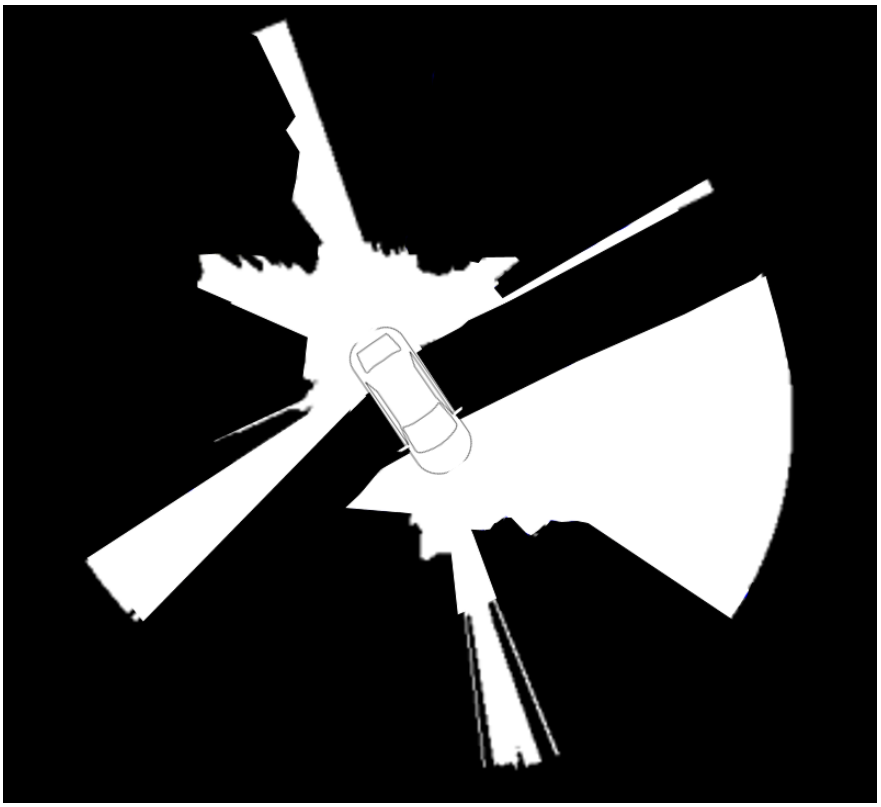


Automotive free space estimation with 4D short-range radar

Master's thesis in System, Control and Mechatronics

Jonathan Falk, William Millqvist

DEPARTMENT OF ELECTRICAL ENGINEERING

CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2022
www.chalmers.se

Master's thesis 2022

Automotive free space estimation with 4D short-range radar

Jonathan Falk, William Millqvist

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Electrical Engineering
Chalmers University of Technology
Gothenburg, Sweden 2022

Automotive free space estimation with 4D short-range radar
Jonathan Falk, William Millqvist

Supervisor: Sara Tell, Volvo Cars
Supervisor: Alexander Åstrand, Volvo Cars
Supervisor: Ahmet Oguz Kislal, Electrical Engineering
Examiner: Erik Agrell, Electrical Engineering

Master's Thesis 2022
Department of Electrical Engineering
Chalmers University of Technology
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Cover: Visualization of free space (in white) around the car using 4D short-range radars.

Gothenburg, Sweden 2022

Automotive free space estimation with 4D short-range radar
Jonathan Falk, William Millqvist
Department of Electrical Engineering
Chalmers University of Technology

Abstract

The development of autonomous driving and advanced driver assistance system in the car industry is progressing fast. Different sensors are used to make predictions and estimations of the area surrounding the car. In this thesis, we investigate how free space estimations in parking scenarios can be made using a new prototype of short-range 4D radars and compare it to estimations made by ultrasonic sensors.

We propose a simulations localization and mapping (SLAM) method combining both the information captured from a single radar scan, together with the difference from two consecutive scans. A dynamic filter using a random sample consensus (RANSAC) algorithm separates the static and dynamic detections. The detections are clustered into landmarks and used in a Gauss-Newton optimized scan matching with outlier rejection. An instantaneous ego-vehicle estimation using the Doppler velocity of the radar detections is used to estimate the velocity and yaw rate (ego-vehicle is referring to the vehicle that contains the sensors). The radar information is then combined with data from an inertial measurement unit (IMU) in an extended Kalman filter, together with a coordinated turn motion model. The mapping of the environment is done with a 2D occupancy grid, which is updated while the vehicle moves through the environment. With the occupancy grid, the free- and occupied space can be classified, by using the field of view for the different sensors. The estimated free space and occupied space are classified as either true or false and a percentage of correct and incorrect classified cells is used as a metric.

The number of detections from the radar is a bit inconsistent over time and our methods faced some problems when the number of detections was low. Otherwise, our methods showed promising results, with accurately estimated trajectories using radar data, and increased robustness when combined with IMU data. The dynamic filter is successfully able to filter out dynamic radar detections, and the ability to do so is a big advantage. Comparing the free space estimations from the ultrasonic sensor and the short-range radar shows that the performance is similar, with a small advantage for the radar. The conclusion that can be drawn is that the radar has more advantages compared to the ultrasonic sensors, with the ability to estimate the trajectory of the vehicle, filter dynamic objects, as well as satisfactory estimate the surrounding free space.

Keywords: 4D short-range radars, Ultrasonic sensors, SLAM, Sensor fusion, Scan matching, Occupancy grid, Free space estimation

Acknowledgements

Firstly, we would like to thank our supervisor at Chalmers, Ahmet Oguz Kislal, for the help, support, and valuable input to the thesis. Thanks also to our examiner Erik Agrell. We would like to thank the Radar and Ultrasonic group at Volvo Cars for the warm welcome. A big thanks to our supervisors at Volvo Cars, Sara Tell and Alexander Åstrand for helping us define the scope, your valuable ideas, and all the support. We would also like to thank Karl Vanäs for the technical input and Joel Salmenius for giving us the opportunity to be a part of the team.

Jonathan Falk, Gothenburg, May 2022
William Millqvist, Gothenburg, May 2022

Nomenclature

Below is the nomenclature of parameters and variables that have been used throughout this thesis. Bold letters are used to indicate vectors or matrices.

Parameters

l_0	Log-odds when no measurements used
N_{RANSAC}	Iterations in RANSAC algorithm
N_{min}	Minimum number of close detections in DBSCAN algorithm
p_{free}	Probability of a free cell
p_{occ}	Probability of a occupied cell
ϵ	Threshold used in RANSAC algorithm
ϵ_{dist}	Distance threshold used in DBSCAN algorithm
ϵ_O	Outlier rejection threshold

Variables

$\mathbf{b}$	Vector describing the ego motion of a vehicle
$\mathbf{C}$	Transformation matrix
c_{light}	Speed of light
c_{sound}	Speed of sound
$\mathbf{d}$	Radar detection
$\mathbf{e}()$	Error function
E	Expected value
$\mathbf{F}$	Linearized motion model
$f()$	Motion model function
$\mathbf{J}$	Jacobian
$\mathbf{H}$	Linearized measurement model
$\mathbf{h}()$	Measurement model
$\mathbf{I}$	Identity matrix
$\mathbf{K}$	Kalman gain
$\mathbf{l}$	Log-odds grid
M	Total number of sensors

$\mathbf{m}$	Grid
$\mathbf{m}^*$	The most likely grid given the sensor data
N	Total number of detections in a radar scan
$\mathbf{P}$	Estimated covariance
p	Probability density function
$\mathbf{Q}$	Covariance of process noise
$\mathbf{q}$	Process noise
$\mathbf{R}$	Rotation matrix
$\mathbf{r}$	Measurement noise
r	Range radar
r_{uss}	Range ultrasonic sensor
$\mathbf{S}$	Innovation covariance
SNR	Signal to noise ratio
$\mathbf{t}$	Translation vector
T	Sampling time
$\mathbf{U}$	Point cloud
$\mathbf{u}$	Point in space (x,y,z)
v	Velocity
v^D	Doppler velocity
$\mathbf{W}$	Covariance of measurement noise
x	x position
$\mathbf{x}$	State vector
$\tilde{\mathbf{x}}$	Vehicle pose (used in occupancy grid)
y	y position
$\tilde{\mathbf{y}}$	Innovation
z	z position
$\mathbf{z}$	Measurements
α	Angle from center of vehicle to the radar placement
γ	Roll angle
θ	Azimuth angle (yaw angle)
μ	Mean
σ	Standard deviation
Σ	Covariance
τ	Time-of-flight
ϕ	Elevation angle (pitch angle)
ω	Yaw rate

Contents

List of Figures

List of Tables

1

Introduction

One of the most interesting research areas today is the progression of autonomous vehicles. Cars are becoming more and more advanced and much of the development focuses on self-driving features. For advanced driver assistance systems (ADAS) and autonomous drive (AD), it is crucial to have a reliable perception of the environment around the car. Different sensors are used to collect data, which is used to make estimations and predictions of the surroundings. Cameras, lidars, radars, and ultrasonic sensors (USS) are some of the sensors used in vehicle navigation which all have different benefits and drawbacks.

A classic radar will determine an object's location in range, azimuth angle, and relative speed (also known as the Doppler velocity). Recently, groundbreaking new 4D radars have been developed that also measure the elevation angle to the object, increasing the dimensions of detected objects from the 2D plane to 3D space. This broadens the field of use for the radar, making it a strong option compared to other sensors. A prototype of a 4D radar with a short maximum range, high accuracy, and high resolution is analyzed in this thesis.

An autonomous vehicle needs to know the environment around the car, and where it is located in the environment. Updating a map at the same time as localizing the vehicle in the map is known as the simultaneous localization and mapping (SLAM) problem. The map can then be used to classify areas that are free from objects, also known as drivable space or free space. The free space can then be used for example in path planning or emergency braking.

The two ADAS areas with the most current development are functions used at low speed and functions used at high speed. Low-speed scenarios include for example parking, while high-speed scenarios consist of highway driving, among others. Ultrasonic sensors are mostly used in low-speed scenarios for short-range functions such as parking assistance, blind-spot detection, and free space estimation. In this thesis, we will investigate if this free space estimation can be done with the advanced short-range 4D radar, and we will compare the performances of the different types of sensors.

1.1 Related work

SLAM is a classical problem in the field of autonomous navigation. The three main approaches to SLAM are Bayesian filters (Kalman filters), particle filters, and graph-based approaches discussed in [1, 2, 3]. The use of 4D radars in SLAM is limited and no publication about it could be found. When it comes to other automotive radars there is however a lot of information. Four short-range radars are used in [4] together with a particle filter and a 2D occupancy grid. An odometer is used to update how the vehicle moves and the results are robust and accurate. The paper presented in [5] uses a feature- extraction, and matcher for radar SLAM using landmarks. Graph optimization is used for optimizing the map. An iterative closest point (ICP) scan matching is used together with odometry in [6] with high performance. Instead of using landmarks or all radar detections, this paper creates sub-maps that contain multiple scans and then uses a feature-based matching.

Authors of [7] and [8] present an ego-vehicle estimation using both one and multiple radars (ego-vehicle is referring to the vehicle that contains the sensors). Stationary targets are detected using the Doppler velocity and the best fit, using a RANSAC algorithm, is used to estimate the velocity and yaw rate of the vehicle. In [9], a method of estimating the ego-motion using long-range radars, key point extraction, and graph matching is presented.

When it comes to SLAM using ultrasonic sensors (USS), only a few papers can be found. A particle filter SLAM algorithm, using only 2D range USS is described in [10]. This is however using four fixed features with known locations and a mobile robot moving inside the area surrounded by the fixed features. A lidar-based SLAM implementation is presented in [11]. Lidar data is used in an ICP-based scan matching algorithm and the transformations are used as the measurement update in a Kalman filter.

Regarding free space estimation, [12] investigates a high-resolution radar to estimate the free space with 2D occupancy grid mapping. The amplitude of the measurement is taken into account, as well as clustering for neighboring cells and estimating borders of objects to refine the grid. The free space was successfully estimated and updated over time. Authors of [13] propose a compact free space representation using a so-called parametric free space map. Their paper also contains a thorough analysis of the advantages and disadvantages of different methods and metrics for representing automotive environments.

For comparison between different SLAM algorithms, a metric is proposed in [14] based on the relative displacements between poses. The downside is that some pre-processing is needed, like measuring the distance between the start and end of the trajectory. In [15], a method for evaluating automotive occupancy grid maps is presented. Using a map score and percentage of the free and occupied cells, different methods are analyzed, but no convincing conclusion has been drawn.

1.2 Purpose

The purpose of this thesis is to investigate how well free space can be estimated with data available from 4D short-range radars and compare its performance with ultrasonic sensors. This is done since new advanced radars have emerged that could result in increased performance compared to the sensors used today. The field of use for these new radars has only been investigated narrowly and could potentially lead to better automobile perception. This means higher safety and more advanced driver assistance functions.

1.3 Research questions

The main objective of the thesis is to estimate the free space surrounding the car as accurately as possible. Different methods will be investigated, for both radar and USS, to see how the estimation could be performed and what is the state-of-the-art. The most promising methods will be implemented and sensor data from Volvo Cars will then be used to validate these methods. How to compare and evaluate free space estimations will be investigated as well.

It is possible to make estimations of the surrounding free space without incorporating any SLAM algorithm, but to get a more robust result, SLAM will be used. This is to be able to use information captured at previous time steps. The main research questions can be summarized in the following parts.

- What is the state-of-the-art when it comes to free space estimation for both short-range radars and ultrasonic sensors?
- How can these state-of-the-art methods be implemented given the data and sensors available?
- How does one compare free space estimations from the radar and the USS, when the sensors have different properties?
- How does the new short-range radar estimation compare to the existing solution with ultrasonic sensors? How can the solution be evaluated if it is accurate and robust enough to be used in cars? What are the strengths and weaknesses of the different sensors?

1.4 Limitations

Since the radars and the USS have limited range (20 meters for the radar and 5 meters for the USS), low-speed scenarios, such as parking sequences in relatively small areas will be of focus. The low speed together with a high sampling rate will result in a detailed map of the environment, which would not be possible at high speed due to the limited range.

The new short-range radar is being tested in the United States and is not available in Gothenburg, which means that the data that will be used is collected with a car in the US. This means that there will be a limited amount of data and a limited

possibility to collect new data. The methods implemented cannot be tested online in a real car for the same reason, but will be evaluated offline on collected data. The computational time for the algorithm is not of high importance, since optimizing code could be very time-consuming. Instead, the thesis will focus on how the algorithm can obtain the most accurate and robust results possible.

The algorithm for the radar does necessarily not need to be the same as for the USS. Since the different sensors have different properties, it may be better to use different algorithms. There will not be any work done by combining the different sensors to estimate the free space.

1.5 Thesis outline/Disposition

In Chapter 2 the background theory is presented. It includes information about sensors and all the basic concepts. Chapter 3 describes the methods used for the different sensors and how the comparison was performed. Chapter 4 presents the results and Chapter 5 contains discussions about the methods used and the results. Chapter 6 presents the conclusions of the thesis, which consists of a concise analysis of the project.

1.6 Contributions

The major contributions of the thesis can be summarized as a method of performing SLAM on 4D radar data and a fair method of comparing free space estimations from different sensors when no clear ground truth exists. Our method of performing radar SLAM utilizes both the Doppler velocity captured in a single time step as well as the difference in position of detections between multiple time steps, to estimate the position, velocity, heading, and angular velocity. We propose a scan matching algorithm with outlier rejection using landmarks instead of all radar detections, which improves performance. By modeling the area where our data was collected, using Google maps satellite view and camera images, the environment can be reconstructed as it was on the day of the testing. Converting the generated map to an occupancy grid makes it possible to classify correctly and incorrectly labeled free and occupied space. This makes it possible to compare free space estimations across different sensors, no matter the range or the number of sensors.

1.7 Ethical and sustainability aspects

In the field of autonomous vehicles, there are a lot of ethical aspects. Some of the major topics are concerning safety, responsibility, and the attitude of the public toward self-driving vehicles [16]. The development of autonomous driving is moving rapidly with new features and more advanced driver assistance systems that make the vehicle less and less dependent on the driver. These new functions need to be thoroughly tested since small errors in real traffic situations can lead to accidents,

which could cause serious injuries or even take human lives.

People can over-rely on collision avoidance functions taking dangerous decisions or not paying attention to the traffic. In case of an accident, it is also not sure who is responsible. The person behind the wheel, the company manufacturing the car, or the programmers writing the code for the function. At which level of self-driving does the driver lose the responsibility of the vehicle?

Introducing autonomous vehicles into society will probably have both advantages and disadvantages. More developed and autonomous vehicles could increase the life expectancy of the vehicles. Better fuel management and energy efficiency could also be an aspect that leads to less fuel consumption and a lower climate impact. However, autonomous vehicles will likely lead to higher total mileage driven [16]. The distance driven will also be affected by the distribution between private and rentable vehicles. Autonomous car-pooling could probably decrease the number of vehicles used, eliminating the need for a private car. This would mean that fewer parking lots are needed since the vehicles are out in circulation most of the time. The autonomous driving field is closely related to the electrification of the autonomous industry and using our resources more cleverly will make the world more sustainable and contribute to a better future.

