the test vehicle.

96 Type 153) positioned at each corner to provide 360-degree coverage, as shown in Figure 4.3, mounting parameters in Table 4.1. The radars operate at 20 Hz in short-range mode, with a maximum detection range of 20 meters. Additionally, the system includes an odometer sensor operating at 3 Hz, which is synchronized with the radar data to maintain consistency in motion estimation. These sensors provide the necessary data for radar-based positioning in urban environments. The collected radar scans are processed to filter dynamic objects and noise, ensuring that only meaningful static features are retained for positioning.

NavINST Reference Maps

The reference maps used in NavINST testing were gathered from two open-source datasets: OpenStreetMap (Figure 3.6) and Kingston Lidar Maps (Figure 3.7). These maps serve as the baseline for estimating radar scan positioning. The choice of map for each segment is highly dependent on the static

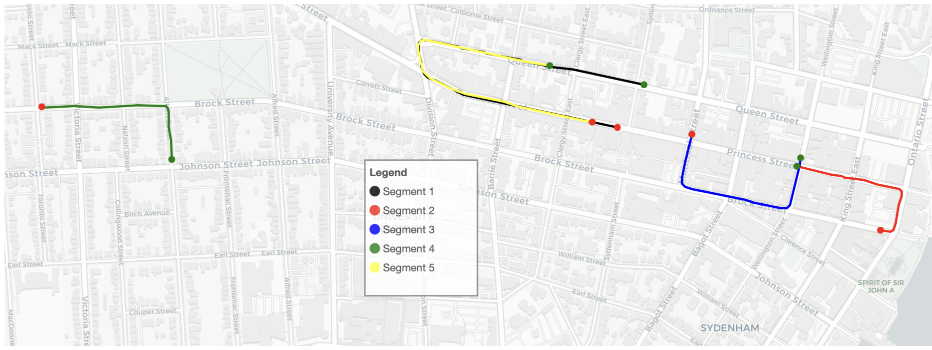


Figure 4.4: Five urban segments located in the City of Kingston, ON [53]

environment, which will be further analyzed in the results and discussion section.

Figure 3.6 are used for segment 1/2/3/5, and Figure 3.6 are used for segment 4.

NavINST Test Segments

Five urban driving segments are chosen from the NavINST dataset on which to test the point cloud filtering and positioning pipeline. The segments are shown in Figure 4.4. The statistics for each segment are detailed in Table 4.2.

Table 4.2: Trajectory Statistics

Driving Segments	**1**	**2**	**3**	**4**	**5**
Distance Travelled (m)	1040	361	473	401	769
Duration (Secs)	270	72	180	90	180
Average Speed (km/h)	13.86	18.07	8.50	16.04	15.44

To evaluate the performance of ES-EKF and UKF in urban segments, both filters were executed using the same dataset, ensuring identical start and end indices for a fair comparison. The radar point cloud filters were applied for point cloud preprocessing. After which the two sensor fusion were independently applied and evaluated for positioning accuracy. Additionally, we evaluated the filters with fewer ICP iterations to examine the trade-off between positioning accuracy and computational efficiency. Reducing the number of iterations lowers overall processing time, making real-time implementation more feasible while potentially affecting positioning accuracy.

Segments 1 & 5

Due to their distinct traffic and environmental characteristics, Queen Street and Princess Street were selected as test segments within downtown Kingston. Queen Street is a two-lane, bidirectional road, whereas Princess Street is a one-way street with two lanes. Both streets feature several sections with on-street parking. Both segments pass through areas with significant pedestrian activity, parked vehicles, and urban obstructions. The primary difference between these segments is the time of data collection. Segment 1 was conducted in the early morning, when traffic congestion was low, while Segment 5 was conducted in the evening when traffic and pedestrian activity were higher. By comparing these two segments, we can evaluate the performance and robustness of radar-IMU fusion across varying lighting and environmental conditions.

Segment 2

Segment 2 continues along the remainder of Princess Street before turning onto a bidirectional four-lane road. At the end of this segment, it passes through a park and a large open parking structure with minimal static detections, making radar-based positioning more challenging. This section provides an opportunity to evaluate how the radar-IMU fusion performs in low-feature environments where fewer landmarks are available for positioning.

Segment 3

Segment 3 covers another section of downtown Kingston, where a significant portion of the trajectory is spent on a two-lane, one-way street in front of a hospital. This area experiences high traffic volumes, frequent intersection stops, and pedestrian activity, making it a challenging environment for sensor fusion.

Segment 4

Segment 4 is located in a residential neighbourhood with wide, two-lane streets. The primary challenge in this segment is the abundance of parked vehicles, dense tree coverage, and enclosed yards with fences, which can obstruct radar detections and limit the number of reliable static features for positioning.

4.3 Evaluation Metrics

To evaluate the proposed filtering methods' performance on radar-based ICP positioning estimation accuracy, we analyzed the results in terms of filtering effectiveness, trajectory alignment accuracy, and error analysis. The metrics for each trajectory were selected to quantify the alignment between the estimated vehicle trajectory and the ground truth, providing insights into the overall accuracy and consistency of the positioning estimation.

The first metric is the Maximum Error (Max Error), which represents the most significant Euclidean distance between any point on the estimated trajectory and the corresponding point on the ground-truth trajectory. This metric highlights the worst-case segment and is critical for understanding the maximum deviation that may occur.

The Root Mean Square Error (RMSE) is calculated as the square root of the mean of the squared Euclidean distances between the estimated trajectory points and the ground truth points. It provides an overall measure of positioning accuracy by emphasizing the larger errors more significantly than the smaller ones, offering insight into the system's performance. The Mean Square Error (MSE), which is the mean of the squared Euclidean distances between the estimated and ground truth, is helpful in analyzing the variance in positioning errors and assessing consistency. While RMSE provides a comparable scale of errors, MSE retains the squared error values, making it more sensitive to larger deviations.

We also evaluate the percentage of trajectory points within specific error thresholds to assess the algorithm's performance at different levels of precision. These thresholds are within three meters and one meter.

By analyzing these metrics collectively, we can quantify each filtering method's improvements to the radar-based positioning system. This comprehensive evaluation allows for a detailed assessment of both the effectiveness of the filtering approaches and the robustness of the positioning algorithm under various urban segments.

4.4 Radar Point Cloud Filter Analysis

Radar data requires preprocessing to enhance its reliability for map registration and sensor fusion. This section evaluates the impact of the point cloud filtering techniques presented in this work on improving radar data quality when applied to radar-based positioning methods. The goal is to refine the radar point cloud by removing dynamic objects, reducing noise, and isolating meaningful static landmarks, ensuring more accurate and robust positioning. The effectiveness of these filtering methods is analyzed based on trajectory alignment accuracy and the improvements achieved in overall positioning. The following subsections describe the performance of the velocity filtering, geometric filtering, and SVM classification methods.

4.4.1 Velocity Filter

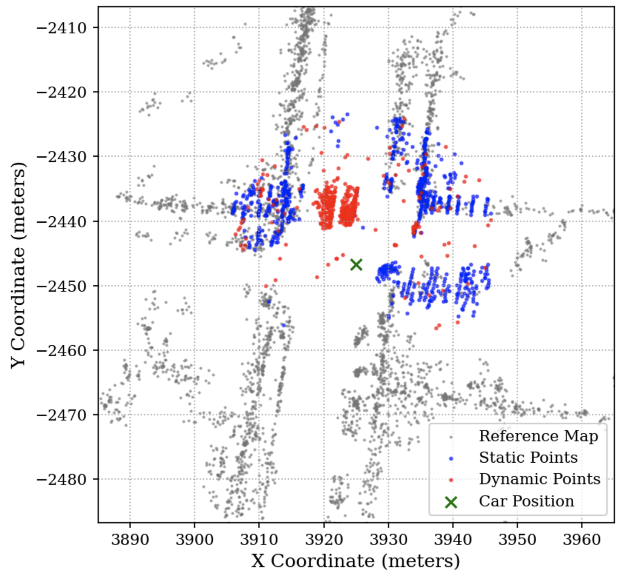
As demonstrated in Figure 4.5a, as the vehicles are passing through a major intersection, the proposed method could identify two incoming vehicles based on their detected Doppler velocity as well as small moving objects at the street corners, most likely corresponding to pedestrians or cyclists in the area. In Figure 4.5b, the vehicle is stopped at a traffic light, observing the crossing vehicle. Interestingly, when the vehicle is stationary, as seen in Figure 4.5b, the number of total detected points decreases significantly compared to when the vehicle is moving in Figure 4.5a. This reduction occurs because radar systems rely heavily on relative motion and Doppler shifts to detect objects effectively. When the vehicle is stationary, the radar struggles to generate sufficient returns from objects in the environment, resulting in fewer detected points. This limitation directly impacts the system's overall performance. With fewer points available, the lack of sufficient data for ICP alignment can lead to errors in trajectory estimation and pose refinement. This highlights a key challenge in radar-based systems, where the effectiveness of detection and alignment is influenced by the vehicle's motion.