2
Theory

This chapter starts by describing the different types of sensors and then explains the main concepts used in the thesis. From state space models to occupancy grid mapping and scan matching.

2.1 Sensors

The different sensors that are used are radar, USS, and IMU. Both the radar and USS are proximity sensors that measures the range to objects, while the IMU measures the angular rate and acceleration. This section provides a deeper look into the different types of sensors.

2.1.1 Radar

The radar is a sensor that works by radiating electromagnetic waves and detects the returning echo to determine the range and angle to objects in the environment [17]. It uses the time-of-flight from when the signal is emitted to when it returns after reflecting on the target to calculate the range. The time-of-flight τ is related to the range r as [18]

$$\tau = \frac{2r}{c_{light}}, \tag{2.1}$$

where c_{light} is the speed of light. The radar also measures the relative velocity between the object and the sensor, by using the difference in frequency between the transmitted and received signal, also known as the Doppler velocity. If the detected object is moving towards the sensor, the returning wavelength will be shorter and of higher frequency than the transmitted signal which is used to calculate the object's relative velocity to the sensor.

Radars transmit waves in a variety of frequencies ranging from the so-called high frequencies (3-30 MHz) to the so-called millimeter wavelengths (above 40 GHz) [17]. One commonly used radar in the automotive industry is the millimeter-wave radar operating at a frequency of 76 to 81 GHz [19]. The high frequency in the millimeter-wave radar results in high angular accuracy. A sensing technology often used with these short wavelengths is the frequency modulated continuous wave (FMCW). A FMCW radar emits a series of chirps consisting of sinusoids whose frequency changes linearly over time. The FMCW radar consists of a synthesizer, which creates the chirp, a transmitting antenna that transmits the chirp, a receiving antenna that

receives the signal, and a mixer that combines the sent and received chirp and generates a new signal. The direction (azimuth angle) to the detected object is given by the phase shifts between the horizontally placed antennas.

One of the main advantages of radar technology is its robustness. The radio waves are less affected by extreme temperatures, bad lighting, or weather conditions such as rain or fog, compared to other sensors [19]. The waves can also penetrate objects and reflect on a target, for example, inside a car. The disadvantages are noisy data and a limited angular resolution of the received signal.

2.1.1.1 4D radar

The most recent development in the automotive radar area is the new 4D radar emerging, adding a new dimension to the traditional radar. Instead of just measuring the range, azimuth, and Doppler velocity of an object, these radars also have an angular resolution in elevation (some traditional radars can get some information about the elevation, but is often very unreliable). This makes it possible to separate targets with the same range in both azimuth and elevation. By arranging the antennas both horizontally and vertically, two (or more) phase centers are created and the elevation can be calculated. Knowing the height of an object or at which elevation an object is located is a great advantage. For example, approaching a tunnel or an underpass of a bridge and knowing that it is possible to drive underneath. Adding the extra dimension makes it comparable to the lidar sensor, for a considerably lower price. The radar used in our vehicle has a shorter range than most radars used in the automotive industry, with a maximum range of 20 meters. The field of view is $\pm 60^\circ$ for both elevation and azimuth.

The notation used in this thesis is as follows. The i'th radar detection is denoted $\mathbf{d}_i$ where $i \in 1, 2, \ldots, N$. In a single scan, the radar measures the range r_i [m], the azimuth angle θ_i [rad] and the elevation ϕ_i [rad] of the object. It also measures the signal to noise ratio SNR_i [dB] and Doppler velocity [m/s] of each detection v_i^D and is concatenated in a vector as

$$\mathbf{d}_i = \begin{bmatrix} r_i & \theta_i & \phi_i & v_i^D & \text{SNR}_i \end{bmatrix}. \tag{2.2}$$

From the range, azimuth, and elevation, a detection can be expressed in Cartesian coordinates as,

$$x_i = r_i cos(\phi_i) cos(\theta_i), \tag{2.3}$$
$$y_i = r_i cos(\phi_i) sin(\theta_i) \tag{2.4}$$
$$z_i = r_i sin(\phi_i). \tag{2.5}$$

The x_i, y_i, and z_i position is used to describe a point $\mathbf{u}_i$ in the coordinate system as,

$$\mathbf{u}_i = \begin{bmatrix} x_i & y_i & z_i \end{bmatrix}^T. \tag{2.6}$$

Multiple points from a single scan at time step k is known as a point cloud $\mathbf{U}_k$, which is defined as

$$\mathbf{U}_k = \begin{bmatrix} \mathbf{u}_1 & \mathbf{u}_2 & \dots & \mathbf{u}_N \end{bmatrix}. \tag{2.7}$$

2.1.2 Ultrasonic sensor

USS is one of the sensors commonly used in the automotive industry. It is a proximity sensor that uses ultrasonic sound waves to find the range to the closest object in the field of view. The sensor consists of a transmitter, which emits a high-frequency sound wave that reflects on the closest object and the echo is registered by the receiver [20]. The range r_{uss} can then be calculated as

$$r_{uss} = \frac{\tau c_{sound}}{2}, \tag{2.8}$$

where τ stands for time-of-flight and c_{sound} is the speed of sound (which in air is about 340 m/s) [21]. The division by two appears since the sound wave travels back and forth to the object.

Since the speed of sound is considerably lower than the speed of light ($\approx$ 300 000 km/s), the USS is best suited for short-range applications. The USS used on our vehicle has a detection range between 0.15 and 5 meters and a horizontal field of view angle of $\pm$ 30 degrees. In the field of view, the range to the closest object is returned no matter what the angle is. This means that the azimuth angle to the object is not measured.

Using multiple USS with overlapping fields of views could increase the performance since a sensor can pick up echoes from a neighboring sensor, a so-called cross echo [22]. This means that a single sensor can detect an echo from in front of the sensor as well as a cross echo from the right and left, as seen in Figure 2.1.

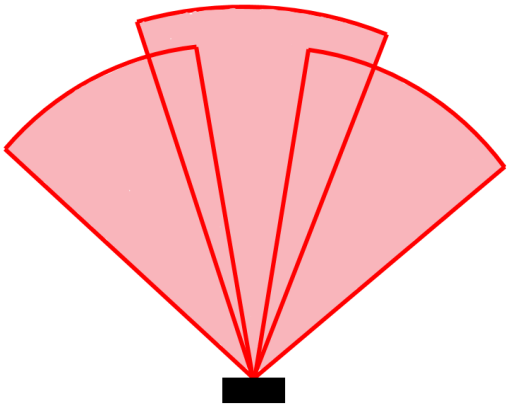


Figure 2.1: The field of view of an ultrasonic sensor. The black is the sensor and the three red areas are the field of view for the three echos.

The USS is a reliable sensor used in both automotive and industrial applications. In the automotive area, USS is used in parking assistance and blind-spot detection.

Some of the main benefits are the ability to detect objects independent of their shape, color, transparency, or lightning conditions. The cost of the sensor is very low and it is robust to external conditions like dust, snow, or rain. Some of the limitations are short-range, and no azimuth angle given to the object. The accuracy of the sensor is also affected by the angle of the object, which could reflect the sound wave in another direction.

2.1.3 IMU

The inertial measurement unit (IMU) sensors are used to measure an object's specific force (acceleration) and angular rate. An IMU consists of a three-axis gyroscope, which measures the angular velocity, and a three-axis accelerometer. Some IMUs contains a magnetometer as well but this is not used in the automotive industry because of the magnetic fields of the vehicle [23]. From the accelerometer, pitch and roll can be estimated, while from the gyroscope, pitch, roll, and yaw can be estimated (pitch is the rotation around the y-axis, roll is the rotation around the x-axis, and yaw around the z-axis). In total, the IMU estimates the 6 degrees of freedom, shown in Figure 2.2, with position (x, y, and z) and orientation (pitch, roll, and yaw) [20].

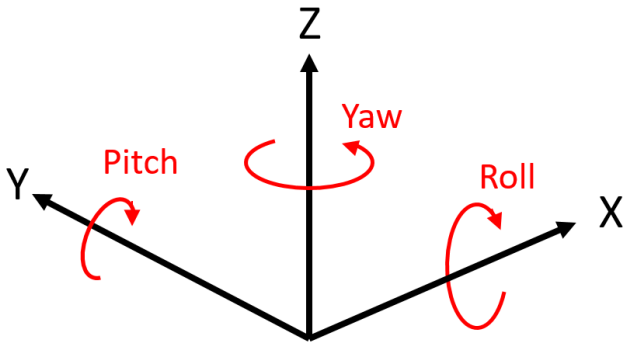


Figure 2.2: Description of the coordinate system and roll (γ), pitch (ϕ), and yaw (θ)

The benefits of IMU are the low power usage, low cost, and small size [23]. The IMU used on the test vehicle has a high sampling frequency compared to the radar and USS (about 10 times higher). The IMU is often used in the automotive industry as the source of information in tunnels and in parking garages, where signals from GPS are not available. One weakness with the IMU is that the gyroscope tends to drift over long periods of time [20]. The position is given by integrating the acceleration twice and any error in the acceleration will lead to a large error in position.

2.2 Sensor setup and coordinate systems

The collected data are sampled with the following sensor setup. Four radars located at the corners of the car are used, which are shown in Figure 2.3 as blue boxes. Twelve ultrasonic sensors are used, where six are placed in the front and six in the back and are presented as green circles. The placement of the sensors is calculated relative to the vehicle frame. The vehicle frame (subscript v) has the origin at the rear axle, presented as x_v, y_v, and z_v, which is positive upwards. The longitudinal velocity is the velocity along the x_v axle and is presented as v_v. θ_v is the angle between the $x_v - z_v$ plane and the direction that the sensor is pointing. ω_v is the angular velocity of θ_v. Each sensor has a sensor frame, and the x-direction is the direction that the sensor is pointing. This will be used to describe where the detections are located relative to the sensor. The coordinate system for the front right radar is presented in Figure 2.3 and is named x_s, y_s, and z_s (subscript s for sensor frame). θ_s is the azimuth angle, which is the angle between the detection and the $x_s - z_s$ plane. The elevation angle ϕ_s is the angle between the detection and the $x_s - y_s$ plane.

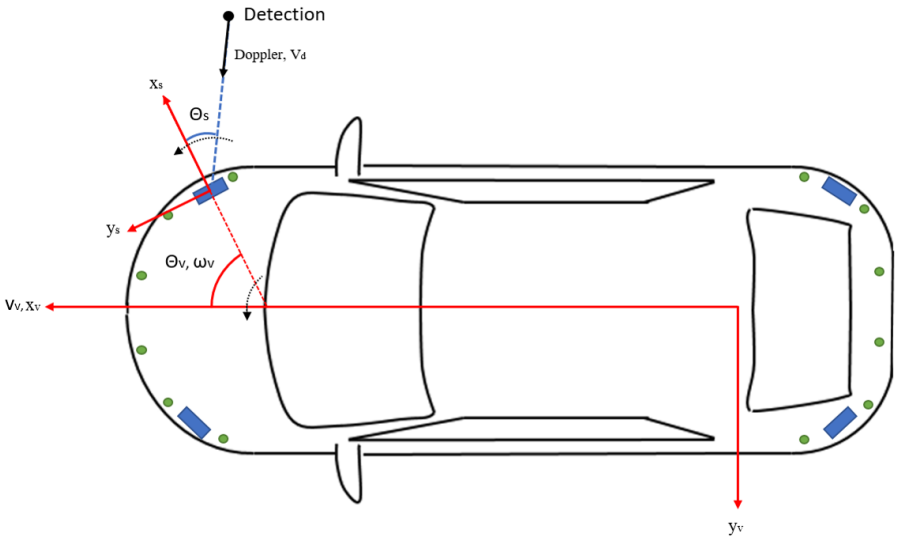


Figure 2.3: The position of the radars and the USS. The green circles are the USS and the green boxes are the radars. Also, the sensor frame (x_s, y_s, θ_s) and vehicle frame $(x_v, v_v, y_v, \theta_v, \omega_v)$ are shown.

To describe the trajectory of the car, a global coordinate system is defined. It is placed with the origin at the initial position of the vehicle frame at the start of each sequence. The axes are defined as x_g, y_g and z_g (subscript g for global frame). The angle between x_g and x_v is the yaw angle θ_g and describes how much the car has turned since the start. The three different coordinate systems, together with the parameters of the global frame, are shown in Figure 2.4.

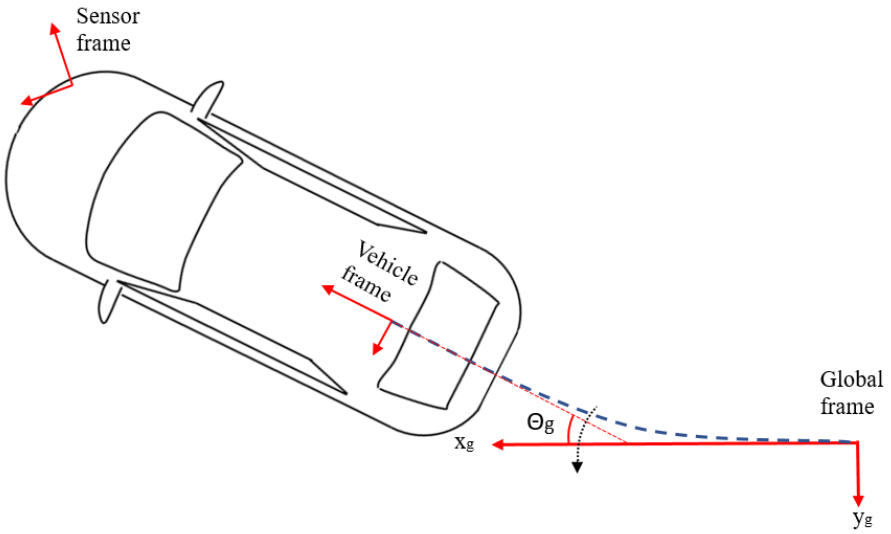


Figure 2.4: An overview of the position of the sensor, vehicle and global frame. The coordinates for the global frame (x_g, y_g, θ_g) is presented and the dotted blue line is the trajectory since the start

2.2.1 Homogeneous transformations

To be able to know where each measurement is relative to the global frame, the detections must first be transformed to the vehicle frame. Since the sensors have both a translation and a rotation, it is suitable to use homogeneous coordinates and transformations. Transforming Euclidean coordinates to homogeneous coordinates allows expressing affine transformations using matrices. The transformation from Euclidean to homogeneous coordinates is done by adding an extra dimension. The transformation from sensor position to vehicle position is shown as

$$\begin{bmatrix} x_v \\ y_v \\ z_v \\ 1 \end{bmatrix} = \mathbf{C}_s^v \begin{bmatrix} x_s \\ y_s \\ z_s \\ 1 \end{bmatrix} = \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0} & 1 \end{bmatrix} \begin{bmatrix} x_s \\ y_s \\ z_s \\ 1 \end{bmatrix}, \tag{2.9}$$

where $\mathbf{C}_s^v$ is the transformation matrix from sensor frame to vehicle frame, and $\mathbf{0}$ is a 1x3 row vector containing zeroes. $\mathbf{R}$ is the rotation matrix and $\mathbf{t}$ is the translation vector, which is shown in equations (2.10)-(2.14). The rotation matrix is dependent on which axes are rotated around and by how much. Rotation can be done around the x, y, and z axes and each rotation has its own rotation matrix. To get the total rotation over multiple axes, it is possible to multiply the matrices with each other. The three rotation matrices are,

$$\mathbf{R}_x(\gamma) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & cos(\gamma) & -sin(\gamma) \\ 0 & sin(\gamma) & cos(\gamma) \end{bmatrix} \tag{2.10}$$

$$\mathbf{R}_y(\phi) = \begin{bmatrix} cos(\phi) & 0 & sin(\phi) \\ 0 & 1 & 0 \\ -sin(\phi) & 0 & cos(\phi) \end{bmatrix} \tag{2.11}$$

$$\mathbf{R}_z(\theta) = \begin{bmatrix} cos(\theta) & -sin(\theta) & 0 \\ sin(\theta) & cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix}, \tag{2.12}$$

and γ, ϕ, and θ are the angles. The total rotation around the x, y, and z axis is given as

$$\mathbf{R} = \mathbf{R}_z\mathbf{R}_y\mathbf{R}_x. \tag{2.13}$$

The translation is the distance in x, y, and z that the point has moved. The $\mathbf{t}$ vector is defined as

$$\mathbf{t} = \begin{bmatrix} x_i \\ y_i \\ z_i \end{bmatrix}. \tag{2.14}$$

When this transformation is done the detections will be in the vehicle frame. When the car moves, it is transformed into the global frame. The transformation matrix from the vehicle frame to the global frame $\mathbf{C}_v^g$ is calculated as the previous transformation, but with other angles and translations. The transformation to the global frame can be done either from the vehicle frame, but also directly from the sensor frame as

$$\begin{bmatrix} x_g \\ y_g \\ z_g \\ 1 \end{bmatrix} = \mathbf{C}_v^g \begin{bmatrix} x_v \\ y_v \\ z_v \\ 1 \end{bmatrix} = \mathbf{C}_v^g\mathbf{C}_s^v \begin{bmatrix} x_s \\ y_s \\ z_s \\ 1 \end{bmatrix}. \tag{2.15}$$

2.3 State space models

A state space model is a mathematical model describing a physical system. The system is defined by two equations, the motion model and the measurement model. The models will be presented for discrete-time steps, where the subscript k stands for the current time step.

2.3.1 Motion model

A motion model describes how the state vector changes over time. Depending on the states at the previous time, the states at the current time can be estimated using the motion model. There are many different types of motion models and the best choice depends on the properties of the vehicle. The general motion model is described by

$$\mathbf{x}_k = \mathbf{f}(\mathbf{x}_{k-1}, \mathbf{q}_{k-1}), \tag{2.16}$$

where the function $\mathbf{f}$ depends on the previous state and an additive white Gaussian noise. $\mathbf{x}_k$ is the current state, $\mathbf{x}_{k-1}$ is the previous state and $\mathbf{q}_{k-1}$ is the previous noise which is distributed with $\mathcal{N}(\mathbf{0}, \mathbf{Q}_{k-1})$.

A suitable motion model for a car is the coordinated turn (CT) model, which assumes constant velocity and yaw rate. The reason why CT is a good motion model for a

car is that the velocity and turning rate change relatively slowly. Our state vector is given by

$$\mathbf{x}_k = \begin{bmatrix} x_k \\ y_k \\ v_k \\ \phi_k \\ \omega_k \end{bmatrix}. \tag{2.17}$$

The x_k and y_k coordinates are the positions in meters relative to the position at time zero. The velocity v_k is the longitudinal velocity of the car in meters per second. The angle ϕ_k is the yaw angle in radians relative to the time zero and the angular velocity ω_k is the yaw rate in radians per second. The discrete CT motion model is described by

$$\mathbf{f}(\mathbf{x}_{k-1}, \mathbf{q}_{k-1}) = \begin{bmatrix} x_k \\ y_k \\ v_k \\ \phi_k \\ \omega_k \end{bmatrix} = \begin{bmatrix} x_{k-1} + \frac{2v_{k-1}}{\omega_{k-1}} sin(\frac{\omega_{k-1}T}{2}) cos(\phi_{k-1} + \frac{\omega_{k-1}T}{2}) \\ y_{k-1} + \frac{2v_{k-1}}{\omega_{k-1}}\ sin(\frac{\omega_{k-1}T}{2}) sin(\phi_{k-1} + \frac{\omega_{k-1}T}{2}) \\ v_{k-1} \\ \phi_{k-1} + T\omega_{k-1} \\ \omega_{k-1} \end{bmatrix} + \mathbf{q}_{k-1}, \tag{2.18}$$

where T is the sampling time and the noise covariance matrix for this model is given as

$$\mathbf{Q}_{k-1} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & T\Sigma_v & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & T\Sigma_\omega \end{bmatrix}. \tag{2.19}$$

The variable Σ_v is the variance of the longitudinal velocity and Σ_ω is the variance of the yaw rate.

2.3.2 Measurement model

The measurement model describes the relation between the measurement and the states. For the remainder of the thesis, we will use $\mathbf{z}_{k,source}$ to describe the measurements, where k is the time step, and *source* is the source that the measurements come from. In a similar fashion, $\mathbf{r}_{k,source}$ is the measurement noise which is distributed with $\mathcal{N}(\mathbf{0}, \mathbf{W}_{k,source})$ and $\mathbf{h}(\mathbf{x}_k, \mathbf{r}_{k,source})$ is the measurement model. Note that

$$\mathbf{z}_{k,source} = \mathbf{h}(\mathbf{x}_k, \mathbf{r}_{k,source}). \tag{2.20}$$

The measurements that are used from the IMU are the longitudinal velocity v_k and the yaw rate ω_k. This results in a linear measurement model, which is

$$\mathbf{z}_{k,IMU} = \begin{bmatrix} v_k \\ \omega_k \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \mathbf{x}_k + \mathbf{r}_{k,IMU}, \tag{2.21}$$

where the subscript IMU shows that the measurements are obtained from IMU. Another type of measurement that will be used for the radars is the estimation from scan matching, explained later in Section 2.8, and the INED, explained later in Section 2.9. From scan matching, the position and yaw angle of the car can be estimated, and from INED the velocity and yaw rate can be estimated. The measurement model for the radar is then

$$\mathbf{z}_{k,radar} = \begin{bmatrix} x_k \\ y_k \\ v_k \\ \phi_k \\ \omega_k \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \mathbf{x}_k + \mathbf{r}_{k,radar}, \tag{2.22}$$

where subscript *radar* shows that the measurements are obtained from scan matching. These two types of measurement can be used separately to estimate the states. However, if both are used together, the different sampling rates need to be taken into consideration.

2.4 Kalman filter

A Kalman filter is used to estimate the states given the measurements [24]. It is a filter, based on Bayesian statistics. The linear Kalman filter consists of two main steps, the prediction step, and the update step. The prediction step estimates what the states should be in the next time step, given the previous states. It also estimates the uncertainty of the prediction of the states. When the prediction step is done, it uses the measurements in the current time step in the update step, to improve the estimation. The filter calculates the updated state by using a weighted average between the measurements and the predictions. The weighted average is calculated from the uncertainty of both estimations. The Kalman filter works recursively and uses the current estimation and covariance in the next time step. Figure 2.5 shows a block scheme of how the prediction step and the update step are used in the Kalman filter.

Throughout this thesis, we use the notation $\mathbf{x}_{i|j}$ to show the estimated states of $\mathbf{x}$ at i'th time step given the measurements up to the j'th time step. Moreover, to keep the notation simple, we will omit the subscript that shows the source of the data, whenever it is immaterial. For instance, $\mathbf{x}_{k|k-1}$ is the currently estimated states, given measurement up to the previous time step, and the data may be coming from any arbitrary source.

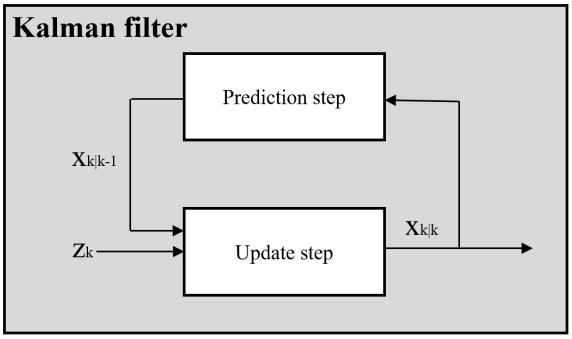


Figure 2.5: The Kalman filter block scheme, which include the prediction and update step.

2.4.1 Extended Kalman filter

The extended Kalman filter is a nonlinear filter that linearizes the models around the current estimate of the mean and covariance of the states [25]. As for the linear Kalman filter, the prediction step is performed first, and then the update step is performed. The equations for the prediction step and the update step are given as,

$$\begin{aligned}
&\textbf{Prediction step} \\
&\mathbf{x}_{k|k-1} = \mathbf{f}(\mathbf{x}_{k-1|k-1}, \mathbf{q}_{k-1}) \\
&\mathbf{P}_{k|k-1} = \mathbf{F}_k \mathbf{P}_{k-1|k-1} \mathbf{F}_k^T + \mathbf{Q}_k
\end{aligned} \tag{2.23}$$

$$\begin{aligned}
&\textbf{Update step} \\
&\tilde{\mathbf{y}}_k = \mathbf{z}_k - \mathbf{h}(\mathbf{x}_{k|k-1}, \mathbf{r}_{k,source}) \\
&\mathbf{S}_k = \mathbf{H}_k \mathbf{P}_{k|k-1} \mathbf{H}_k^T + \mathbf{W}_{k,source} \\
&\mathbf{K}_k = \mathbf{P}_{k|k-1} \mathbf{H}_k^T \mathbf{S}_k^{-1} \\
&\mathbf{x}_{k|k} = \mathbf{x}_{k|k-1} + \mathbf{K}_k \tilde{\mathbf{y}}_k \\
&\mathbf{P}_{k|k} = (\mathbf{I} - \mathbf{K}_k \mathbf{H}_k) \mathbf{P}_{k|k-1},
\end{aligned} \tag{2.24}$$

and all variables not previous defined are presented in Table 2.1.

Table 2.1: Variables used in the extended Kalman filter.

Variable	Description
$\mathbf{P}$	Estimated covariance
$\mathbf{F}$	Linearized motion model
$\mathbf{Q}$	Covariance of process noise
$\mathbf{W}$	Covariance of measurement noise
$\tilde{\mathbf{y}}$	Innovation
$\mathbf{H}$	Linearized measurement model
$\mathbf{S}$	Innovation covariance
$\mathbf{K}$	Kalman gain
$\mathbf{I}$	Identity matrix

The Jacobians $\mathbf{F}$ and $\mathbf{H}$ are calculated by linearizing the motion model and the measurement model. The linearization of the models is done around the estimated states. The equations of these matrices are,

$$\begin{aligned}\mathbf{F}_k &= \left.\frac{\partial \mathbf{f}}{\partial \mathbf{x}}\right|_{\mathbf{x}_{k-1|k-1}} \\ \mathbf{H}_k &= \left.\frac{\partial \mathbf{h}}{\partial \mathbf{x}}\right|_{\mathbf{x}_{k|k-1}}.\end{aligned} \tag{2.25}$$

2.5 Occupancy grid mapping

Occupancy grid mapping is a method to classify if a location is occupied by an object or not in a static environment. A rectangular, absolute orientation grid is created, and by using the detections, the probability that each cell in the grid is occupied can be calculated. The variable $\mathbf{m}$ represents a grid, which is a vector of cells. Each cell represents the probability that a specific area in the environment, relative to a fixed point, is occupied by an object. The main task is to compute the probability of each cell being occupied given the sensor data and is described by

$$\mathbf{m}^* = \underset{\mathbf{m}}{\operatorname{argmax}}\ P(\mathbf{m}|\tilde{\mathbf{x}}_{0:k}, \mathbf{z}_{0:k}), \tag{2.26}$$

where $P()$ is a probability mass function and $\tilde{\mathbf{x}}_{0:k}$ is the pose, which is the x and y coordinates and the yaw angle of the vehicle. The subscript $0:k$ stands for the time steps from 0 to k. This means that the most likely map at the current time $\mathbf{m}^*$ shall be found given the measurements and the pose of the vehicle at all time steps up to the current time. Occupancy grids can be divided into two categories, feature-, and grid-based representation. Feature-based representation describes the environment by describing the positions of landmarks that are found. Grid-based representation defines a grid where each cell represents a part of the environment. Each cell is assumed to be occupied by an object or marked as free space and contains the probability of this. In this thesis, the grid-based representation will be used. Occupancy grid is sometimes called mapping with known poses since the pose is assumed to be known. To make our analysis tractable, the following assumptions were made:

1. A cell is assumed to be either occupied or free
2. The cells are independent of each other
3. The environment is static

The estimation of the map as given in (2.26) is a joint probability between all cells. We assume cells are independent so the joint probability in (2.26) may be simplified as

$$P(\mathbf{m}|\tilde{\mathbf{x}}_{0:k}, \mathbf{z}_{0:k}) = \prod_n P(m_n|\tilde{\mathbf{x}}_{0:k}, \mathbf{z}_{0:k}), \tag{2.27}$$

where m_n is the probability of cell n being occupied. To make it easier to update the map, the probability in (2.27) will be transformed to a log-odds probability. The

equation for calculating the log-odds $l_{(k,n|0:k)}$ from the probability is

$$l_{(k,n|0:k)} = \log\left(\frac{P(m_n|\tilde{\mathbf{x}}_{0:k}, \mathbf{z}_{0:k})}{1 - P(m_n|\tilde{\mathbf{x}}_{0:k}, \mathbf{z}_{0:k})}\right), \tag{2.28}$$

which can be positive or negative. It is used because the log-odds ratio is an additive quantity, which means that we may add them in a grid to get the total log-odds ratio over time. To update a cell with new measurements,

$$l_{(k,n|0:k)} = l_{(k,n|k)} + l_{(k-1,n|0:k-1)} - l_0, \tag{2.29}$$

will be used [26]. It was derived using the static state binary filter, which was the reason why assumption 1 and 2 was made. $l_{(k-1,n|0:k-1)}$ is the recursive term, which is the cell n from the previous time step given measurements up to the previous time step. l_0 is the prior, which is the probability of it being occupied without using any measurements and is described by

$$l_0 = \log\frac{\Pr(m_n = 1)}{1 - \Pr(m_n = 1)}, \tag{2.30}$$

where $\Pr(m_n = 1)$ is the probability of cell m_n being occupied. $l_{(k,n|k)}$ is the log-odds probability of a cell being occupied given the current measurements and is calculated as

$$l_{(k,n|k)} = \log\left(\frac{P(m_n|\tilde{\mathbf{x}}_k, \mathbf{z}_k)}{1 - P(m_n|\tilde{\mathbf{x}}_k, \mathbf{z}_k)}\right). \tag{2.31}$$

This equation is also known as the "inverse sensor model" and is a function used to describe the probability of a cell being occupied using only the measurement and pose at the last time step. It means that if a cell is between the sensor and an object, the space is probably free and gets the probability p_{free}, which is between 0 and 0.5 If the cell is where an object was measured the space is probably occupied and gets the value p_{occ}, which is between 0.5 and 1. To calculate the cells that are between the object and the sensor, Bresenham's line algorithm was used [27]. Bresenham's line algorithm takes the position of two cells as input and draws a line between these two cells. Then it calculates the cells that together make up this line in a grid and return a vector of the included cells. In our case, the input to Bresenham's line algorithm will be the sensor position and the position of a detection.

Figure 2.6 shows how the inverse sensor model sets the values of the cells. Each cell gets a log-odds probability and all cells create the log-odds grid $\mathbf{l}_{(k|k)}$. The black cells to the right are the positions of the detections from the sensor and have the value p_{occ}. The cells between the sensor and the detections have the value p_{free}, which is shown as white cells. The gray cells are not seen by the sensor and therefore got a probability of 0.5. These probabilities are then transformed into log-odds ratio, which is then returned and used for updating the cell in equation (2.29).

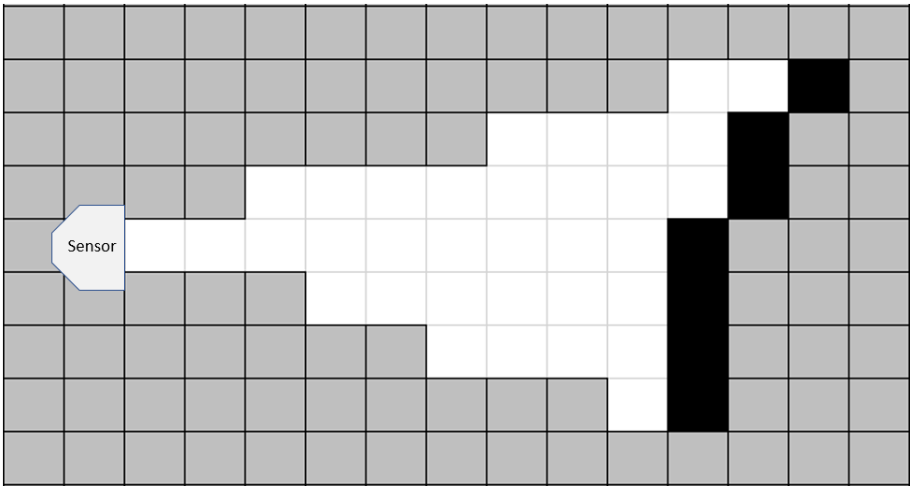


Figure 2.6: An example of how the inverse sensor model works.

When multiple time steps are added and the grid $\mathbf{l}_{(k|0:k)}$ is calculated, the probability of the cells will have different values between 0 and 1 and probabilities will be updated. This grid can then be used to create a binary occupancy grid, where the cells have the values 0 or 1. Since the probability of 0.5 gets a log-odd of 0, the binary grid is calculated by selecting cells with log-odds either over or under 0. For example, to get the binary occupancy grid free from objects, the cells that are under 0 in the log-odds grid are chosen. Updating the map at the next time step is done by iterating through all cells in the grid and if a cell is in the perceptual field of the sensor, it is updated using (2.31). If the cell is outside the perceptual field of the sensor, it stays the same as the previous time step.

2.6 Outlier detection

The raw sensor data will contain both outliers and some noise. This section will explain the theory behind the algorithm used to remove these outliers.