These two figures demonstrate the velocity filter's effectiveness in separating static points from dynamic objects by applying the proposed threshold. Computational efficiency is demonstrated by the velocity filter, achieving an average processing time of 1.2 ms per radar frame, compatible with real-time processing.

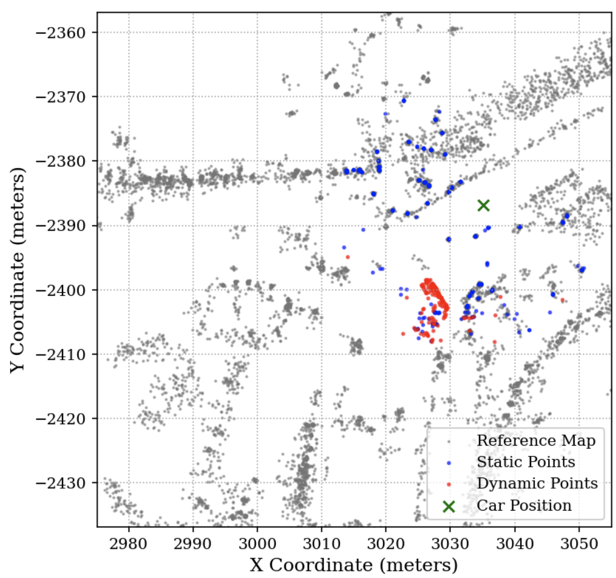
4.4.2 Geometric Filter

Figure 4.6 demonstrates the results of the geometric filter applied to the static points identified by the velocity filter. The red points represent dynamic objects that were filtered out by the velocity filter, while the black points correspond to noise or small clusters with fewer than 10 points, rejected by the geometric filter. The coloured clusters represent the meaningful static landmarks retained by the geometric filter, which strongly correlate with the static environment when compared to the preprocessed map.

Visual inspection was utilized to carefully tune the geometric filter, balancing sensitivity and accuracy. Overly strict filter parameters risk removing meaningful points from the point cloud, whereas overly loose constraints allow noise to persist, introducing errors in the registration process. Despite this fine-tuning, the geometric filter remains computationally efficient, with an average processing time of 2.1 ms per radar frame, ensuring real-time processing capabilities for autonomous applications.

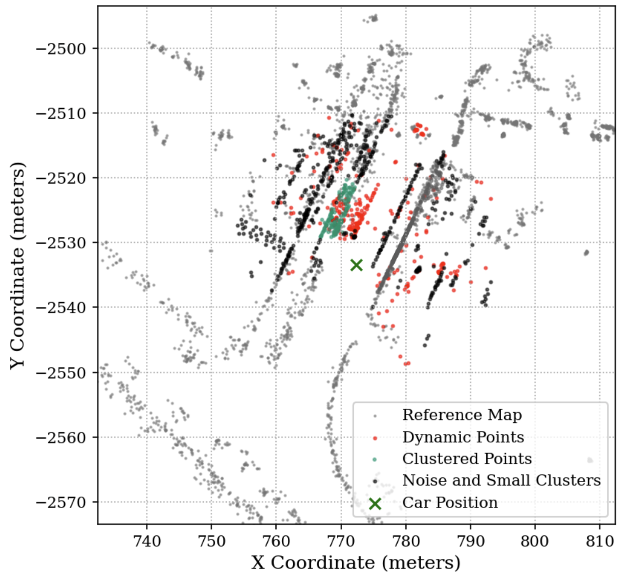


(a) Scene within Sequence 145.

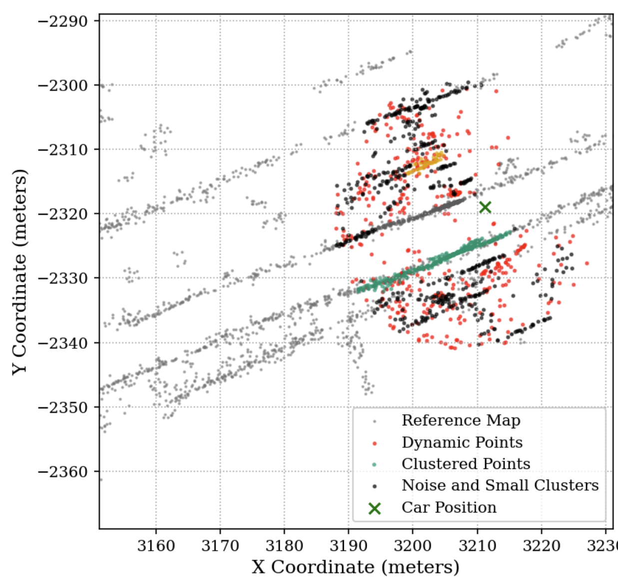


(b) Scene within Sequence 113.

Figure 4.5: Velocity filter with a threshold of 0.3 m/s applied.



(a) Scene within Sequence 145.



(b) Scene within Sequence 113.

Figure 4.6: Geometric filter applied static points returned from the velocity filter, dynamic detection and noise are also plotted for visualization.

4.4.3 SVM Classification

As shown in table 4.3, the model achieved an accuracy of 80% during the training phase, with precision, recall, and F1-scores above 0.75 for the two major classes. The meaning of the performance metrics is described below:

- Precision: ratio of correctly predicted positive observations to the total predicted positive observations
- Recall: ratio of correctly predicted positive observations to all observations in the actual class
- F1 Score: combination of Precision and Recall

Training time was extensive, highlighting the computational intensity and the complexity involved in predicting vehicles, static environments, and other classes. The model's ability to distinguish between vehicles, static environments, and other objects was validated through its performance metrics. Balancing the data points for each class was a crucial step during the training process, as SVM models struggle with the classification of unbalanced datasets. An unbalanced dataset can lead to a bias in the model towards the larger class. In this case, class 1 was much larger than class 0, and both were much larger than class 2. To balance the dataset, the total number of detections from class 0 was used to randomly sample class 1 to ensure equal representation between the two classes. Class 2, being much smaller, was left unchanged.

RadarScenes sequences 1 through 126 and 146 through 153 were used for training, with 90% of the detected points used for training and 10% for testing. Sequences 141 to 145 were allocated for validation only and were not seen by the model during training. These four sequences provide a challenging environment in which to test the algorithm's performance in urban areas. Sequences 127 through 145 were not used due to computational limitations.

Table 4.4 shows the results of this testing phase. Recall that class 1 is defined as static environments, while class 0 corresponds to vehicles. Observing the F1-Score for both class 0 and 1 achieved an accuracy of 70% and 71%, respectively, indicating that the SVM performs well, but additional improvements can be made.

The testing was completed with balanced data between the two major classes to ensure that the SVM model's performance could be accurately gauged. By maintaining a balanced dataset during testing, the model's true capabilities in distinguishing between vehicles and static environments in complex urban settings can be evaluated.

Table 4.3: Classification Report for SVM Training

Class	Precision	Recall	F1-score	Support
0	0.85	0.76	0.80	515,953
1	0.77	0.83	0.80	515,230
2	0.78	0.81	0.80	410,595
Accuracy		0.80		1,441,778
Macro Avg	0.80	0.80	0.80	1,441,778
Weighted Avg	0.80	0.80	0.80	1,441,778

Table 4.4: Classification Report for SVM Testing on a Balanced Trajectory

Class	Precision	Recall	F1-score	Support
0	0.71	0.68	0.70	810,337
1	0.73	0.69	0.71	810,337
2	0.28	0.40	0.33	175,896
Accuracy		0.66		1,796,570
Macro Avg	0.57	0.59	0.58	1,796,570
Weighted Avg	0.68	0.66	0.67	1,796,570

Table 4.5: Classification Report for SVM Model on an Unbalanced Trajectory

Class	Precision	Recall	F1-Score	Support
0	0.16	0.46	0.24	203728
1	0.95	0.53	0.68	2256456
2	0.06	0.64	0.11	67760
Accuracy		0.53		2527944
Macro Avg	0.39	0.54	0.35	2527944
Weighted Avg	0.86	0.53	0.63	2527944

The final comparison between the raw point cloud, geometric filtering, and SVM prediction for a sequence is shown in Figure 4.7. The left plot

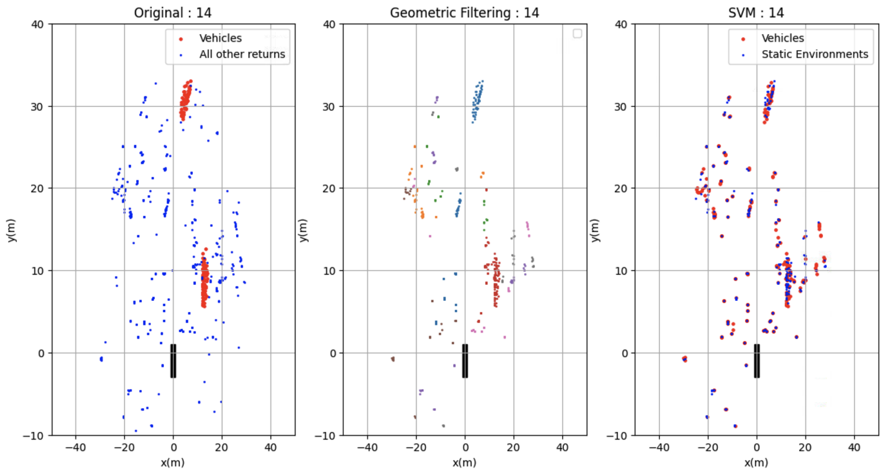


Figure 4.7: SVM prediction for vehicles, the left plot is the raw point cloud, the middle is when geometric filtering is applied, and the right plot is using SVM Predicted data.

displays the original radar detections, highlighting vehicles in red and all other returns in blue. The middle plot shows the results of geometric filtering, where distinct clusters are identified. The right plot shows the SVM prediction, where vehicles and static environments are classified based on the training model.

While the SVM model can accurately identify the two main vehicle clusters, it suffers from overclassification due to the high density of static environment points. This issue arises from the model's attempt to generalize across an imbalanced dataset, where the number of static points significantly outweighs dynamic vehicle detections. As shown in Table 4.5, the classification report highlights the limitations of the SVM model in handling such an unbalanced trajectory. The precision and recall values indicate that the model struggles to correctly classify all classes, particularly for Class 0 and Class 2, which have significantly lower F1 scores compared to Class 1.

Due to these limitations, we ultimately did not use the SVM for classification in the final trajectory estimation. Instead, the filtering approach relied on velocity and geometric filtering techniques to ensure robustness in radar-based positioning, avoiding the inconsistencies introduced by the SVM's inability to handle the dataset imbalance effectively.

4.5 The Effect of Point Cloud Filtering on Trajectory Estimation

Accurate trajectory estimation is essential for evaluating the effectiveness of radar-based positioning. This section compares the estimated vehicle trajectories obtained through scan-to-map registration and sensor fusion against ground truth data. It analyzes the impact of different filtering techniques on trajectory accuracy, highlighting improvements achieved through velocity filtering and geometric filtering. The results demonstrate how radar point cloud refinement enhances pose estimation in urban environments.

The outcomes are first presented using the RadarScenes dataset, showcasing the effectiveness of the proposed filtering and positioning methods in diverse urban conditions. These findings are then further validated using NavINST data, demonstrating improvements across a different dataset.

4.5.1 RadarScenes

The performance of the proposed filtering approach was further evaluated for a land vehicle navigation problem by comparing the estimated trajectories generated by the ICP-based pipeline presented in [38] to the ground truth trajectories for two test sequences.

The estimated trajectories are presented in Figures 4.8 and 4.9, along with its error metrics detailed in Tables 4.6 and 4.7. For the first trajectory, Figure 4.8, both filtering approaches improved positioning accuracy compared to the raw point cloud. The velocity filter reduced the RMS error from 1.319 m to 1.131 m, representing an approximate 14% improvement. Applying the geometric filter to the static points achieved the lowest RMS error of 0.902 m, corresponding to a 31.6% improvement compared to the raw point cloud (see Table 4.6).

Significant improvements were also observed in the proportion of trajectory points falling within specific error thresholds. For example, with the geometric filter applied, 76.78% of trajectory points had errors within 1 m,

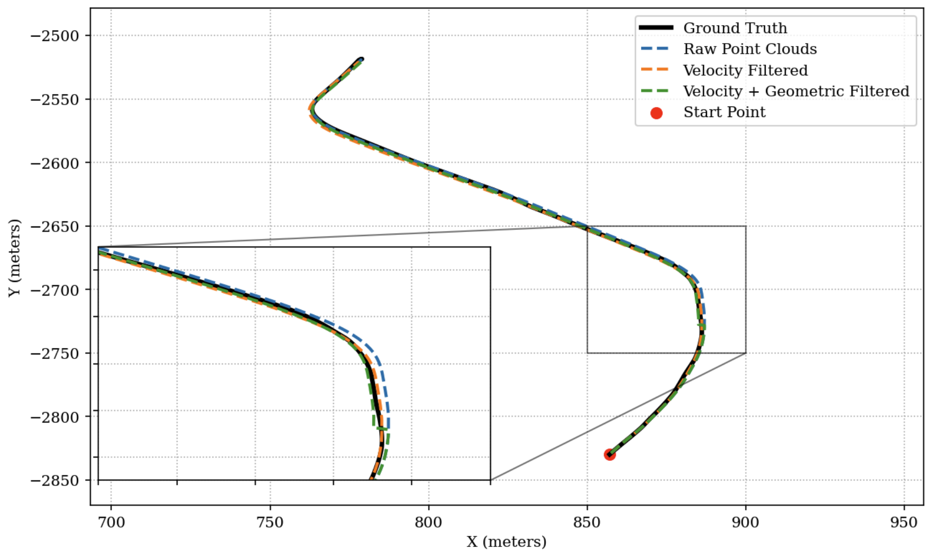


Figure 4.8: Comparison of Estimated and Ground Truth for Trajectory 1.

compared to only 27.73% for the unfiltered point cloud. Similarly, the proportion of trajectory points with errors within 50 cm increased dramatically, from 2.84% for the raw point cloud to 51.10% with the geometric filter applied (see Table 4.6).

The second trajectory, Figure 4.9, shows excellent improvements for the velocity filter only, while the geometric filter on the static points reduced positional accuracy, which can be attributed to multiple factors, including the number of radar returns, removal of static points that were classified as noise, and environmental challenges like on road obstruction due to traffic. Velocity filtering reduced the RMS error from 3.986 m to 0.643 m, achieving approximately 84% improvement. Using the raw point cloud for ICP resulted in positional accuracy being within 1 m only 3.8% of the time, whereas the velocity filter increased this metric dramatically to 89.06% of the time, demonstrating its ability to isolate meaningful static points in urban environments ((Table 4.7)). The geometric filter resulted in reduced positional accuracy compared to the velocity filter, which can be attributed to several factors. For instance, vehicles moving at high speeds can distort the spatial positioning of static points, making it challenging for the geometric filter to cluster them

Table 4.6: Error Metrics for Trajectory 1 across raw, velocity-filtered, and geometric-filtered point clouds.

Filter	Error Metric	Raw Point Cloud	Velocity Filtered	Velocity + Geometric Filtered
Trajectory 1	**Max Error (m)**	2.740	2.384	**2.385**
	Within 2m (%)	**92.18**	90.28	91.47
	Within 1m (%)	27.73	66.11	**76.78**
	Within 50cm (%)	2.84	44.55	**51.10**
	RMS Error (m)	1.319	1.131	**0.902**
	MSE (m)	1.741	1.280	**0.960**

accurately. Conversely, when the vehicle slows down or comes to a complete stop, such as at a road crossing, the radar experiences minimal returns from the environment. This lack of radar detections causes the geometric filter to classify many points as noise, further reducing the point cloud available for registration. The surrounding environment can also be challenging; if the static features are too far apart or large groups of parked vehicles are parked closely together, this can cause misclassification within the point cloud. In areas where static features are spaced too far apart or large groups of parked vehicles are closely clustered, the geometric filter may struggle to differentiate meaningful static landmarks from noise. High-traffic areas can introduce additional challenges, such as large moving vehicles obstructing the radar's line of sight and limiting its ability to capture static environments.

Table 4.7: Error Metrics for Trajectory 2 across raw, velocity-filtered, and geometric-filtered point clouds.

Filter	Error Metric	Raw Point Cloud	Velocity Filtered	Velocity + Geometric Filtered
Trajectory 2	**Max Error (m)**	6.286	**1.517**	3.628
	Within 2m (%)	18.33	**100.00**	78.63
	Within 1m (%)	3.84	**89.06**	48.37
	Within 50cm (%)	0.00	**49.36**	24.83
	RMS Error (m)	3.986	**0.643**	1.638
	MSE (m)	15.892	**0.414**	2.683

The velocity filter processes data at an average speed of 1.2 ms per frame, while the geometric filter takes 2.1 ms per frame. The ICP processing time per

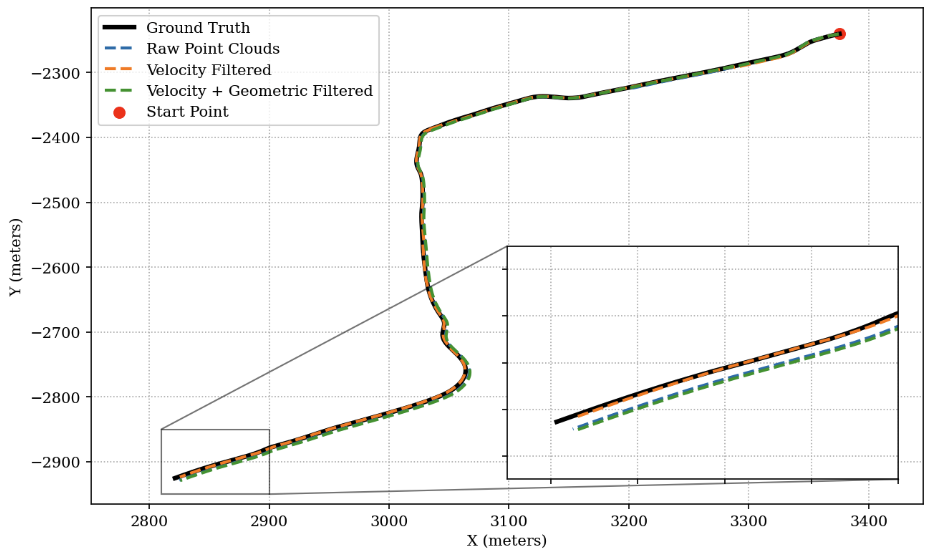


Figure 4.9: Comparison of Estimated and Ground Truth for Trajectory 2.