2.6.1 RANSAC

A method of separating inliers and outliers from data is the random sample consensus (RANSAC) algorithm. The algorithm is an iterative process and works by randomly selecting a small number of points. The number of points is selected to be the minimal number of points needed to model the parameters. A model is then fitted to these points and all the points within a certain predefined threshold ϵ are treated as inliers. The number of inliers and parameters is saved, and the process is then repeated N_{RANSAC} times. The iteration with the highest number of inliers is chosen as the best fit [28]. The algorithm is effective to remove outliers if the proportion of outliers is small. If the number of outliers is equal to or bigger than the number of inliers, the algorithm will base the fit on the outliers, and not work as intended.

2.7 Clustering

Data association is one of the more challenging problems when it comes to radar data. Given high sparsity and high amount of noise, it is hard to associate points with objects. Clustering is used to find points with similar characteristics and classify them together into larger entities.

2.7.1 DBSCAN

Density based spatial clustering of applications with noise (DBSCAN) [29] is a widely used clustering algorithm. It was presented in 1996 and the algorithm manages to cluster objects in arbitrary shapes and works on large data sets, something that wasn't possible before. It requires little information about the data with only two tuning parameters.

The DBSCAN algorithm works by randomly selecting a point and checking the number of points within a radius threshold ϵ_{dist}. If the number of points inside the range ϵ_{dist} is more than the tuning parameter N_{min}, that point is classified as a core point, and a cluster is formed with all the points within ϵ_{dist}. Then a new arbitrary point is chosen until all have been visited. A point part of a cluster that is not a core point is called density-reachable, and the points not part of any cluster is classified as noise [30].

2.8 Scan matching

A radar scan $\mathbf{U}_k = [\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_n]$ contains all the points $\mathbf{u}_i = [x_i, y_i, z_i]^T$ given by the radar at a single time step. Given another scan $\mathbf{U}_{k-1} = [\tilde{\mathbf{u}}_1, \tilde{\mathbf{u}}_2, \ldots, \tilde{\mathbf{u}}_m]$, with $\tilde{\mathbf{u}}_i = [x_i, y_i, z_i]^T$ (tilde is used to be able to separate the two point clouds) it is possible to match these scans and get a rotation matrix $\mathbf{R}$ and translation vector $\mathbf{t}$, describing the transformation. If the point cloud is noisy and contains clutter the match will not be perfect, and the goal is to get as good of a match as possible. There are a few different methods of performing scan matching, including map-to-map, iterative closest point, iterative closest line, and iterative closest plane. Since radar data contains points describing the unknown environment, ICP is the most reasonable method.

2.8.1 Iterative closest point

The most common method of ICP is to use singular value decomposition (SVD) to calculate the rotation matrix $\mathbf{R}$ and translation vector $\mathbf{t}$. The first step is to compute the correspondences between the two point clouds by iterating through the first cloud and finding the nearest neighbor in the second cloud. Then, the center

of mass for the two clouds is calculated as

$$\boldsymbol{\mu}_{u_k} = \frac{1}{n}\sum_{i=1}^{n}\mathbf{u}_i \tag{2.32}$$

$$\boldsymbol{\mu}_{u_{k-1}} = \frac{1}{m}\sum_{i=1}^{m}\tilde{\mathbf{u}}_i. \tag{2.33}$$

The cross-covariance between the corresponding points is calculated as

$$\text{cov}(\boldsymbol{\mu}_{u_k}, \boldsymbol{\mu}_{u_{k-1}}) = E[(\mathbf{u}_i - \boldsymbol{\mu}_{u_k})(\tilde{\mathbf{u}}_i - \boldsymbol{\mu}_{u_{k-1}})^T]. \tag{2.34}$$

Using SVD, the cross-covariance matrix can be decomposed according to

$$\text{SVD}(\text{cov}(\boldsymbol{\mu}_{u_k}, \boldsymbol{\mu}_{u_{k-1}})) = \mathbf{USV}^T. \tag{2.35}$$

The rotation and translation can then be calculated as

$$\mathbf{R} = \mathbf{UV}^T, \tag{2.36}$$

$$\mathbf{t} = \boldsymbol{\mu}_{u_{k-1}} - \mathbf{R}\boldsymbol{\mu}_{u_k}. \tag{2.37}$$

With $\mathbf{R}$ and $\mathbf{t}$ the point cloud $\mathbf{U}_k$ is transformed. New correspondences are then computed between all the points in the same two clouds, where $\mathbf{U}_k$ now has been transformed. The process of calculating a new $\mathbf{R}$ and $\mathbf{t}$ matrix and transforming the point cloud is repeated until the correspondences are the same as in the previous iteration.

Instead of using SVD, the correspondences could be optimized using Gauss-Newton´s method [31]. This makes is possible to remove outliers, which means that we do not use the correspondences with a large distance. This is done by minimizing the mean square error of the distances between the correspondences with the error function

$$\mathbf{e}_i(\mathbf{R}, \mathbf{t}) = \mathbf{R}\tilde{\mathbf{u}}_i + \mathbf{t} - \mathbf{u}_i. \tag{2.38}$$

Since $\mathbf{t}$ is dependent on x, y, and z, and $\mathbf{R}$ is dependent on θ, the error function can be expressed as $\mathbf{e}_i(\bar{\mathbf{x}})$, where $\bar{\mathbf{x}} = [x, y, z, \theta]$ and the Jacobian is in this case equal to

$$\mathbf{J}_i(\bar{\mathbf{x}}) = \begin{bmatrix} \frac{\partial \mathbf{e}_i(\bar{\mathbf{x}})}{\partial x} & \frac{\partial \mathbf{e}_i(\bar{\mathbf{x}})}{\partial y} & \frac{\partial \mathbf{e}_i(\bar{\mathbf{x}})}{\partial z} & \frac{\partial \mathbf{e}_i(\bar{\mathbf{x}})}{\partial \theta} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & -sin(\theta)x - cos(\theta)y \\ 0 & 1 & 0 & cos(\theta)x - sin(\theta)y \\ 0 & 0 & 1 & 0 \end{bmatrix} \tag{2.39}$$

With the Jacobian and error function over all detections, the optimal $\Delta\bar{\mathbf{x}}^*$ of our state is, according to [31], given as

$$\Delta\bar{\mathbf{x}}^* = -(\sum_i(\mathbf{J}_i(\bar{\mathbf{x}})^T\mathbf{J}_i(\bar{\mathbf{x}})))^{-1}\sum_i(\mathbf{J}_i(\bar{\mathbf{x}})^T\mathbf{e}_i(\bar{\mathbf{x}})). \tag{2.40}$$

Then our state is updated as

$$\bar{\mathbf{x}} \leftarrow \bar{\mathbf{x}} + \Delta\bar{\mathbf{x}}^*, \tag{2.41}$$

and if the initial point is close to the optimal result, $\bar{\mathbf{x}}$ will converge after iterating.

2.9 Instantaneous ego-motion estimation using Doppler

In [8], the instantaneous method of estimating the ego-motion of a vehicle using multiple radars, abbreviated as INED, is described. This state-of-the-art method makes it possible to estimate the velocity and yaw rate in one single time step. Thus, throughout this section we drop the subscript k to show the time instance. The system motion equations, relating the radar data to the vehicle ego-motion (v_x, v_y and ω) are derived as follows. v_x is the velocity in the x direction which is the same as v used in the state vector in the Kalman filter. The relative velocity of a stationary target is equal to the inverse sensor velocity vector. Given detection number $i \in 1, 2, \ldots, N$, sensor number $j \in 1, 2, \ldots, M$, and the angle between the center of the vehicle frame and the sensor α_j, a linear combination can be constructed as

$$\underbrace{\begin{bmatrix} -v_1^D \\ \vdots \\ -v_N^D \end{bmatrix}}_{\mathbf{v}_j^D} = \underbrace{\begin{bmatrix} cos(\theta_{1,s} + \alpha_j) & sin(\theta_{1,s} + \alpha_j) \\ \vdots & \vdots \\ cos(\theta_{N,s} + \alpha_j) & sin(\theta_{N,s} + \alpha_j) \end{bmatrix}}_{\mathbf{M}_j} \underbrace{\begin{bmatrix} v_{j,x} \\ v_{j,y} \end{bmatrix}}_{\mathbf{v}_j}, \tag{2.42}$$

where $\theta_{i,s}$ is the azimuth angle from sensor frame to detection i, and $v_{j,x}$ and $v_{j,y}$ is the velocity for sensor j in x and y direction. Using the Ackermann condition (no wheel slip at rear axis), (2.42) can then be written for all sensors as

$$\begin{bmatrix} \mathbf{v}_1^D \\ \mathbf{v}_2^D \\ \vdots \\ \mathbf{v}_M^D \end{bmatrix} = \begin{bmatrix} \mathbf{M}_1 & \mathbf{0} & \ldots & \mathbf{0} \\ \mathbf{0} & \mathbf{M}_2 & \ldots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \ldots & \mathbf{M}_M \end{bmatrix} \begin{bmatrix} \mathbf{v}_1 \\ \mathbf{v}_2 \\ \vdots \\ \mathbf{v}_M \end{bmatrix}. \tag{2.43}$$

The velocity of sensor j ($v_{j,x}$, $v_{j,y}$) is

$$\underbrace{\begin{bmatrix} v_{j,x} \\ v_{j,y} \end{bmatrix}}_{\mathbf{v}_j} = \underbrace{\begin{bmatrix} -y_j & 1 & 0 \\ x_j & 0 & 1 \end{bmatrix}}_{\mathbf{S}_j} \underbrace{\begin{bmatrix} \omega \\ v_x \\ v_y \end{bmatrix}}_{\mathbf{b}}, \tag{2.44}$$

where x_j and y_j is the position of the j'th sensor. The velocity of all sensors is then defined as

$$\underbrace{\begin{bmatrix} v_{1,x} \\ v_{1,y} \\ \vdots \\ v_{M,x} \\ v_{M,y} \end{bmatrix}}_{[\mathbf{v}_1^T \ldots \mathbf{v}_M^T]^T} = \underbrace{\begin{bmatrix} -y_1 & 1 & 0 \\ x_1 & 0 & 1 \\ \vdots & \vdots & \vdots \\ -y_M & 1 & 0 \\ x_M & 0 & 1 \end{bmatrix}}_{[\mathbf{S}_1^T \ldots \mathbf{S}_M^T]^T} \underbrace{\begin{bmatrix} \omega \\ v_x \\ v_y \end{bmatrix}}_{\mathbf{b}}. \tag{2.45}$$

Combining (2.43) and (2.45) results in the final system motion equation,

$$\underbrace{\begin{bmatrix} v_1^D \\ v_2^D \\ \vdots \\ v_M^D \end{bmatrix}}_{\mathbf{V}^D} = \underbrace{\begin{bmatrix} \mathbf{M}_1\mathbf{S}_1 \\ \mathbf{M}_2\mathbf{S}_2 \\ \vdots \\ \mathbf{M}_M\mathbf{S}_M \end{bmatrix}}_{\mathbf{A}} \underbrace{\begin{bmatrix} \omega \\ v_x \\ v_y \end{bmatrix}}_{\mathbf{b}}. \tag{2.46}$$

The right side of the equation can be seen as the expected Doppler velocities as

$$\mathbf{V}_{est}^D = \mathbf{A}\mathbf{b}, \tag{2.47}$$

and by minimizing the Doppler residuals between $\mathbf{V}_{est}^D$ and $\mathbf{V}_D$, all the stationary targets can be estimated. The $\mathbf{b}$ vector is the estimated velocities and angular velocity of the ego-vehicle and using a RANSAC algorithm, the best fit of $\mathbf{b}$ can be determined.

3

Methods

This chapter describes the methods used to estimate the trajectory and the free space, and describes how the comparison between different methods is performed. Because of the different specifications of the sensors, the methods will differ some, using the optimal method for the respective sensor. In Section 3.1, the methods used for the radar are described and in Section 3.2, the methods used for the ultrasonic sensors are described.

3.1 Radar methods

An overview of the workflow for the radar methods is shown in Fig. 3.1. Here, detections are first transformed from the sensor frame to the vehicle frame, as described in Section 2.2.1. The detections are classified into dynamic and static detections where the static detections are filtered and clustered using DBSCAN. The clustered detections are used in the scan matching to connect the point cloud at the current time step to the cloud obtained at the previous time step, which results in a rotation- and translation matrix describing the transformation. From these matrices, the next x, y position, and angle θ_g are given. Using the formulas in Section 2.9, the dynamic and static filtering will result in an estimate of the velocity and the yaw rate. With this, we have measurement updates for every state in (2.17), which are the measurement model in the update step of the extended Kalman filter. A coordinated turn motion model is used in the prediction step. This will also be combined with IMU data to try to improve the estimation of the trajectory. The estimation of trajectory block from Fig. 3.1 is illustrated and described in detail in Fig. 3.4 and Section 3.1.3.

The radar data filtered from outliers and dynamic objects will be added to the estimated pose to create a point cloud map over time. A static occupancy grid will then be updated for each time step, originating from the point cloud map. The dynamic detections will be added to a separate dynamic occupancy grid for just the current time step. These two grids are then merged, and a binary grid is created. From this binary grid, the area containing free space is classified. The free space estimation block from Fig. 3.1 is illustrated and described in detail in Fig. 3.5 and Section 3.1.4.

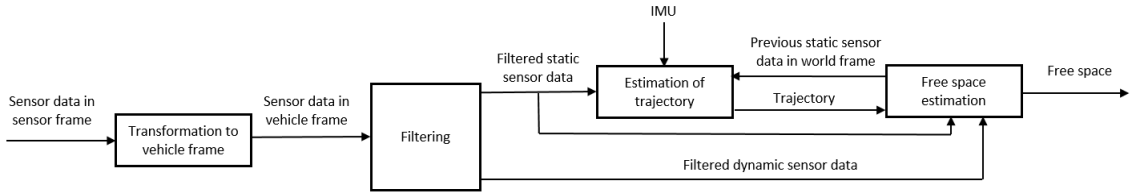
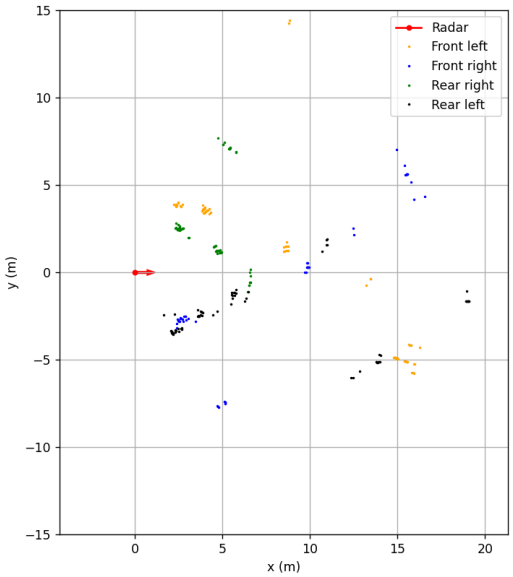


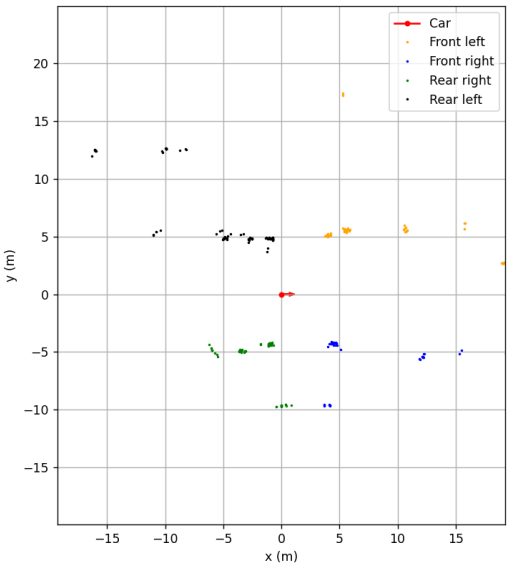
Figure 3.1: Block scheme of the methods used for the estimation using radar.

3.1.1 Transformation to vehicle frame

The measurements from the four radars are transformed to the vehicle frame as described in Section 2.2.1. This is done to see where each measurement is located relative to the car. The transformation used is a translation and a rotation around the z- and y-axis, since the sensors are mounted with a pitch and a yaw angle relative to the vehicle's coordinate system. In Fig. 3.2a the 3D points from all four radars at the same time step are plotted in the sensor frame (in 2D). The different colors represent from which radar each point is detected, and the direction of the red arrow is the x-axis of the sensor frame for all four radars. The same points but in the vehicle frame are shown in Fig. 3.2b. The direction of the red arrow is in this case the x-axis of the vehicle frame.



(a) Point cloud in sensor frame. The red arrow is the x direction of the sensor frame for all four radars.



(b) Point cloud in vehicle frame. The red arrow is the x direction of the vehicle frame.

Figure 3.2: Plots of a point cloud in both sensor and vehicle frame. The different colors represent from which sensor the detection is collected.