Table 4.8: Processing times for ICP computation on velocity-filtered and geometric-filtered point clouds for Trajectories 1 and 2.

Trajectory	ICP for Velocity (s)	ICP for Velocity & Geometric (s)
Trajectory 1	0.016	0.016
Trajectory 2	0.030	0.010

epoch, as detailed in Table 4.8, highlights the computational efficiency of the proposed filtering methods. With the velocity filter applied, ICP achieves a maximum processing time of 0.031 s (32 Hz), whereas the addition of the geometric filter further reduces it to 0.019 s (52 Hz). These results demonstrate that the filtering pipeline operates well within the update rate of automotive radars, providing additional computational headroom for real-time applications.

Experimental results indicate that radar-based positioning accuracy is significantly enhanced in urban environments using the proposed velocity and geometric filtering methods. However, their effectiveness varies depending on specific environmental conditions, the characteristics of the radar point cloud, and both the radar sensor's detection rate and the number of detections per scan. When the radar sensor has a high detection rate and captures more detections per scan, the filters perform better due to the abundance of data available for processing. This increased data density allows for more accurate identification and classification of objects, improving the overall positioning accuracy. In situations where the radar detection rate is low, either sensor-related or when the vehicle is stationary, the filters have less data to work with, which can reduce the filter's effectiveness and accuracy of the overall positioning solution.

The velocity filter consistently improved positioning accuracy across different trajectories by effectively removing dynamic objects. By utilizing Doppler velocity measurements and the vehicle's motion, the filter distinguishes between static and dynamic objects with high efficiency. The computational simplicity of the velocity filter ensures it can operate in real-time applications without imposing significant processing overhead. However, the geometric filter's performance is more sensitive to the density and distribution of the radar point cloud. In segments where the point cloud is dense, the geometric filter can refine the data by eliminating noise and small irrelevant clusters. This refinement enhances the performance of the ICP algorithm, leading to more accurate positioning. When the point cloud is sparse, which can occur when the vehicle is stationary or moving slowly, the geometric filter may inadvertently remove too many points. This over-filtering results in insufficient data for accurate ICP alignment. Ultimately reducing positioning accuracy.

An additional challenge arises from the presence of large clusters of parked vehicles along the curbsides. These parked cars often produce radar point clouds similar to static structures like building edges. This issue becomes particularly pronounced when the vehicle is turning at intersections. In these segments, the parked cars at these locations can be misinterpreted as building edges. As a result, the ICP alignment process could incorrectly adjust the vehicle's trajectory, causing an offset or preventing the turn from being fully completed. This misclassification highlights the need for more sophisticated object differentiation methods within the filter to distinguish between parked vehicles and true static structures, especially during maneuvers such as turns. Another major factor affecting the filter's performance is the radar sensor's reliance on relative motion between the sensor and the detected object. When the vehicle is stopped or moving slowly, the radar registers fewer returns from static objects due to the lack of relative motion, leading to an even sparser point cloud. This limitation shows the importance of vehicle motion in radar-based positioning systems.

The proposed filtering methods show significant promise for improving radar-based positioning in urban environments. The approach can be further improved by addressing these identified challenges through adaptive filtering strategies, fusion with additional onboard motion sensors, and filter refinement to adjust to real-time environmental changes.

4.5.2 NavINST

This section presents the trajectory estimation results and performance analysis across five different segments from the NavINST dataset. Unlike the RadarScenes evaluation, which focuses on assessing the effectiveness of velocity filtering and geometric filtering on raw radar-based positioning, the NavINST dataset extends the analysis by incorporating multi-sensor fusion. Specifically, the UKF and ES-EKF frameworks are used to integrate radar-based ICP positioning with onboard motion sensors, including IMU and odometer data.

The comparison between UKF and ES-EKF highlights key differences in terms of accuracy, robustness, and computational efficiency under varying urban conditions. Additionally, an analysis of the impact of filtering techniques—including velocity filtering and geometric filtering—on sensor fusion accuracy is conducted. The influence of ICP iteration reduction is also assessed to understand its trade-off between positioning performance and computational efficiency.

Radar ICP-based positioning was fused with IMU and odometry measurements during this evaluation. The results provided insights into how pre-filtered radar data improves state estimation when combined with onboard motion sensors, offering a more robust and accurate positioning framework for GNSS-denied environments.

Segment 1

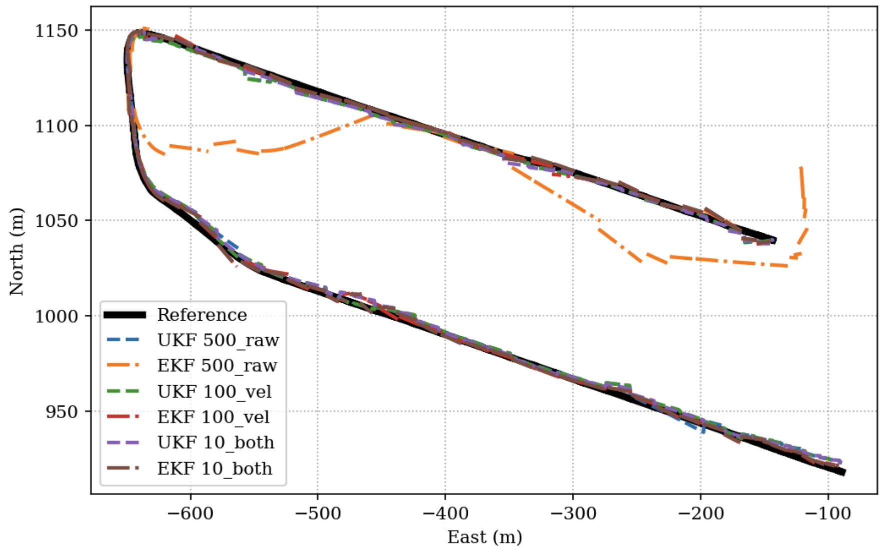


Figure 4.10: Trajectory estimation for segment 1

For this urban trajectory segment, failed estimations were observed for ES-EKF when using 500 ICP iterations and 100 iterations with raw radar point clouds for both ES-EKF and UKF (Table 4.9). The results, as shown in Figure 4.10, demonstrate that applying the velocity filter significantly improves performance. In contrast, the geometric filter primarily acts as a fine-tuning step rather than a critical component in the filtering process. A key observation is that UKF demonstrates robust estimation capability even without extensive preprocessing, while ES-EKF, although less robust under challenging conditions, often provides higher accuracy when it functions correctly. The ability to preprocess the radar point cloud enhances overall estimation accuracy and improves the algorithm's computational efficiency, allowing for a reduction in ICP iterations without compromising positioning performance. One challenge observed in this segment was related to the reference map. Certain areas contained gaps between buildings when, in reality, non-building structures such as parking lots were present. Additionally, while navigating Princess Street, the trajectory exhibited zig-zag behaviour, which required more corrections. This effect was likely influenced by large clusters of parked vehicles, causing occlusions and inconsistencies in the radar-based positioning. This segment was recorded during daytime off-peak hours, resulting in

Table 4.9: Error Metrics for UKF and EKF for Segment 1

ICP Iterations	Filter	ES-EKF RMSE(m)	ES-EKF Error 1m (%)	ES-EKF Error 3m (%)	UKF Error RMSE(m)	UKF Error 1m (%)	UKF Error 3m (%)
10	Both	2.133	26.134	82.700	2.935	21.598	60.216
	Vel	2.123	27.171	83.197	4.035	19.654	52.203
50	Both	2.065	27.019	84.622	2.990	6.328	57.084
	Vel	2.057	27.797	84.320	4.035	19.654	52.203
100	Both	2.104	26.485	84.467	2.874	6.352	59.571
	Vel	2.101	27.803	84.381	2.817	6.351	64.442
	Raw	118.263	5.528	36.375	249.1	6.610	41.521
500	Both	2.095	25.988	84.489	2.969	6.330	59.667
	Vel	2.092	26.312	84.295	2.911	6.351	62.275
	Raw	54.545	5.513	36.401	2.895	7.445	61.028

lower traffic density and minimal pedestrian activity. The reduced number of dynamic obstacles likely contributed to more stable sensor fusion, improving trajectory estimation accuracy.

Segment 2

Segment 2 evaluates the filter performance in an environment consisting of one long straight segment followed by two right-hand turns. This segment highlights ICP alignment challenges, particularly in areas with large clusters of parked cars, as shown in Figure 4.11. These parked vehicles, when processed by ICP, can be misclassified as static building edges, leading to misalignment errors in the estimated trajectory.

During the first right-hand turn, the radar's ability to detect static objects

Table 4.10: Error Metrics for UKF and EKF for Segment 2

ICP Iterations	Filter	ES-EKF RMSE(m)	ES-EKF Error 1m (%)	ES-EKF Error 3m (%)	UKF Error RMSE(m)	UKF Error 1m (%)	UKF Error 3m (%)
10	Both	5.821	14.367	48.377	4.918	5.925	29.545
	Vel	5.845	14.773	48.052	4.933	5.925	29.942
50	Both	7.836	14.169	48.812	4.175	5.979	32.187
	Vel	8.122	14.169	49.631	4.179	6.143	30.958
100	Both	14.606	14.951	34.477	4.125	8.252	30.310
	Vel	3.299	18.522	63.688	4.107	8.252	30.310
	Raw	3.744	25.305	70.220	3.994	7.925	31.699
500	Both	3.239	17.384	67.994	4.060	8.205	30.382
	Vel	3.264	17.628	68.075	4.062	10.154	30.138
	Raw	3.753	20.911	69.894	3.917	9.098	32.413

decreases due to the limited field of view and occlusions. However, the positioning solution remains accurate as the combination of building corners and structural edges provides reliable reference points for correction and realignment. This highlights the importance of environmental features in aiding radar-based positioning, even in complex urban settings.

A significant source of error in this segment occurs just before the second right turn. On the right side of the vehicle, there is a large open parking lot enclosed by concrete road barriers, spanning nearly half a block. Since these barriers are not represented in the OSM reference map, this section is left open, leading to inconsistencies in map matching and positioning estimation. Additionally, this road segment consists of four lanes, further reducing radar point cloud density, making positioning more difficult. These factors contribute to position estimation failures at this corner.

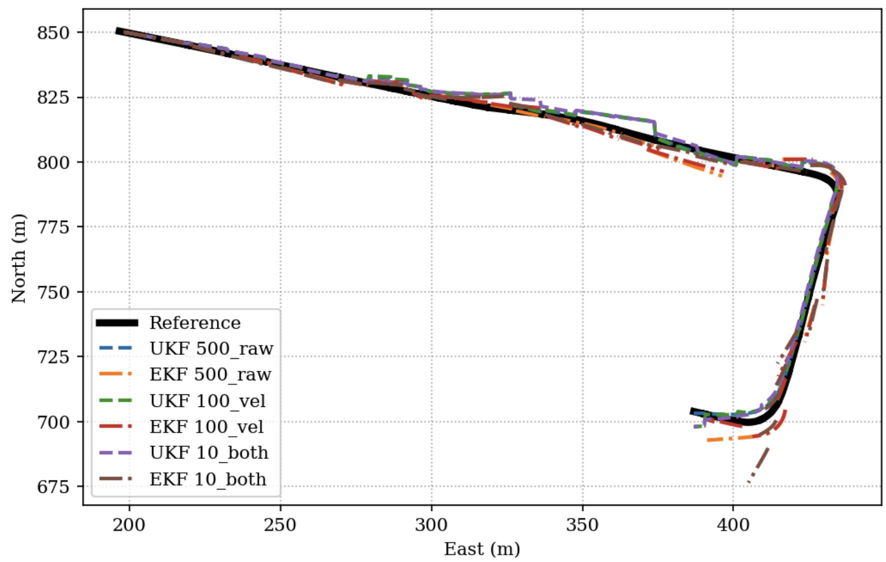


Figure 4.11: Trajectory estimation for segment 2

By observing Table 4.10, we observe that applying the geometric filter reduced the effectiveness of the positioning solution. This is likely due to the higher vehicle speed in this segment, where even minimal point removal in the geometric filter reduces the availability of important structural features for ICP alignment. Consequently, the geometric filter, while beneficial in some segments, may negatively impact positioning accuracy in high-speed environments where radar scan density is already sparse.

Segment 3

For this segment, the percentage of trajectory error is lower with UKF compared to ES-EKF, as shown in Table 4.11. However, the robustness of the UKF positioning solution fails at the second right-hand turn, Figure 4.12. The key landmarks at the intersection include a building to the right of the vehicle. At the same time, the left side is an open parking lot for a nearby hospital, lacking any structural barriers. This environment increases the likelihood of incorrect ICP matches, leading to a trajectory break in UKF estimations.

Examining the RMS values for successfully completed UKF estimations,

Table 4.11: Error Metrics for UKF and EKF for Segment 3

ICP Iterations	Filter	ES-EKF RMSE(m)	ES-EKF Error 1m (%)	ES-EKF Error 3m (%)	UKF Error RMSE(m)	UKF Error 1m (%)	UKF Error 3m (%)
10	Both	3.020	9.637	50.909	3.020	19.143	57.203
	Vel	2.945	9.342	50.178	3.038	19.143	57.690
50	Both	4.943	8.652	42.385	264.510	19.054	66.591
	Vel	4.935	8.522	43.616	308.167	19.054	66.688
100	Both	5.013	7.615	43.195	394.433	19.054	66.850
	Vel	4.700	9.170	44.524	53.44	19.054	66.559
	Raw	5.272	14.841	42.093	2.791	19.054	62.378
500	Both	4.921	11.666	46.047	115.713	19.054	66.818
	Vel	4.639	11.925	46.111	1442.607	19.054	66.494
	Raw	59.917	13.970	39.214	2.793	19.504	62.281

it is evident that preprocessing the radar point cloud contributed to estimation errors. Additionally, running ICP for only 10 iterations as a fine-tuning step for UKF prediction improved the overall estimation accuracy. This suggests that a lower number of ICP iterations prevents incorrect feature associations, ensuring the algorithm aligns the radar point cloud to the nearest correct matches rather than searching for alternative, less reliable alignments.

One important observation in this segment is that UKF's reliance on ICP corrections affects its performance, particularly in areas with fewer distinct environmental features. Since UKF does not predict error states but instead estimates the full position, any incorrect ICP correction can significantly shift the trajectory, leading to positioning errors. In contrast, ES-EKF does not estimate absolute position but instead predicts the error in the estimated state. If ES-EKF receives a large correction from ICP, the fusion filter is designed to moderate the correction, preventing drastic position shifts that could degrade positioning stability. This behaviour makes ES-EKF more resilient to large

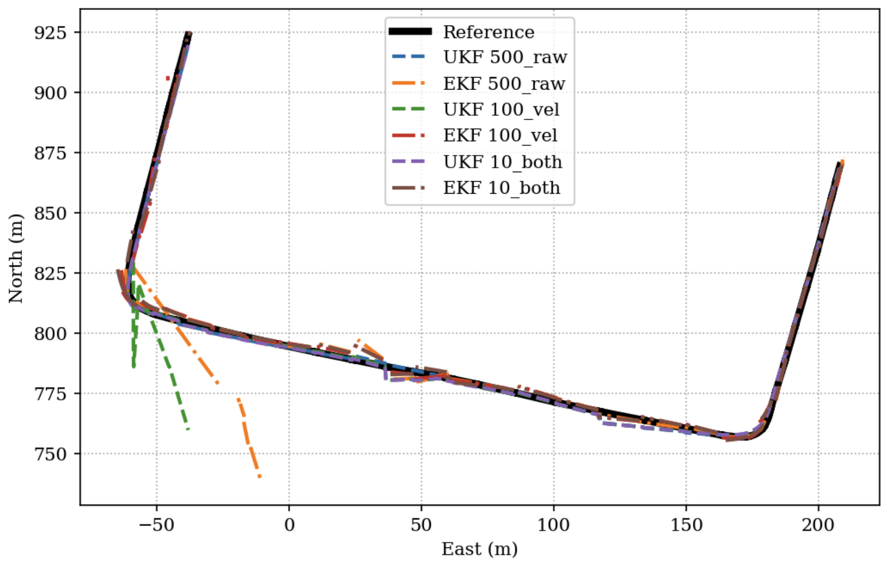


Figure 4.12: Trajectory estimation for segment 3

ICP errors, as the filter smooths out sudden position changes, reducing the risk of significant trajectory deviations.

Segment 4

Segment 4 is located within a suburban environment characterized by wide streets, large trees, and residential homes with front yards and fences. As observed in Figure 4.13 and Table 4.12, all ES-EKF positioning estimations failed except the trajectories with 10 ICP iterations, whereas UKF was able to maintain a valid trajectory estimation except the 50 ICP iteration.

In suburban areas, OSM reference maps lack sufficient structural detail, leading to failures in ICP-based positioning. The radar detects parked cars, trees, and fences, which can be misclassified as building edges, causing ICP misalignment and positioning errors. However, when switching to a higher-resolution LiDAR-based reference map, positioning accuracy improved significantly. The Lidar map better represents the suburban environment, allowing UKF to maintain a stable trajectory, even though ES-EKF failed in most cases.