3.1.2 Filtering

The radar data contains some noise and outliers. Two major types of filtering will be done to the point cloud in the vehicle frame. First, the dynamic and static objects will be classified and filtered and then the detections are clustered to find objects and remove outliers. The detections are also filtered using SNR. An overview of the filtering is shown in Fig. 3.3.

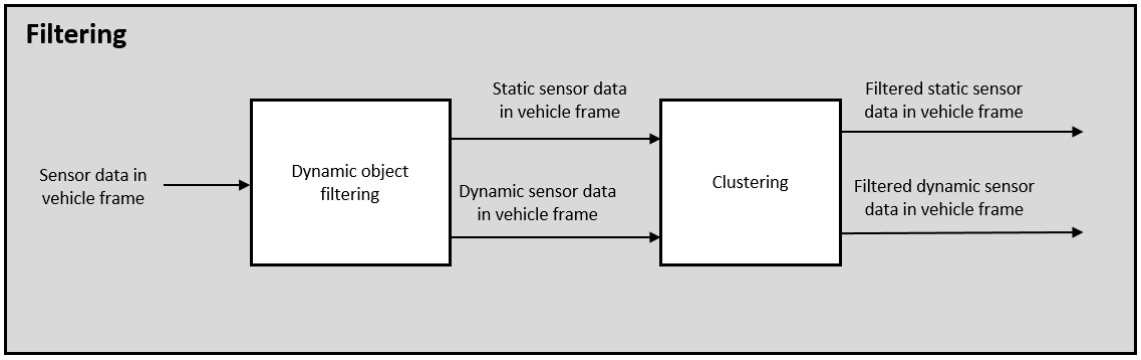


Figure 3.3: A scheme of the filtering block, using radars.

3.1.2.1 Dynamic object filtering

A linear combination of all detections is created as in (2.46). Then a RANSAC algorithm is performed with 1000 iterations. Three random detections from are chosen from minimum two different sensors, and the vector $\mathbf{b} = [\omega \ v_x \ v_y]^T$ is solved for using (2.47). Since all points have different Doppler velocities, the goal is not to maximize the number of inliers. Instead, a cost function is introduced that minimizes the squared residuals of the expected Doppler velocities and real Doppler velocities. The residuals within a given threshold ϵ are chosen as inliers and the score is calculated. The three random points with the lowest score after all iterations are the best estimate of the vector $\mathbf{b}$ and the points outside the threshold are classified as dynamic.

3.1.2.2 Clustering

The points are clustered using a DBSCAN algorithm with different tuning parameters. The different clusters are then analyzed and the mean in x, y, and z is calculated for each cluster. With this mean value of each cluster, it is possible to represent an entire cluster with just one point, a so-called landmark. Representing a point cloud with landmarks instead of detections drastically reduces the number of points used to describe the environment. The change in the position and the number of landmarks make it more stable over time than the positions and the number of detections. Using landmarks in the succeeding scan matching lowers the computational complexity since one of the major contributions is the calculation of the corresponding detections in different time steps. The core and density-reachable points are saved for later point cloud representation, while the noise is discarded.

3.1.3 Estimation of trajectory

The estimation of trajectory is used to connect multiple time steps to be able to create a map. The methods are summarized in Fig. 3.4. The trajectory is estimated using input from the radar methods scan matching and INED, as well as combined with data from the IMU.

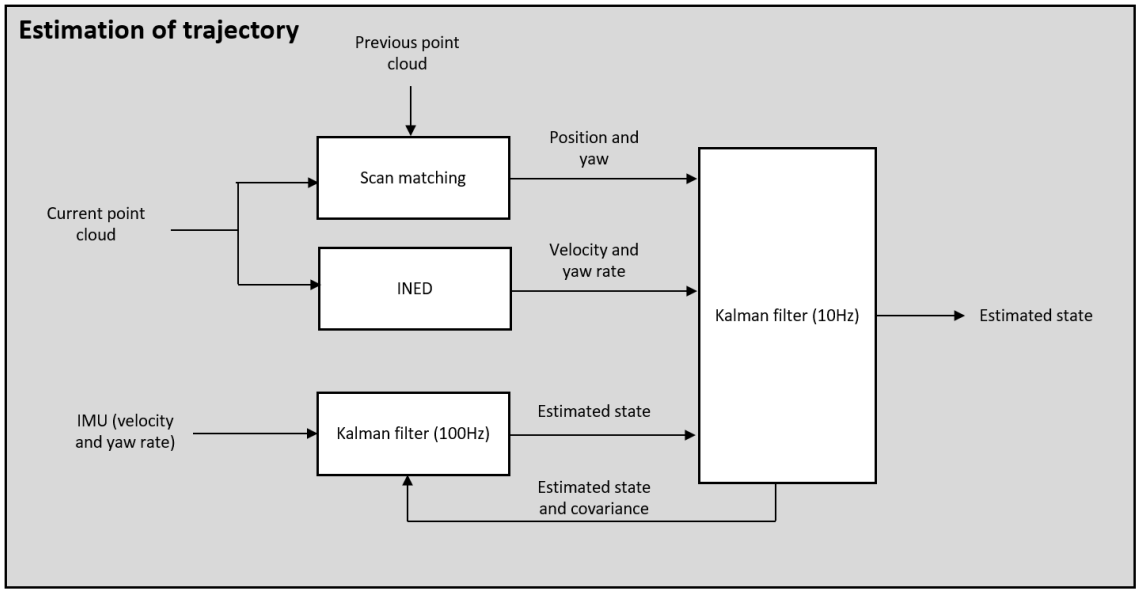


Figure 3.4: A scheme of the estimation of trajectory block, using radars.

3.1.3.1 Scan matching

Scan matching is an algorithm used to match the current radar point cloud, $\mathbf{U}_k$, to a previous time step, $\mathbf{U}_{k-1}$, described in Section 2.8. The transformation can then be used to calculate how much the ego-vehicle has moved in x,y, and yaw, from one time to another. This is done by using the landmarks from the clustering instead of all detections. First, all the correspondences between the two scans are calculated, by iterating through all landmarks in $\mathbf{U}_k$ to find the closest landmark in $\mathbf{U}_{k-1}$. Then we propose an outlier rejection algorithm with a tuning parameter ϵ_O. The outlier rejection works as if the distance between two corresponding landmarks is larger than the threshold distance ϵ_O, these landmarks are omitted from the optimization. The outlier rejection is used since the number of landmarks could be different between two consecutive scans, and a landmark with no close correspondences does not affect the least square problem. A small ϵ_O optimizes the transformation on landmarks close together and while driving at low speed, with a high sampling frequency, the movements between scans are small. The same outlier rejection technique could also be applied using all detections.

After the algorithm has converged, the transformation matrix $\mathbf{C}$ is multiplied by the last position described in homogeneous coordinates, which results in a new position. This new position is used to create the new $\mathbf{C}_o^g$ matrix, describing the transformation from the origin in the global frame. The landmarks are then transformed with this

matrix to the estimated positions (of the landmarks) in the global frame. In the next time step, the new landmarks are transformed with $\mathbf{C}_o^g$ and matched with the previous.

3.1.3.2 Instantaneous ego-motion estimation using Doppler

From the dynamic object filtering described in Section 3.1.2.1, the detections with the best fit will derive from the best estimate of $\mathbf{b} = [\omega \ v_x \ v_y]^T$. The variables ω and v_x exists in our state vector and is later used in our update step in the extended Kalman filter, which increases robustness of our estimated trajectory.

3.1.3.3 Extended Kalman filter

The coordinated turn motion model is used together with a combination of measurements in the update step. One estimation of the trajectory is done with measurements from the two radar methods, scan matching, and INED. One estimation is done with the velocity and the yaw rate from the IMU sensor, and one combining the radar methods and IMU. The difference in the update step of the different combinations is the $\mathbf{z}_k$ vector from (2.24).

Since the IMU has a sampling frequency 10 times higher than the radar, a separate EKF is used for the IMU before fusing with the radar. This EKF when using IMU will be called EKF_{IMU} and the other one EKF_{fuse}. The state and covariance matrix is updated in the EKF_{IMU} from the EKF_{fuse} when radar data is available. The EKF_{IMU} estimates our entire state vector and when combining with the radar methods, which also gives a estimate of the entire state vector, we have multiple measurements of the same state vector. Two different measurement fusion methods for multiple sensors are compared in [32] with similar results. Our approach is the first method which fuses the information as,

$$\mathbf{z}_{k,\text{fuse}} = \begin{bmatrix} \mathbf{z}_{k,IMU}^T & \mathbf{z}_{k,radar}^T \end{bmatrix}^T. \tag{3.1}$$

The tuning for the different cases is done as follows. The $\mathbf{Q}$ matrix is as in (2.19) and the $\mathbf{W}_k$ matrix is,

$$\mathbf{W}_{k,radar} = \text{diag}\begin{bmatrix} \Sigma_{x,radar} & \Sigma_{y,radar} & \Sigma_{v,radar} & \Sigma_{\phi,radar} & \Sigma_{\omega,radar} \end{bmatrix}, \tag{3.2}$$

$$\mathbf{W}_{k,IMU} = \text{diag}\begin{bmatrix} \Sigma_{v,IMU} & \Sigma_{\omega,IMU} \end{bmatrix}, \tag{3.3}$$

where the variables gets assigned different values, depending on the amount of noise. From the IMU both the yaw rate and the velocity are accurate, and they will get a low value. From the scan matching, the position and angle are rather accurate while from the INED, the velocity is accurate but not the angular rate. Fusing both the measurement models results in a tuning matrix as,

$$\mathbf{W}_{k,fuse} = \begin{bmatrix} \mathbf{W}_{k,IMU} & \mathbf{0} \\ \mathbf{0} & \mathbf{W}_{k,radar} \end{bmatrix}. \tag{3.4}$$

3.1.4 Free space estimation

The main purpose of the free space block is to classify areas that are free, using the trajectory and the radar scans. The entire process is visualized in Figure 3.5.

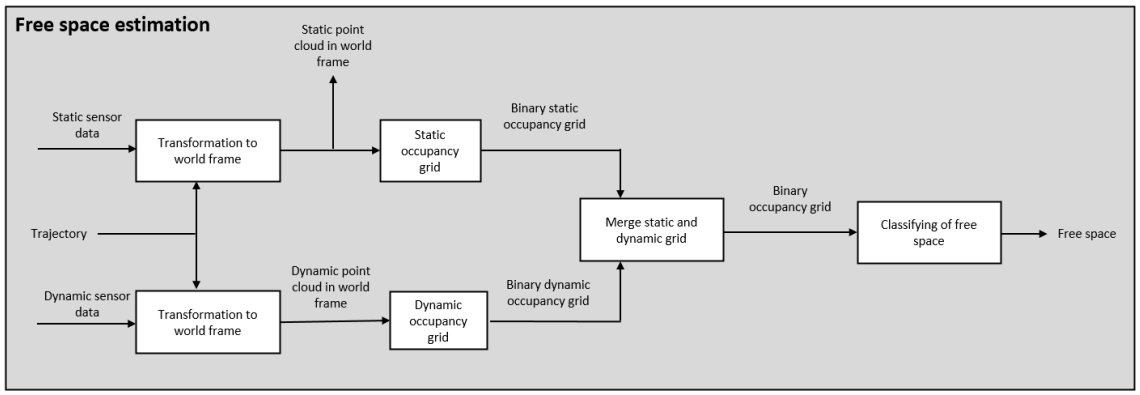


Figure 3.5: A scheme of the estimation of free space block, using radars.

3.1.4.1 Mapping with point clouds

The filtered static detections are transformed from the vehicle frame to the global frame to create a point-based map that is updated at every time step. Using the estimated trajectory from the previous step, each scan is transformed from the origin to the current location with the new $\mathbf{C}_o^g$ matrix. The point cloud is represented in 3D, since the accuracy of the estimations from the scan matching and INED increases while using 3D data. The dynamic detections in every time step are not used in updating the map since it then will be distorted over time.

3.1.4.2 Mapping with occupancy grid

The point cloud is transformed to 2D by compressing the detections between 0 and 3 meters on the z-axis. An occupancy grid is then created based on the methodology described in Section 2.5. The reason for using a grid is because it is easier to classify the free space by looking at the number of cells in a grid. For the comparison between the radar and USS, the grid needs to be in the 2D plane because of the two dimensional data given by the USS. A separate dynamic occupancy grid is created for the current time step. From the two occupancy grids, two binary grids are made, which describe the cells as either free or occupied.

The binary occupancy grid is improved by analyzing the neighboring cells. If a specific cell has neighbors that are estimated to be occupied, the cell has a higher probability to be occupied. The binary grid describing where objects are located will be clustered to make the objects denser, as described in [33]. If a cell has neighboring cells on both sides that are occupied, either vertical or horizontal, as in Figure 3.6, the middle cell is also set to be occupied. It is done once every radar scan to save computational time.

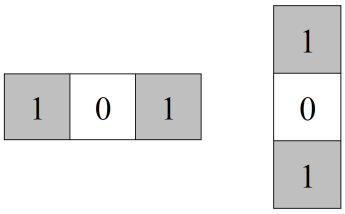


Figure 3.6: An explanation of which cells that are change to improve the binary grid. In this case, the zeros will be changed to ones.

The dynamic and static binary grids are then merged. From this merged binary grid, the free space was estimated using the following steps. First, by using the field of view for each radar at that exact moment, the area between the sensor and the object is classified as free space. By using the field of view for the radars, an area where no detections are found will be classified as free space. The binary grid will be updated over time and result in a grid describing the surroundings from the start. Merging with the dynamic grid at the current time step and using the field of view will result in an accurate representation of the environment.

3.2 Ultrasonic methods

Because of the low amount of information from the ultrasonic sensors, the methods for estimating the free space are slightly different. In general, the process can be described as the following. The measurements from the 12 sensors are transformed to the vehicle frame and then to the global frame. A CT motion model is created for the vehicle and an extended Kalman filter is implemented to estimate the trajectory of the car. An occupancy grid is created to model the environment with the filtered data from the sensors at the estimated position. A binary grid is created from the occupancy grid, and the free space is classified from this. Figure 3.7 shows a block scheme of this procedure.

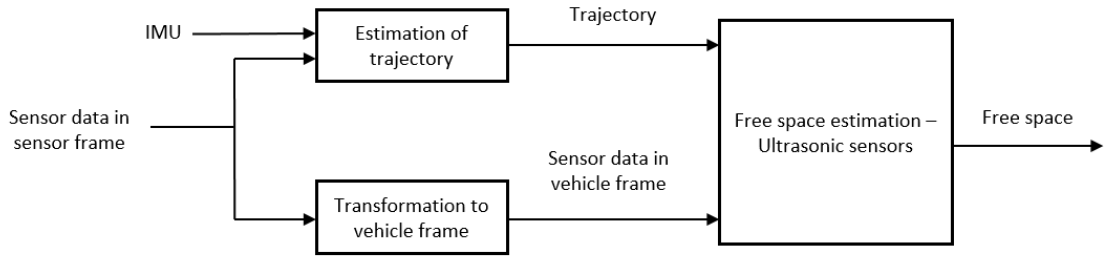


Figure 3.7: Block scheme of the methods used for the estimation using USS.

3.2.1 Estimation of trajectory

Since the data from the USS only contains ranges, it is not possible to estimate the trajectory from only this. Instead, the trajectory is used from the radar and IMU data. This was described more in detail in Section 3.1.3.3.

3.2.2 Free space estimation

Using the field of view for each sensor up to the measured range, it is possible to estimate the free space from a single scan. The free space from each sensor is first transformed into the vehicle frame. The transformation to the vehicle frame is then performed with a translation of x and y, and a rotation around the z-axis. Then the total free space is calculated as the union of the individual field of view from all 12 sensors. The free space is represented as a binary grid, where the free space has the value 1 and the rest 0. Fig. 3.8 shows the free space from the ultrasonic sensors in white. The axes represents the index of each cell in the grid. In this example, there are cars parked to the left of the vehicle.

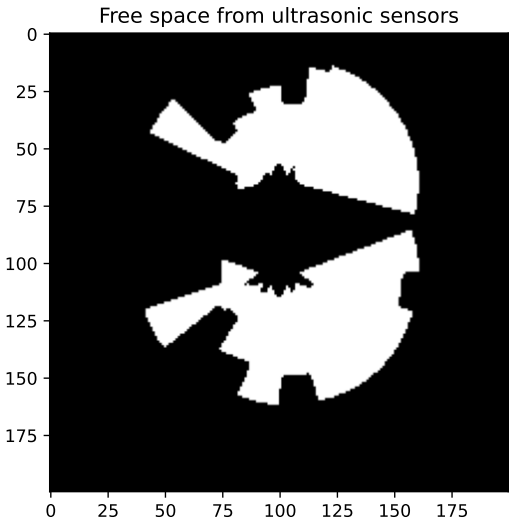


Figure 3.8: Free space given the raw USS measurements for one time step.