Table 4.12: Error Metrics for UKF and EKF for Segment 4

ICP Iterations	Filter	ES-EKF RMSE(m)	ES-EKF Error 1m (%)	ES-EKF Error 3m (%)	UKF Error RMSE(m)	UKF Error 1m (%)	UKF Error 3m (%)
10	Both	3.599	8.339	73.449	3.661	6.426	42.746
	Vel	3.566	12.342	72.515	3.660	6.437	42.724
50	Both	30.886	4.536	9.139	19.111	5.992	43.646
	Vel	34.517	4.536	9.206	19.108	5.992	43.635
100	Both	147.220	4.485	4.485	3.633	8.004	40.411
	Vel	121.323	4.450	4.450	3.216	9.313	48.217
	Raw	147.714	4.491	4.491	3.179	8.312	49.738
500	Both	114.270	4.536	4.536	3.633	8.004	40.411
	Vel	114.246	4.536	4.536	3.630	6.581	40.411
	Raw	148.201	4.536	4.536	3.563	6.276	40.430

For ES-EKF, reducing ICP iterations improved positioning accuracy. A lower iteration count resulted in closer point matches, preventing the ICP algorithm from converging to incorrect local minima [40]. This effect was particularly pronounced in LiDAR maps, where increased point density led to the ICP algorithm searching for alternative matches rather than aligning with the true motion correction required for IMU drift compensation.

Pre-filtering the radar point cloud did not significantly improve performance in this segment. The likely reason is that suburban environments contain fewer dynamic objects, meaning the velocity filter removed fewer points, limiting its effectiveness. Furthermore, geometric filtering removed crucial static features, reducing point cloud density and negatively impacting ICP accuracy. These results suggest that while pre-filtering techniques benefit urban environments with dynamic obstacles, they may not always be advantageous in suburban settings with sparse but critical static features.

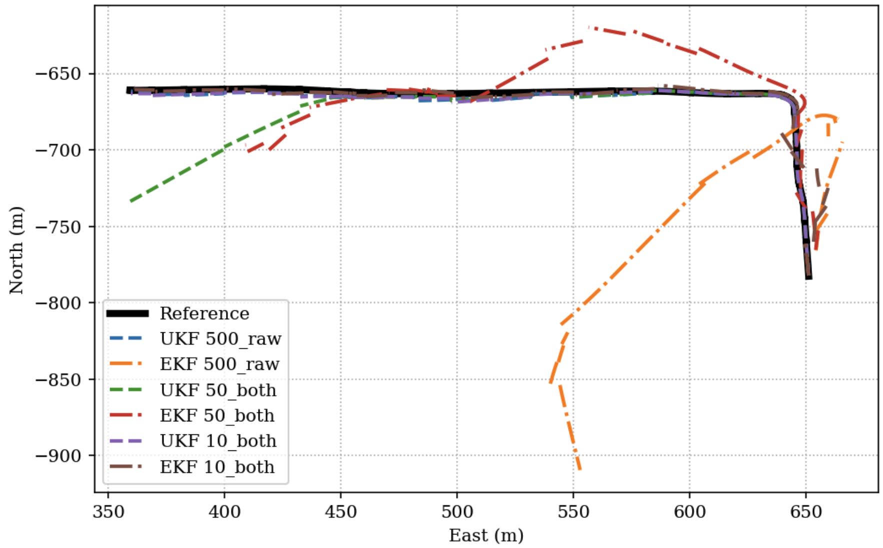


Figure 4.13: Trajectory estimation for segment 4

Segment 5

Segment 5 takes place in the same location as Segment 1 but under conditions with increased vehicle and pedestrian activity. This segment evaluates the performance of both filters in a more dynamic urban environment. As observed in Table 4.13 and Figure 4.14, the ES-EKF outperformed UKF, achieving significantly lower RMS errors across multiple configurations.

The failed UKF solutions at 500 ICP iterations for both pre-filters applied suggest that the geometric filter removed a significant number of static environmental points, leading to misalignment in ICP-based corrections. This highlights a key limitation in radar-based positioning: Over—filtering can reduce the number of available features for ICP registration, negatively impacting positioning accuracy.

Interestingly, the best performance for both fusion filters was observed at 50 ICP iterations. Limiting the number of ICP iterations may have constrained the registration process to focus on the closest matching points, reducing the likelihood of ICP converging to an incorrect local minimum. This finding suggests that a carefully tuned ICP iteration count can enhance stabil-

Table 4.13: Error Metrics for UKF and EKF for Segment 5

ICP Iterations	Filter	ES-EKF RMSE(m)	ES-EKF Error 1m (%)	ES-EKF Error 3m (%)	UKF Error RMSE(m)	UKF Error 1m (%)	UKF Error 3m (%)
10	Both	2.347	21.304	74.708	3.388	5.347	58.814
	Vel	2.284	17.563	77.900	3.400	5.156	58.490
50	Both	1.878	31.108	85.224	2.948	4.213	65.975
	Vel	1.866	34.349	84.738	2.973	4.083	65.360
100	Both	1.893	24.368	85.256	3.221	4.213	45.748
	Vel	1.870	23.914	85.450	3.401	4.106	43.239
	Raw	31.606	19.420	58.553	2.941	3.532	36.390
500	Both	1.911	20.642	70.706	65.029	4.213	12.897
	Vel	1.879	23.882	71.905	3.426	4.083	42.968
	Raw	27.570	19.426	58.540	2.921	3.532	57.129

ity and robustness, particularly in environments with dynamic obstacles such as moving vehicles and pedestrians.

4.6 Real Time Performance Analysis

Beyond accuracy and robustness, computational efficiency is crucial for real-world implementation. The ES-EKF has already been demonstrated to operate in real-time, ensuring seamless integration into high-frequency sensor fusion pipelines. Experimental evaluation indicates that the combined UKF and ICP correction process operates comfortably within real-time constraints. In our updated results, the velocity update—responsible for efficiently filtering dynamic radar returns—requires only 0.965 ms per frame. Although the geo-

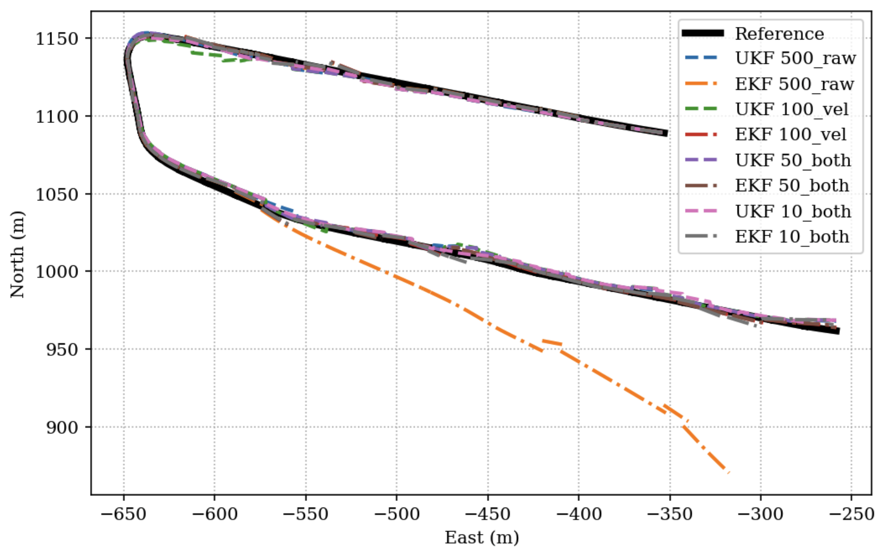


Figure 4.14: Trajectory estimation for segment 5

metric filtering stage is more computationally intensive, it averages 47.392 ms per frame, while the subsequent UKF update itself is highly efficient at just 0.131 ms on average. These timings confirm that even when incorporating ICP-based corrections, the overall processing remains fast enough for real-time automotive radar applications. Further analysis of the ICP processing reveals a clear trade-off between iteration count and processing time. Specifically, when running 500 iterations, the average processing time is 136.003 ms; reducing the count to 100 iterations brings the time down to 94.236 ms; 50 iterations require 76.592 ms on average; and a further reduction to 10 iterations results in only 25.690 ms per frame. Notably, while higher iteration counts can offer marginal alignment improvements, accuracy gains beyond 50 iterations become negligible. Thus, significantly reducing the number of ICP iterations can dramatically enhance computational efficiency without compromising positioning accuracy.

Chapter 5

Conclusion and Future Work

5.1 Conclusion

This study developed a radar-based positioning system using a combination of velocity and geometric filtering techniques designed to improve scan-to-map registration in urban areas. The proposed point cloud filtering methods were tested using the RadarScenes and NavINST datasets, demonstrating their effectiveness in improving trajectory estimation through radar point cloud refinement.

The velocity filter, leveraging Doppler velocity and odometry data, effectively removed dynamic objects from radar scans while maintaining a high processing speed suitable for real-time applications. The geometric filter, using DBScan, further improved the radar point cloud by removing additional ghost detections and noise. These two filters significantly improved the quality of radar data for ICP alignment, thereby improving positioning performance.