To make the system more reliable, the free space is estimated using multiple scans and the estimated trajectory. An occupancy grid is used to get rid of noise and to remember how the free space was at previous time steps. To update the grid over time, a decision of the thickness of the objects has to be made, since the position of the object is unknown (somewhere along the radius of the field of view). The area between the sensor up to the field of view gets a low probability of being occupied. The area outside the field of view with the decided thickness gets a high probability of being occupied. By transforming to the global frame using the trajectory and then updating the occupancy grid, a binary grid free from objects is estimated. Selecting the area that is in the field of view of the sensors is used to estimate the free space at the current time step. Fig. 3.9 shows the block scheme of how the free space is calculated using multiple scans.

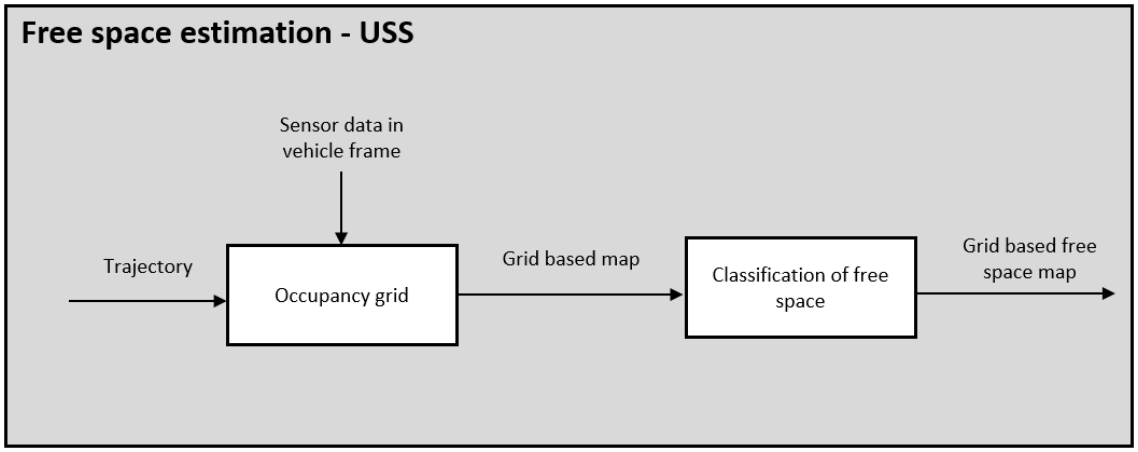


Figure 3.9: A scheme of the estimation of free space block, using USS.

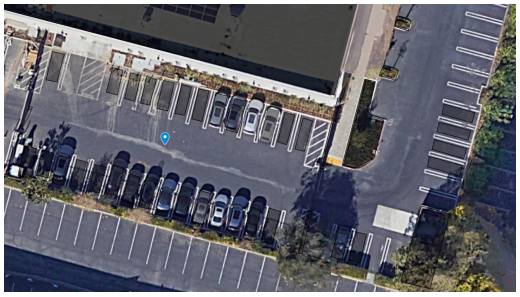
3.3 Methods of comparing results

A visual inspection of different results is easy to perform. However, to compare estimations and get a solid, measurable result of the performance when no clear ground truth exists is more difficult. We will next explain our methodology for the comparison of the different sensors for both trajectory and free space estimation.

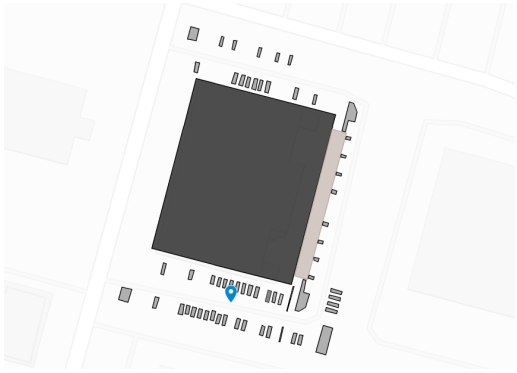
The visual evaluation of both the trajectory and the free space will be done by comparing it with a generated map from Google maps. All the data collections has been done at the same parking lot and this lot is modeled. The building and major obstacles are modeled from the satellite view on Google maps, while the positions of cars are estimated from camera images from the car and then added to the satellite view as similar shapes. A small part of the parking lot in satellite view can be seen in Fig. 3.10a. The entire map can be seen in Fig. 3.10b. The coordinates are then converted to easting and northing and finally to x and y coordinates in meters. The origin is set to be at the beginning of the sequence. The start position, end position, and heading are estimated using camera images. Fig. 3.11a shows the plotted converted coordinates. The map was also converted to a binary grid with the same grid size. This makes it possible to compare cells that are free with cells that are estimated as free. Fig. 3.11b shows the grid representation of the map, where the white is occupied and the black is free space.

3.3.1 Trajectory evaluation

The trajectory will be visualized in the generated map from Google maps. The start position, end position, and heading are estimated from camera images. The end position is then compared to the final position of the trajectory. It will not give any value of how well it performs over time, but it gives a value of how close the final position of the trajectory is to the estimated final position from camera images. The trajectory will also be evaluated visually to see how the different methods perform.

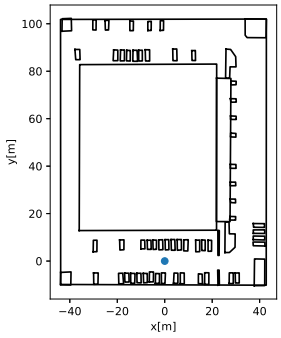


(a) Zoomed in picture of the shapes drawn in satellite view in Google Maps

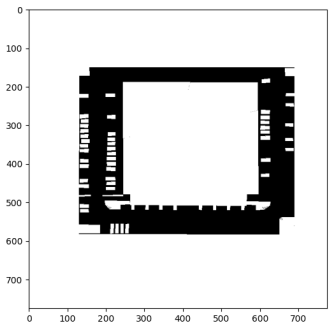


(b) The map of the entire parking lot

Figure 3.10: Images of the parking lot used for data collection



(a) Map showing the outlines of objects



(b) Map represented as a binary grid

Figure 3.11: Map of the environment from Google maps used for evaluation.

3.3.2 Free space evaluation

To compare the different free space estimations, two approaches are used. The first is to visually compare the estimation with the map from Google maps. The second is to use the binary grid to check how many cells are classified correctly and how many cells are classified incorrectly. Each cell in the grid can be classified as true free space, false free space, true occupied area, or false occupied area. True free space is when the cell is indeed free and estimated as free space. False free space is when a cell is occupied but estimated to be free space. The true occupied area is occupied and estimated to be occupied. The false occupied area is estimated to be occupied when it is free space. This will result in a percentage of correctly classified cells, which is comparable for the two different sensors with different ranges and fields of view. To make the free space estimation fair, the same trajectory is used for both radar and USS. This is because the free space estimations are dependent on the trajectories.

4
Results

In this section, we present the results of the different methods and algorithms used to estimate the trajectory and the free space. In the first sections, the sequences used for the data collection are described, as well as some analysis of the data. The final section contains the comparison of the free space estimations.

4.1 Data collection

The data used for verifying the results of our algorithms and methods were collected in the United States. Seven different sequences were collected on two separate days, all during the daytime, in sunny weather with no extreme temperatures. All of the data collections were performed in the same parking lot, as can be seen in Figure 3.10b. A short description of the different sequences can be seen in Table 4.1. Sequences 2 and 3 were collected on another day, meaning the true free space was different.

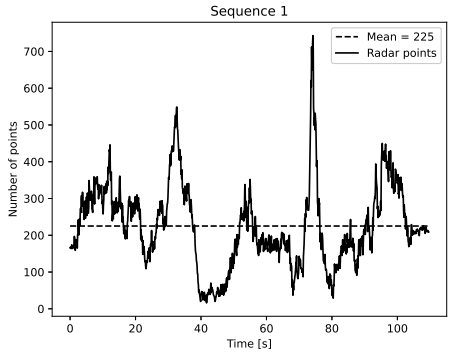
Table 4.1: Description of the different sequences used in testing.

Sequence nr	**Description**	**Duration [s]**	**Top speed [m/s]**	**Environment**
1	The car driving one lap around the building and ending up close to the starting point.	109.59	6.67	Static
2	The car driving straight, parking in a empty space, going back, and finally stopping in a spot close to the starting point.	79.09	1.90	Static
3	Similar to sequence 2, in the same area, but with different starting and stopping places.	73.89	4.18	Static
4	The car driving towards a wall and stopping at the distances 15, 10, 5, 3 and 1 meter in front of it. A person is walking to the right of the car to check the distances	67.09	0.96	Static/Dynamic
5	The car standing still with a person walking behind it	29.91	0	Dynamic
6	The car is standing still with another car driving behind it	40.58	0	Dynamic
7	The car drives towards another car that stands still	27.90	1.59	Static

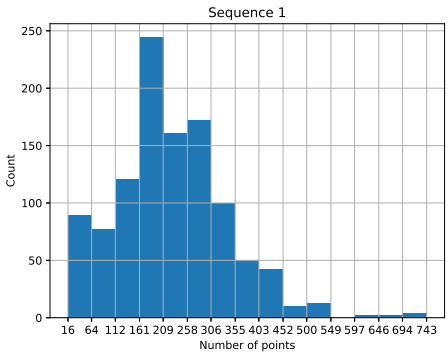
4.2 Radar data representation

From the radars, an irregularity in the number of detections was noticed. The number of detections would vary a lot during the different sequences and affect the performance. The data was analyzed and the result is shown in figures 4.1 - 4.7. Subfigures (a) visualize the change in the number of detections over time (note the different axes). This is the total number of detections for the four radars. The mean is visualized as a dotted line. Subfigures (b) show histograms of the number of detections, to show the distribution (note the different axes and bin size).

Looking at the number of detections over time, one can see that the combined number from the four radars sometimes is less than 50, which is very low. Both the clustering, scan matching, and INED requires a large number of detections to work as well as possible. Sequences 2 and 4 are the only sequences where all time steps contain more than 100 detections. Looking at sequences 2 and 3, which are collected around the same time and area, shows a quite different result. The mean for sequence 2 is 268 detections, while the mean for sequence 3 is only 125, and the distribution is different. The number of detections for the individual sensors was investigated, but no pattern could be found. Looking at the histograms, the different distributions show no clear pattern as well. A large number of detections or a rapidly changing number is not an issue, but a low number of detections is problematic.

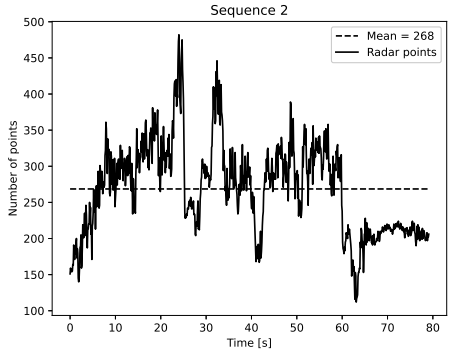


(a) Number of detections over time

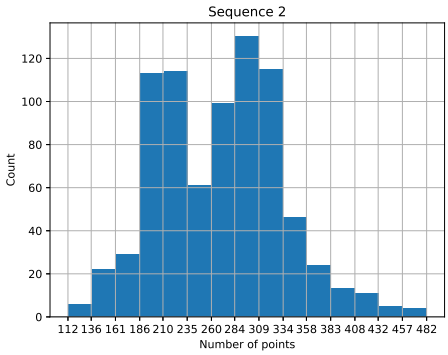


(b) Histogram of the number of detections

Figure 4.1: Number of detections analysis of sequence 1

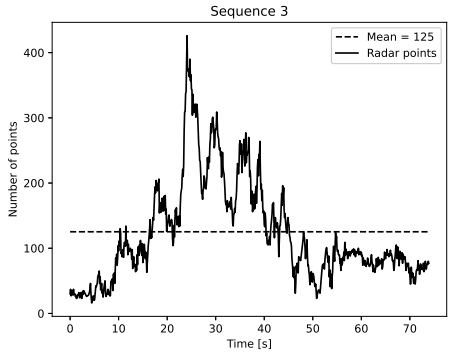


(a) Number of detections over time

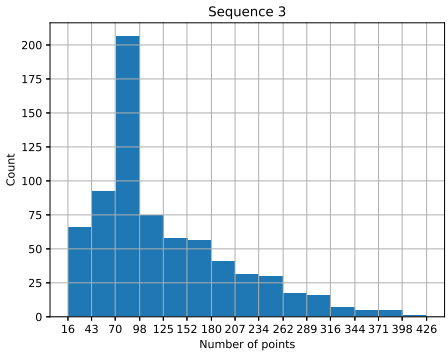


(b) Histogram of the number of detections

Figure 4.2: Number of detections analysis of sequence 2

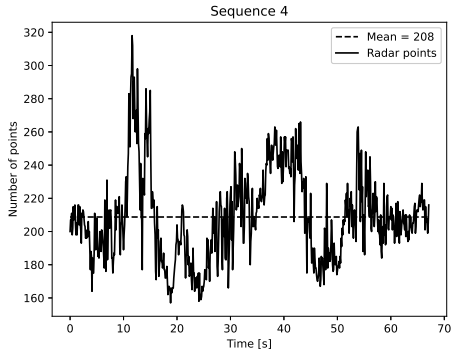


(a) Number of detections over time

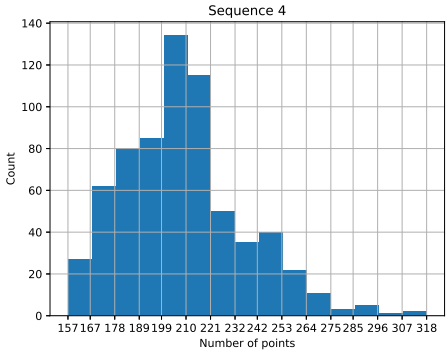


(b) Histogram of the number of detections

Figure 4.3: Number of detections analysis of sequence 3

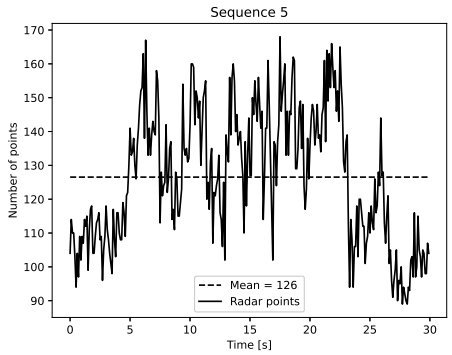


(a) Number of detections over time

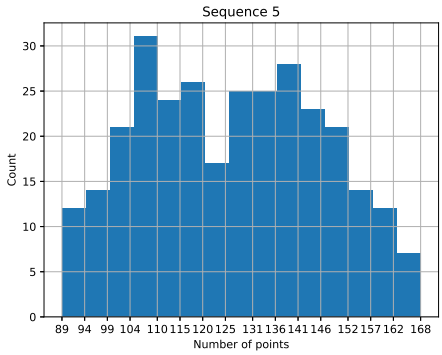


(b) Histogram of the number of detections

Figure 4.4: Number of detections analysis of sequence 4

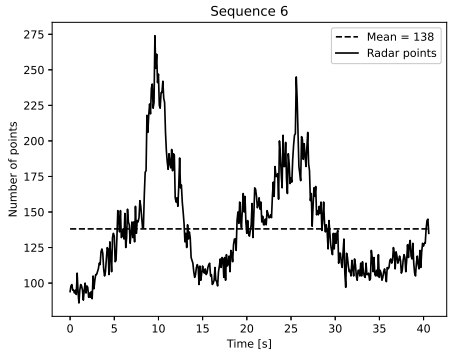


(a) Number of detections over time

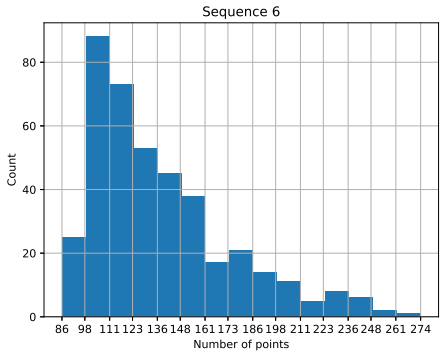


(b) Histogram of the number of detections

Figure 4.5: Number of detections analysis of sequence 5

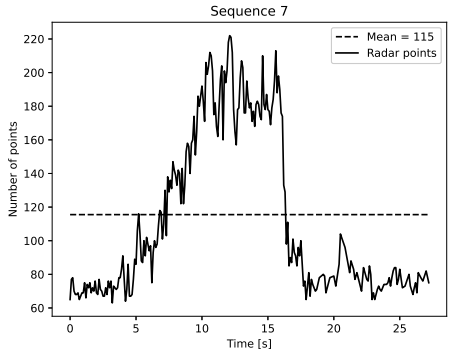


(a) Number of detections over time

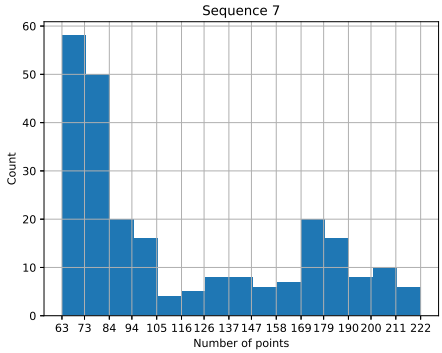


(b) Histogram of the number of detections

Figure 4.6: Number of detections analysis of sequence 6



(a) Number of detections over time

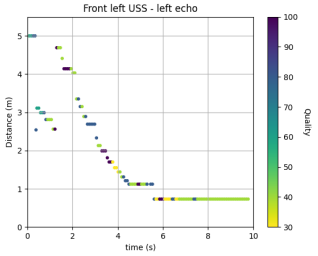


(b) Histogram of the number of detections

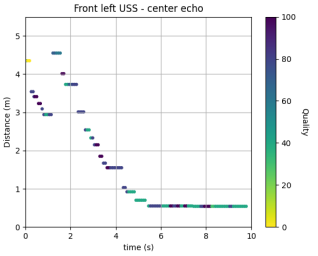
Figure 4.7: Number of detections analysis of sequence 7

4.3 Distance analysis

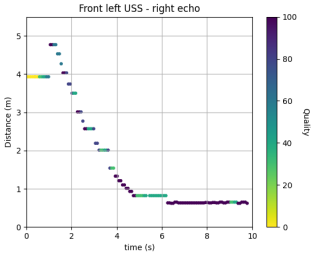
To analyze the performance of the radars and the USS, raw data from sequence 7 was used, where the ego vehicle drives towards another car. The ego vehicle started around 16 meters from the car and drove forward until the front was approximately 0.5 meter from the other car. This was used to see how well the sensors measure the distance to an object, and how much noise the measurements contain. Since the ego vehicle drove straight towards the other car, only the sensors that have a field of view directly forward were analyzed. Two ultrasonic sensors have a field of view forward, the front left and the front right USS. Figure 4.8 shows the measured distances to the car for all echoes from the USS. The color of each measurement represents the quality of the signal. In Figure 4.9, the measured distances to the car in front using radars are visualized. The field of view of the two front radars, looking forward, is used. The color-bar of the detections is the SNR of each detection. Figure 4.9a shows the measured distances for the whole sequence. Figure 4.9b shows zoomed-in data for the final 5 meters.



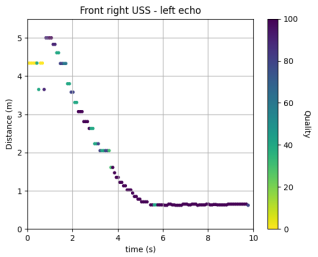
(a) Range measured with front left USS, left echo



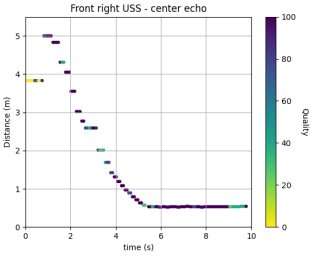
(b) Range measured with front left USS, center echo



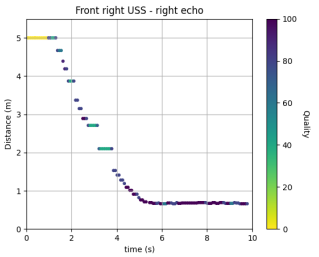
(c) Range measured with front left USS, right echo



(d) Range measured with front right USS, left echo

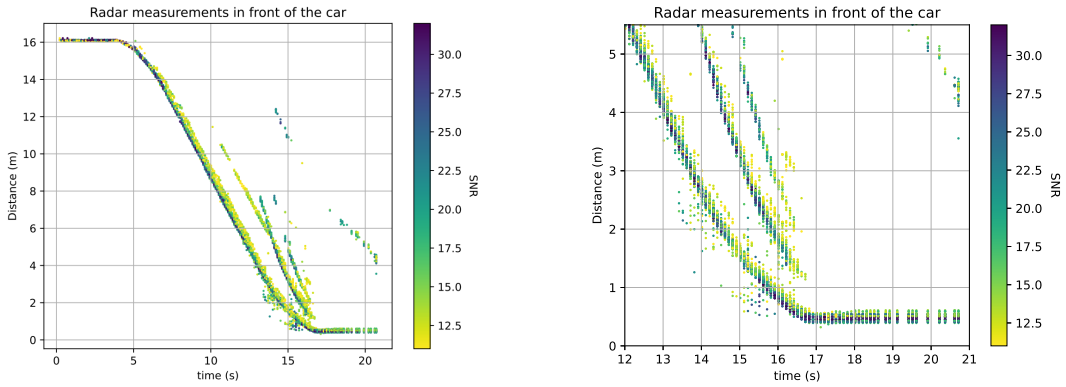


(e) Range measured with front right USS, center echo



(f) Range measured with front right USS, right echo

Figure 4.8: Measured distance to an object in front of the car using USS



(a) Distance to target car, full sequence **(b)** Distance to target car, final 5 meters

Figure 4.9: Measured distance to an object in front of the car using radar

The USS gets one measurement per time step for each sensor. They have a resolution of 1 cm. For short-range, it gives stable measurements with high accuracy. For longer range, it gives lower quality measurements with more varying distances. For ranges over 5 meters, they often provide false readings with much lower quality. The radar measures multiple detections each time step. The SNR are between 12.5 and 32 dB. The reason why it gives multiple ranges at the same time step is that some detections go through the car and measure further away than the side towards the ego vehicle. When the ego vehicle is standing still close to an object, the resulting range is stable.

4.4 Dynamic objects filtering

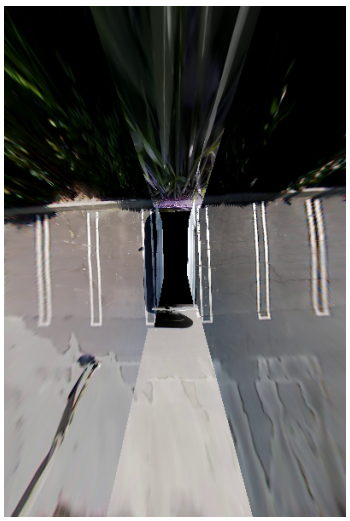
The filtering of dynamic objects was tested on sequences 4, 5, and 6. The method used for classifying the dynamic detections was presented in section 3.1.2.1. The sequences are measured for six seconds. The average number of inliers (static) and outliers (dynamic) are presented in Table 4.2. One thing to note is that the number of static and dynamic detections for sequences 5 and 6 are not affected much by the threshold ϵ.

Table 4.2: Average number of static and dynamic detections using different ϵ

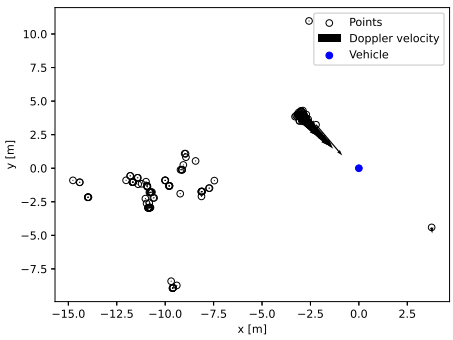
	Sequence 5		**Sequence 6**		**Sequence 4**	
ϵ	Static	Dynamic	Static	Dynamic	Static	Dynamic
0.1	90.80	24.66	151.23	68.63	153.07	47.00
0.2	90.80	24.66	151.23	68.63	189.13	11.93
0.3	92.30	23.16	161.93	57.93	194.23	6.83

The result for sequence 5 are presented in Figure 4.10, sequence 6 in Figure 4.11 and sequence 4 in Figure 4.12, all for a single time step at three seconds. The surroundings, captured with four cameras are fused into one 360° image and can be seen in subfigure (a). In subfigure (b), all radar detections are plotted together

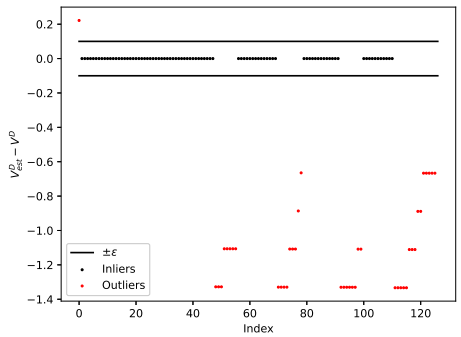
with the direction and magnitude of the Doppler velocities. A RANSAC algorithm is performed with 1000 iterations and the threshold ϵ varying. With all detections in the form of the system motion equation given at (2.46), the best estimate of the state $\mathbf{b} = [\omega, v_x, v_y]^T$ is solved for using (2.47). The estimated state is multiplied with the $\mathbf{R}$ matrix in (2.46) to get the expected Doppler velocities v^D_{est}. These are then compared to the measured Doppler velocities v^D and plotted in subfigures (c) for sequences 5 and 6. Subfigures (d) contains the inliers (black) and outliers (red), which could be compared with the Doppler magnitudes in subfigures (b). In sequence 4 three different thresholds are visualized. The points inside the threshold limits are classified as static detections and these are then visualized in black, while the dynamic detections are visualized in red.



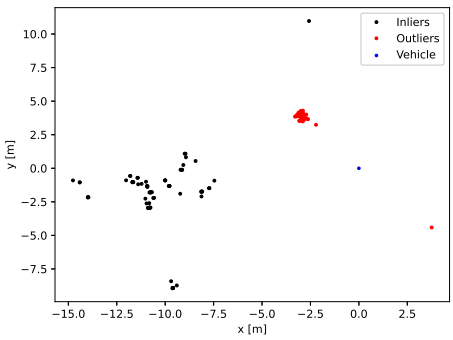
(a) Fused camera image of surroundings



(b) Radar scan with Doppler magnitudes

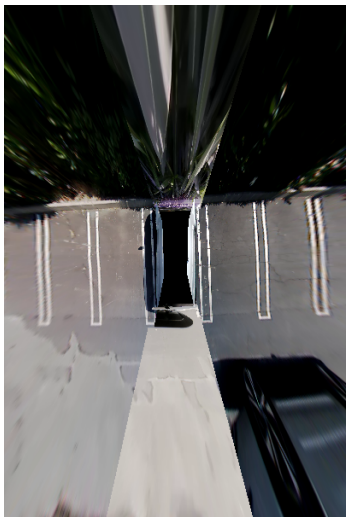


(c) Difference between the expected Doppler and measured Doppler for all detections, given the optimal RANSAC fit ($\epsilon = 0.1$)

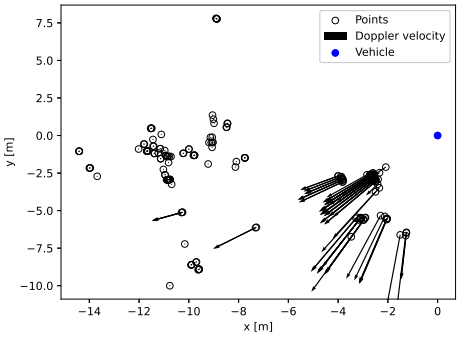


(d) Radar scan with inliers (black) and outliers (red), after RANSAC algorithm

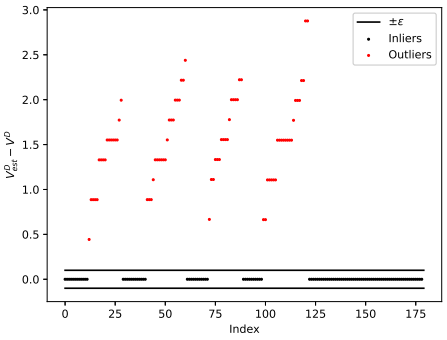
Figure 4.10: Dynamic object classification in a single time step for sequence 5 (a person walking behind the ego vehicle)



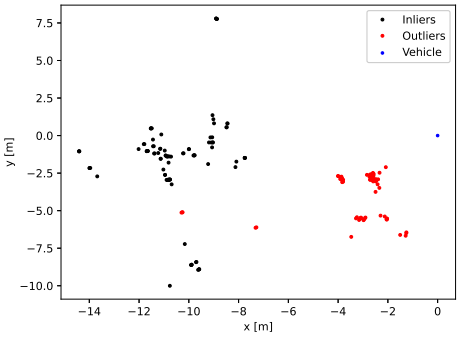
(a) Fused camera image of surroundings



(b) Radar scan with Doppler magnitudes

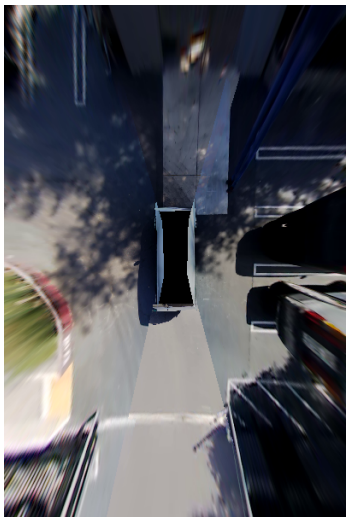


(c) Difference between the expected Doppler and measured Doppler for all detections, given the optimal RANSAC fit ($\epsilon = 0.1$)

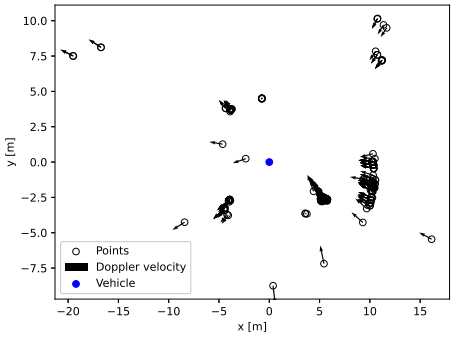


(d) Radar scan with inliers (black) and outliers (red), after RANSAC algorithm

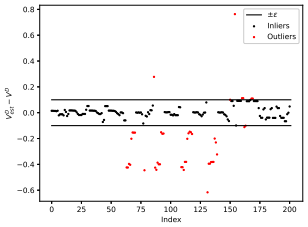
Figure 4.11: Dynamic object classification in a single time step for sequence 6 (a car driving behind the ego vehicle)



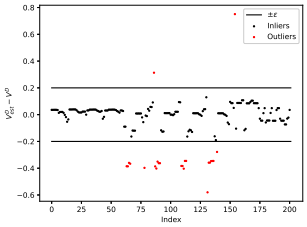
(a) Fused camera image of surroundings



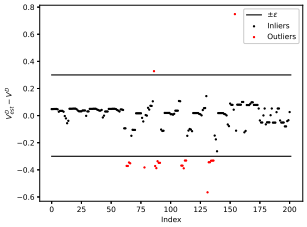
(b) Radar scan with Doppler magnitudes



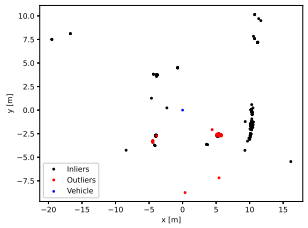
(c) Difference between the expected Doppler and measured Doppler for all detections, given the optimal RANSAC fit ($\epsilon = 0.1$)



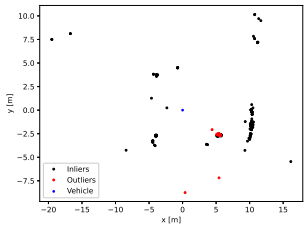
(d) Difference between the expected Doppler and measured Doppler for all detections, given the optimal RANSAC fit ($\epsilon = 0.2$)



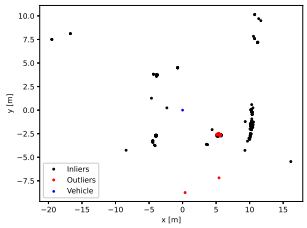
(e) Difference between the expected Doppler and measured Doppler for all detections, given the optimal RANSAC fit ($\epsilon = 0.3$)



(f) Radar scan with inliers (black) and outliers (red), after RANSAC algorithm ($\epsilon = 0.1$)



(g) Radar scan with inliers (black) and outliers (red), after RANSAC algorithm ($\epsilon = 0.2$)



(h) Radar scan with inliers (black) and outliers (red), after RANSAC algorithm ($\epsilon = 0.3$)

Figure 4.12: Dynamic object classification in a single time step for sequence 4, with different thresholds

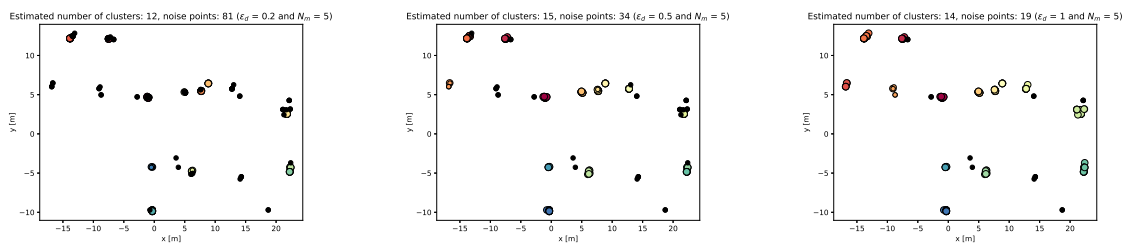
The subplots with the best estimate show the difference between the measured

Doppler velocity and the estimated Doppler velocity for each detection. The horizontal lines are the threshold and the detection with a difference inside the threshold is classified as inliers. In sequences 5 and 6, where the ego vehicle is standing still, all the detections with a Doppler velocity higher or lower than 0 are correctly classified as dynamic. Figure 4.10c and 4.11c show that the error of the static detections is exactly 0. This means that the static detections are estimated perfectly and the differences between the estimated and measured Doppler velocity are equal to 0. This can be concluded by looking at the detections with a Doppler magnitude, which is classified as dynamic and plotted in red in the final subfigure.