Experiments conducted on the RadarScenes dataset showed substantial accuracy improvements when applying the velocity filter. However, the geometric filter's impact was dependent on environmental conditions and provided mixed results in sparse environments with low numbers of static features. The NavINST dataset verified these findings, confirming the filters' ability to enhance positioning accuracy across different urban landscapes.

Two sensor fusion approaches, an Error-State Extended Kalman Filter (ES-EKF) and an Unscented Kalman Filter (UKF), were compared to fuse radar scan to map positioning updates with INS in the NavINST dataset. The ES-EKF demonstrated greater robustness to ICP errors, effectively moderating large corrections and improving trajectory stability. The UKF, while offering a more direct estimation approach, was more susceptible to incorrect ICP updates in feature-sparse environments. The analysis of different ICP iteration counts revealed that reducing iterations could maintain or even improve positioning accuracy by preventing over-corrections and convergence to false local minima while also significantly reducing computational costs.

The experiment shows that using pre-processing techniques on radar point clouds can improve positioning estimation. The combination of velocity filtering, geometric filtering, and sensor fusion techniques provides a robust framework for positioning, particularly in GNSS-denied urban environments where traditional positioning systems struggle.

5.2 Future Work

While the proposed filtering and sensor fusion techniques demonstrated promising results, several areas for improvement and future research remain.

1. **Adaptive Filtering Strategies**: Future work could investigate adaptive filtering strategies that dynamically adjust parameters based on the environment, traffic conditions, and sensor confidence levels.

2. **Multi-Sensor Fusion for Enhanced Positioning**: While this study focused on radar-based positioning, integrating complementary sensors such as Lidar, cameras, and GNSS could provide more reliable positioning in urban environments. Future research could explore sensor fusion frameworks that leverage radar's resilience in adverse conditions while utilizing Lidar and vision for enhanced feature association and landmark-based positioning.

3. **Large-Scale Real-World Evaluation**: Expanding the evaluation to a more diverse set of urban and highway environments, including different weather conditions, would provide deeper insights into the generalizability of the proposed approach.

4. **Real-Time Implementation and Computational Efficiency**: Given the real-time constraints of autonomous driving, optimizing the computational efficiency of radar-based positioning is crucial to achieving a higher refresh rate and ensuring timely updates.

Bibliography

[1] Aboelmagd Noureldin, Tashfeen Karamat, and Jacques Georgy. *Fundamentals of Inertial Navigation, Satellite-based Positioning and their Integration*. Springer, 2013. ISBN: 978-3-642-30465-1. DOI: 10.1007/978-3-642-30466-8. URL: https://doi.org/10.1007/978-3-642-30466-8.

[2] Yanda Guo, Xuyou Li, and Qingwen Meng. "A Runs Test-Based Kalman Filter With Both Adaptability and Robustness With Application to INS/GNSS Integration". In: *IEEE Sensors Journal* 22.23 (2022), pp. 22919–22930. DOI: 10.1109/JSEN.2022.3213700.

[3] Ekim Yurtsever et al. "A survey of autonomous driving: Common practices and emerging technologies". In: *IEEE access* 8 (2020), pp. 58443–58469.

[4] Thomas Clunie et al. "Development of a Perception System for an Autonomous Surface Vehicle using Monocular Camera, LIDAR, and Marine RADAR". In: *2021 IEEE International Conference on Robotics and Automation (ICRA)*. 2021, pp. 14112–14119. DOI: 10.1109/ICRA48506.2021.9561275.

[5] Jelena Kocić, Nenad Jovičić, and Vujo Drndarević. "Sensors and Sensor Fusion in Autonomous Vehicles". In: *2018 26th Telecommunications Forum (TELFOR)*. 2018, pp. 420–425. DOI: 10.1109/TELFOR.2018.8612054.

[6] Dominik Kellner et al. "Instantaneous ego-motion estimation using Doppler radar". In: *16th International IEEE Conference on Intelligent Transportation Systems (ITSC 2013)*. 2013, pp. 869–874. DOI: 10.1109/ITSC.2013.6728341.

[7] A. Kamann et al. “Automotive Radar Multipath Propagation in Uncertain Environments”. In: *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*. 2018, pp. 859–864.

[8] Abhinav Narula, Rushabh Patel, and Sukhvir Singh Cheema. “Automotive Radar-Based Localization Using Sparse Point Clouds”. In: *arXiv preprint arXiv:2005.00704* (2020).

[9] E. Ward and J. Folkesson. “Vehicle localization with low cost radar sensors”. In: *2016 IEEE Intelligent Vehicles Symposium (IV)*. Gothenburg, Sweden, 2016, pp. 864–870. DOI: `10.1109/IVS.2016.7535489`.

[10] Fabian Roos et al. “Ghost target identification by analysis of the Doppler distribution in automotive scenarios”. In: *2017 18th International Radar Symposium (IRS)*. 2017, pp. 1–9. DOI: `10.23919/IRS.2017.8008128`.

[11] Guomeng Zhang. “Single Acceleration Sensor Gyroscope-free Vehicle Inertial Dead Reckoning”. In: *2024 5th International Conference on Computer Engineering and Application (ICCEA)*. 2024, pp. 1668–1671. DOI: `10.1109/ICCEA62105.2024.10603648`.

[12] Ashish Pidurkar, Ranjit Sadakale, and A.K. Prakash. “Monocular Camera based Computer Vision System for Cost Effective Autonomous Vehicle”. In: *2019 10th International Conference on Computing, Communication and Networking Technologies (ICCCNT)*. 2019, pp. 1–5. DOI: `10.1109/ICCCNT45670.2019.8944496`.

[13] Sung-Tae Kim et al. “External Vehicle Positioning System Using Multiple Fish-Eye Surveillance Cameras for Indoor Parking Lots”. In: *IEEE Systems Journal* 15.4 (2021), pp. 5107–5118. DOI: `10.1109/JSYST.2020.3019296`.

[14] Eslam Mounier et al. “LiDAR-Based Multi-Sensor Fusion with 3D Digital Maps for High-Precision Positioning”. In: *IEEE Internet of Things Journal* (2024), pp. 1–1. DOI: `10.1109/JIOT.2024.3492913`.

[15] Hermann Rohling and Christof Moller. “Radar waveform for automotive radar systems and applications”. In: *2008 IEEE Radar Conference*. 2008, pp. 1–4. DOI: `10.1109/RADAR.2008.4721121`.

[16] Eugin Hyun, Young-Seok Jin, and Jong-Hun Lee. “Moving and stationary target detection scheme using coherent integration and subtraction for automotive FMCW radar systems”. In: *2017 IEEE Radar Conference (RadarConf)*. 2017, pp. 0476–0481. DOI: 10.1109/RADAR.2017.7944250.

[17] Xueru Ding et al. “Theory and practice: A two-channel automotive radar for three-dimensional object detection”. In: *2015 European Radar Conference (EuRAD)*. 2015, pp. 265–268. DOI: 10.1109/EuRAD.2015.7346288.

[18] Emma Dawson et al. “Merits and Limitations of Automotive Radar for Land Vehicle Positioning in Challenging Environments”. In: *IEEE Sensors Journal* 23.21 (2023), pp. 26691–26700. DOI: 10.1109/JSEN.2023.3318069.

[19] R. Saussard, S. Zair, and G. Pita-Gil. “Ego-Motion Estimation with Static Object Detections from Low Cost Radars”. In: *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*. Maui, HI, USA, 2018, pp. 1858–1863. DOI: 10.1109/ITSC.2018.8569738.

[20] D. Kellner et al. “Instantaneous ego-motion estimation using multiple Doppler radars”. In: *2014 IEEE International Conference on Robotics and Automation (ICRA)*. Hong Kong, China, 2014, pp. 1592–1597. DOI: 10.1109/ICRA.2014.6907064.

[21] Mitali Kher. *Ackermann Steering Mechanism*. Accessed: 2024-08-28. 2021. URL: https://medium.com/@mitali.kher21/ackermann-steering-mechanism-d98461087949.

[22] P. Besl and N. D. McKay. “A Method for Registration of 3-D Shapes”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 14.2 (1992), pp. 239–256.

[23] Emma Dawson et al. “Radar-Based Multisensor Fusion for Uninterrupted Reliable Positioning in GNSS-Denied Environments”. In: *IEEE Transactions on Intelligent Transportation Systems* 23.12 (2022), pp. 23384–23398. DOI: 10.1109/TITS.2022.3202139.

[24] Lakshay Narula, Peter A. Iannucci, and Todd E. Humphreys. "Automotive-Radar-Based 50-cm Urban Positioning". In: *2020 IEEE/ION Position, Location and Navigation Symposium (PLANS)*. 2020, pp. 856–867. DOI: 10.1109/PLANS46316.2020.9109917.

[25] S. H. Cen and P. Newman. "Precise Ego-Motion Estimation with Millimeter-Wave Radar Under Diverse and Challenging Conditions". In: *2018 IEEE International Conference on Robotics and Automation (ICRA)*. Brisbane, QLD, Australia, 2018, pp. 6045–6052. DOI: 10.1109/ICRA.2018.8460687.

[26] O. Schumann et al. "RadarScenes: A Real-World Radar Point Cloud Data Set for Automotive Applications". In: *2021 IEEE 24th International Conference on Information Fusion (FUSION)*. Sun City, South Africa, 2021, pp. 1–8. DOI: 10.23919/FUSION49465.2021.9627037.

[27] X. Cai and K. Sarabandi. "A Machine Learning Based 77 GHz Radar Target Classification for Autonomous Vehicles". In: *2019 IEEE International Symposium on Antennas and Propagation and USNC-URSI Radio Science Meeting*. Atlanta, GA, USA, 2019, pp. 371–372. DOI: 10.1109/APUSNCURSINRSM.2019.8888647.

[28] D. Niederlöhner et al. "Self-Supervised Velocity Estimation for Automotive Radar Object Detection Networks". In: *2022 IEEE Intelligent Vehicles Symposium (IV)*. Aachen, Germany, 2022, pp. 352–359. DOI: 10.1109/IV51971.2022.9827295.

[29] J. Rock et al. "Deep Interference Mitigation and Denoising of Real-World FMCW Radar Signals". In: *2020 IEEE International Radar Conference (RADAR)*. Washington, DC, USA, 2020, pp. 624–629. DOI: 10.1109/RADAR42522.2020.9114627.

[30] K. Patel et al. "Deep Learning-based Object Classification on Automotive Radar Spectra". In: *2019 IEEE Radar Conference (RadarConf)*. Boston, MA, USA, 2019, pp. 1–6. DOI: 10.1109/RADAR.2019.8835775.

[31] S. Lu et al. "Efficient Deep-Learning 4D Automotive Radar Odometry Method". In: *IEEE Transactions on Intelligent Vehicles* 9.1 (Jan. 2024), pp. 879–892. DOI: 10.1109/TIV.2023.3311102.

[32] Zhichun Zhao et al. "Point Cloud Features-Based Kernel SVM for Human-Vehicle Classification in Millimeter Wave Radar". In: *IEEE Access* 8 (2020), pp. 26012–26021. DOI: 10.1109/ACCESS.2020.2970533.

[33] Yuta Sasaki, Takanori Emaru, and Ankit A. Ravankar. "SVM based Pedestrian Detection System for Sidewalk Snow Removing Machines". In: *2021 IEEE/SICE International Symposium on System Integration (SII)*. 2021, pp. 700–701. DOI: 10.1109/IEEECONF49454.2021.9382618.

[34] Renyuan Zhang and Siyang Cao. "Robust and Adaptive Radar Elliptical Density-Based Spatial Clustering and labeling for mmWave Radar Point Cloud Data". In: *2019 53rd Asilomar Conference on Signals, Systems, and Computers*. 2019, pp. 919–924. DOI: 10.1109/IEEECONF44664.2019.9048869.

[35] Qin Jingjie et al. "A Clustering Algorithm for Traffic Targets Using Millimeter-Wave Radar". In: *2023 Cross Strait Radio Science and Wireless Technology Conference (CSRSWTC)*. 2023, pp. 1–3. DOI: 10.1109/CSRSWTC60855.2023.10427405.

[36] Siyi Xie et al. "Millimeter-Wave Radar Target Detection Based on Inter-frame DBSCAN Clustering". In: *2022 IEEE 22nd International Conference on Communication Technology (ICCT)*. 2022, pp. 1703–1708. DOI: 10.1109/ICCT56141.2022.10072664.

[37] Santhana Raj and Dipanjan Ghosh. "Optimized DBSCAN with Improved Static clutter removal for High Resolution Automotive Radars". In: *2022 19th European Radar Conference (EuRAD)*. 2022, pp. 1–4. DOI: 10.23919/EuRAD54643.2022.9924692.

[38] Emma Dawson. "Integrated Remote Sensing and Map Registration System for High-Precision Positioning in Covered Parking Garages". In: *Proceedings of the 36th International Technical Meeting of the Satellite Division of The Institute of Navigation (ION GNSS+ 2023)*. Denver, Colorado, Sept. 2023, pp. 1633–1643. DOI: 10.33012/2023.19210.

[39] Haoyu Chen et al. "Fusion of LiDAR and MMW Radar Point Clouds for Environment Sensing". In: *2023 International Applied Computational Electromagnetics Society Symposium (ACES-China)*. 2023, pp. 1–3. DOI: 10.23919/ACES-China60289.2023.10249238.

[40] Mohamad Albilani and Amel Bouzeghoub. "Localization of Autonomous Vehicle with low cost sensors". In: *2022 IEEE 19th International Conference on Mobile Ad Hoc and Smart Systems (MASS)*. 2022, pp. 339–345. DOI: 10.1109/MASS56207.2022.00056.

[41] A Aravind et al. "Enhancing GPS Position Estimation Using Multi-Sensor Fusion and Error-State Extended Kalman Filter". In: *2022 International Conference on Distributed Computing, VLSI, Electrical Circuits and Robotics (DISCOVER)*. 2022, pp. 201–206. DOI: 10.1109/DISCOVER55800.2022.9974753.

[42] Madan H T et al. "An Error-State Extended Kalman Filter Based State Estimation and Localization Algorithm for Autonomous Systems". In: *2023 International Conference on Smart Systems for applications in Electrical Sciences (ICSSES)*. 2023, pp. 1–6. DOI: 10.1109/ICSSES58299.2023.10199535.

[43] Lovro Marković et al. "Error State Extended Kalman Filter Multi-Sensor Fusion for Unmanned Aerial Vehicle Localization in GPS and Magnetometer Denied Indoor Environments". In: *2022 International Conference on Unmanned Aircraft Systems (ICUAS)*. 2022, pp. 184–190. DOI: 10.1109/ICUAS54217.2022.9836124.

[44] Simon J. Julier and Jeffrey K. Uhlmann. "A New Extension of the Kalman Filter to Nonlinear Systems". In: *Proceedings of AeroSense: The 11th International Symposium on Aerospace/Defense Sensing, Simulation, and Controls* (1997), pp. 182–193.

[45] E.A. Wan and R. Van Der Merwe. "The unscented Kalman filter for nonlinear estimation". In: *Proceedings of the IEEE 2000 Adaptive Systems for Signal Processing, Communications, and Control Symposium (Cat. No.00EX373)*. 2000, pp. 153–158. DOI: 10.1109/ASSPCC.2000.882463.

[46] Kyunghyun Ryu, Jiseock Kang, and Dongjun Lee. "Performance Comparison between EKF and UKF in GPS/INS Low Observability Conditions". In: *2021 21st International Conference on Control, Automation and Systems (ICCAS)*. 2021, pp. 1911–1916. DOI: 10.23919/ICCAS52745.2021.9649748.

[47] Zhao Pan et al. “UKF and EKF estimator design based on a nonlinear vehicle model containing UniTire model”. In: *2009 International Conference on Mechatronics and Automation*. 2009, pp. 4780–4784. DOI: 10.1109/ICMA.2009.5246445.

[48] Wolfram Burgard, Oliver Brock, and Cyrill Stachniss. “Map-Based Precision Vehicle Localization in Urban Environments”. In: *Robotics: Science and Systems III*. MIT Press, 2008, pp. 121–128.

[49] I. Belhajem, Y.B. Maissa, and A. Tamtaoui. “Improving Vehicle Localization in a Smart City with Low Cost Sensor Networks and Support Vector Machines”. In: *Mobile Netw Appl* 23 (2018), pp. 854–863. DOI: 10.1007/s11036-017-0879-9.

[50] Aqsa Qadir. *Tuning Parameters of SVM: Kernel, Regularization, Gamma, and Margin*. Accessed: 2024-08-28. 2020. URL: https://aqsa-qadir44.medium.com/tuning-parameters-of-svm-kernel-regularization-gamma-and-margin-5f2f6639121a.

[51] Zheng Yu Lang et al. “Increasing Positioning Accuracy in Urban Environments Using Radar-Based Point Clouds”. In: *Proceedings of the 2025 International Technical Meeting of The Institute of Navigation*. Long Beach, California, Jan. 2025, pp. 950–960. DOI: 10.33012/2025.20030.

[52] M. Barjenbruch et al. “Joint Spatial- and Doppler-Based Ego-Motion Estimation for Automotive Radars”. In: *2015 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2015, pp. 839–844.

[53] OpenStreetMap contributors. *OpenStreetMap*. https://www.openstreetmap.org. Accessed: 2024-11-21. 2024.

[54] *CloudCompare (version 2.13.2) [GPL software]*. Retrieved from http://www.cloudcompare.org/. 2024.

[55] Ole Schumann et al. “RadarScenes: A Real-World Radar Point Cloud Data Set for Automotive Applications”. In: *2021 IEEE 24th International Conference on Information Fusion (FUSION)*. 2021, pp. 1–8. DOI: 10.23919/FUSION49465.2021.9627037.

[56] Paulo Ricardo Marques de Araujo et al. *The NavINST Dataset for Multi-Sensor Autonomous Navigation*. 2025. arXiv: 2502.13863 [cs.RO]. URL: https://arxiv.org/abs/2502.13863.