Figure 4.12 shows data from a sequence when the car is moving and this results in more varying errors. With a high threshold, more detections are classified as inliers, and with a low threshold, more detections are classified as outliers. In this case, a person is moving with a similar velocity as the ego vehicle, and no clear deviating Doppler magnitudes can be seen. However, using the difference between the estimated Doppler and the measured Doppler, the majority of the detections of the person are still classified as dynamic. Here, ϵ is of higher importance and affects the performance (unlike when the vehicle is standing still). In conclusion, any detections with a clearly deviating Doppler magnitude are correctly classified as dynamic, and an ϵ value on the higher side is used later in the thesis.

4.5 Clustering

The clustering algorithm is evaluated for a couple of different values of the parameters. Small ϵ_{dist} and a large N_{min} results in a lower number of clusters, while large ϵ_{dist} and small N_{min} results in a high number of clusters. The number of clusters and the number of detections classified as noise can be analyzed but do not say much about the performance. To visualize the concept, Figure 4.13 shows the number of clusters and noise for $N_{min} = 5$ and $\epsilon_{dist} = 0.2$, $\epsilon_{dist} = 0.5$, and $\epsilon_{dist} = 1$, for a single radar scan. The colored points represent the different clusters, while the black points represent the noise. The figure shows the relation between the threshold ϵ_{dist}, and the number of clusters and noise points. With a low ϵ_{dist}, the number of noise points is high, as can be seen in subfigure 4.13a, and with a high ϵ_{dist}, the number of noise points is low, as can be seen in subfigure 4.13c. The number of clusters is larger and the clusters are bigger with a high ϵ_{dist}.

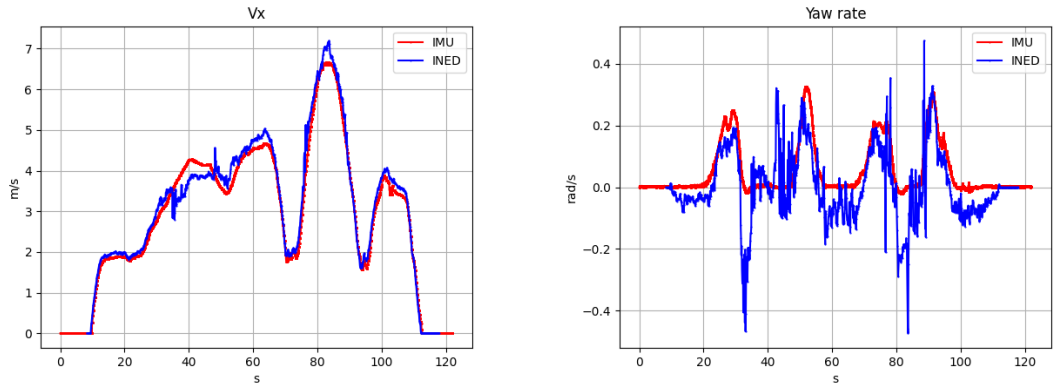


(a) $\epsilon_{dist} = 0.2$ **(b)** $\epsilon_{dist} = 0.5$ **(c)** $\epsilon_{dist} = 1$

Figure 4.13: Cluster and noise points for a radar scan, using different ϵ_{dist}

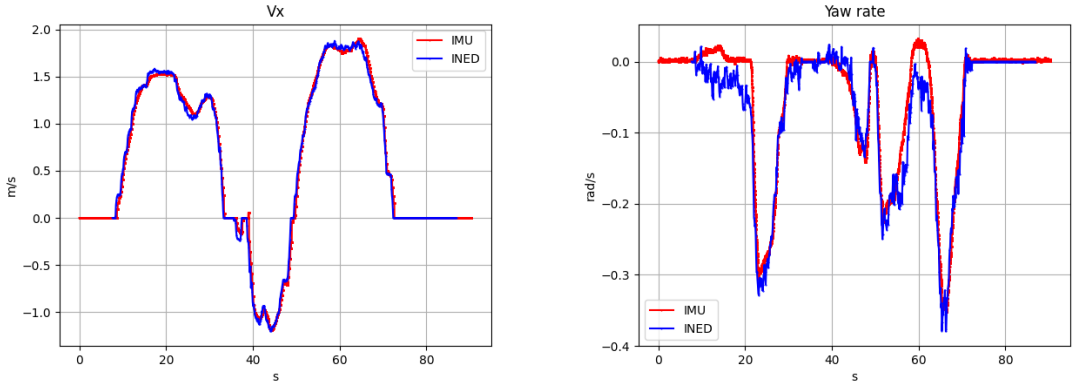
4.6 Instantaneous ego-motion using Doppler

INED was used to estimate the velocity v_x and yaw rate ω. The variable v_y was estimated but is not used, since it is not included in the state vector. To evaluate how well the velocity and yaw rate are estimated, it is compared with the IMU data. In Figure 4.14, 4.15 and 4.16, the v_x and ω are plotted for both IMU and INED, for the three first sequences. The thresholds used in the INED were $\epsilon = 0.5$ m/s and 500 RANSAC iterations. Looking at the first sequence, v_x is estimated well with small deviations at around 40 seconds, when the number of detections is low (see figure 4.1). The estimation of the yaw rate is worse than the velocity, with some negative spikes where the IMU estimates the yaw rate to be zero. Comparing sequences 2 and 3, it is clear that sequence 2 performs better, both for v_x and ω. Figure 4.2 and Figure 4.3 (number of detections over time), shows that sequence 2 has a considerably higher average number of detections. All estimations in sequence 2 are good, with a performance comparable to the IMU.



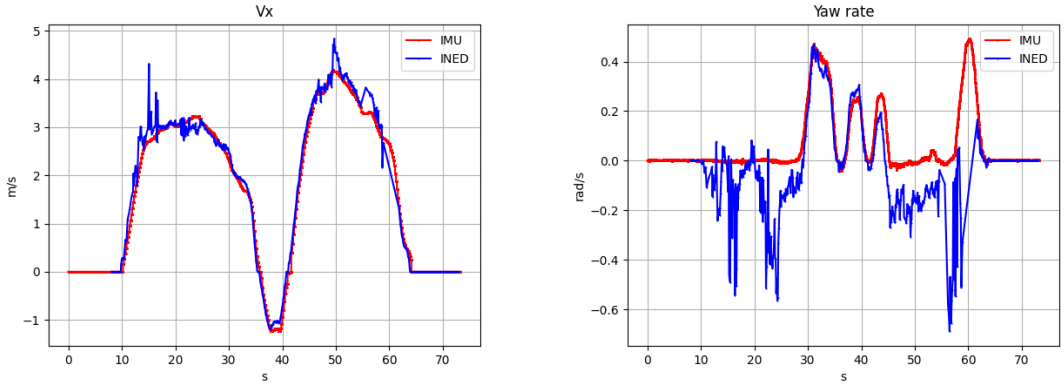
(a) Estimated v_x **(b)** Estimated yaw rate

Figure 4.14: Analysis of INED estimations, compared to the IMU for sequence 1



(a) Estimated v_x **(b)** Estimated yaw rate

Figure 4.15: Analysis of INED estimations, compared to the IMU for sequence 2

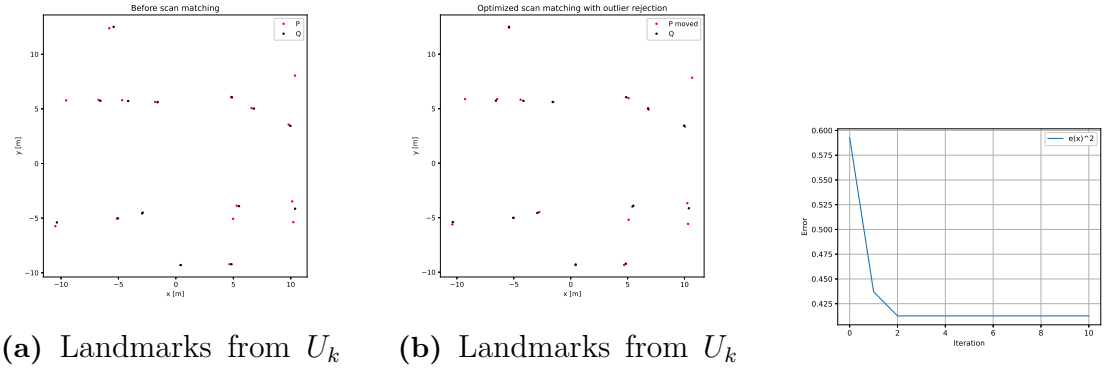


(a) Estimated v_x **(b)** Estimated yaw rate

Figure 4.16: Analysis of INED estimations, compared to the IMU for sequence 3

4.7 Scan matching

The concept of matching two different points cloud captured at two consecutive time steps U_k and U_{k-1}, is visualized first. The point clouds are clustered beforehand with $N_{min} = 5$ and $\epsilon_{dist} = 0.8$ and the landmarks are used. In Figure 4.17a there are 18 landmarks from cloud U_k and 14 landmarks from the previous point cloud U_{k-1}. The point clouds are captured in sequence 1 while driving in approximately 2 m/s, meaning that with a sampling frequency of 10 times per second, the car has traveled approximately 0.2 meters. Figure 4.17b shows the scans after transforming U_k to match U_{k-1} using Gauss-Newton optimization with outlier rejection ($\epsilon_O = 1$). Figure 4.17c shows the error from (2.38) over 10 iterations. The scan matching is done in 3D but transformed to 2D for easy visualization. With the axes ranging from ± 10 meters, in this case, the small transformation is difficult to see. However, the error is decreased in the first two iterations, before the solution converges, meaning the optimization is working.



(a) Landmarks from U_k (= P) and U_{k-1} (= Q) after before matching

(b) Landmarks from U_k (= P) and U_{k-1} (= Q) after scan matching

(c) Error over the number of iterations

Figure 4.17: The concept of the scan matching with landmarks

The scan matching in Figure 4.17 results in a translation and rotation describing the transformation, and in this case, a $\mathbf{C}$ matrix as,

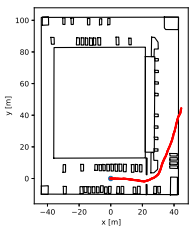
$$\mathbf{C} = \begin{bmatrix} 0.9998 & 0.0156 & 0 & 0.1760 \\ -0.0156 & 0.9998 & 0 & -0.0508 \\ 0 & 0 & 1 & -0.0173 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{4.1}$$

Multiplying this with the last position leads to a new position of

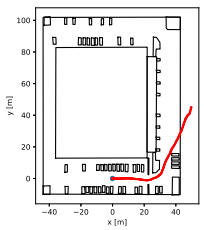
$$\mathbf{p}_{new} = \mathbf{C} \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 0.1760 \\ -0.0508 \\ -0.0173 \\ 1 \end{bmatrix}, \tag{4.2}$$

which means a movement of 0.176 meters in the x-direction, -0.0508 meters in the y-direction, and, -0.0173 meters in the z-direction.

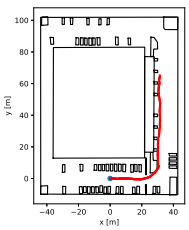
Sequence 1, 2, and 3 are evaluated over multiple time steps, using all detections, the core points, and the landmarks from the clustering, to show the difference. One of the most important tuning parameters is the outlier rejection threshold ϵ_O. A low outlier rejection threshold ($\epsilon_O < 1$ meter) will increase performance, but when the number of detections is low, the scan matching does not find any matching landmarks at all. In sequence 1, at around 40 seconds, the scan matching using landmarks does not find any correspondences (see Figure 4.1a for the number of points over time). In sequence 3 the scan matching does not find any matching detections (or landmarks), at all. Looking at sequence 1 for the first 40 seconds results in the estimations seen in Figure 4.18. This is for $N_{min} = 5$, $\epsilon_{dist} = 0.8$ and $\epsilon_O = 0.75$. Sequence 2 is shown in Figure 4.19, with the same thresholds. Looking at Figure 4.18, it is clear that the best result is made when using the landmarks. Using all detections or the core points, results in a similar performance, with an inability to estimate the turn correctly. The same pattern for all detections, core points, and landmarks is also found in Figure 4.18.



(a) SM using all detections

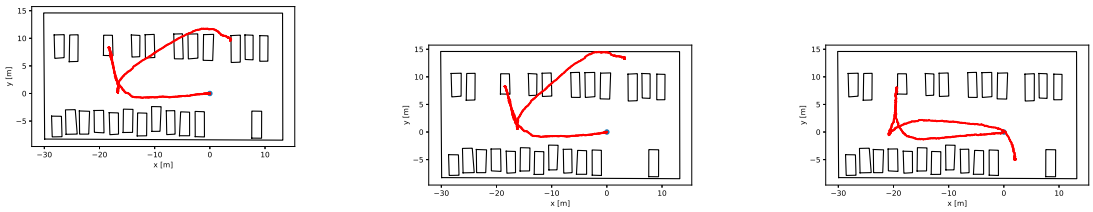


(b) SM using the core points



(c) SM using the landmarks

Figure 4.18: Estimated trajectory over 40 seconds sequence 1 using scan matching



(a) SM using all detections

(b) SM using core points

(c) SM using landmarks

Figure 4.19: Estimated trajectory over entire sequence 2 using scan matching

4.8 Estimation of trajectory

The trajectory has been estimated using different types of measurements. All estimations were done using the extended Kalman filter and a coordinated turn motion model. Since no ground truth of the trajectory exists, we evaluate the accuracy visually as well as compare the final position of the estimated trajectory to an endpoint, estimated from camera images. Since the scan matching only works for the first 40 seconds of sequence 1, this is what is analyzed from this sequence. The tuning of the filter is also performed on this sequence. The estimated trajectories in sequence 1 are plotted in Figure 4.20. The innovation for the variables and the different EKF are visualized in Figure 4.21. The innovation can be seen as the error that the EKF is trying to minimize, and is affected by our tuning. The trajectory from sequence 2 can be seen in Figure 4.22 and the innovation plots in 4.23. Sequence 3 could not be analyzed because of the low number of detections in the beginning. Using the two fused measurement models described in section 3.1.3.3 results in two innovations for every variable in our state vector. One using the state vector from the scan matching and INED (subfigure (c)) and one from the state vector from the EKF_{IMU} (subfigure (d)). The error in distance between the final position and the endpoint can be seen in Table 4.3.

Starting with a look at the trajectory estimated with IMU data in sequence 1, it has a clear drift. Using radar data instead makes the estimation fairly good but the turn is a little slow. Combining the two information sources results in a mixture of the two, and a good estimation. Looking at the innovation plots, most variables are centered around zero. The variables in the innovation plots have different units, for example the IMU in Fig 4.21b, has the velocity in [m/s] and ω in [rad/s]. This makes the plots a bit hard to read, since variables with different units are included. The fusion of radar and IMU makes the innovation for the velocity "work against each other" as seen in subfigures 4.21c and 4.21d. Looking at the estimated trajectories for sequence 2 in Figure 4.22, it is not as obvious which estimation that gives the best result. Both the separate IMU and radar EKF estimates the trajectory quite well, but the fused version performs better and is more robust. Table 4.3 confirms that the combined EKF is most accurate.

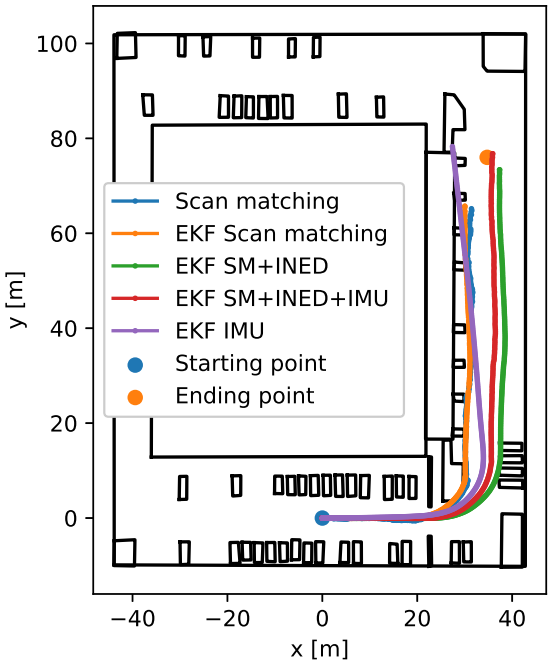
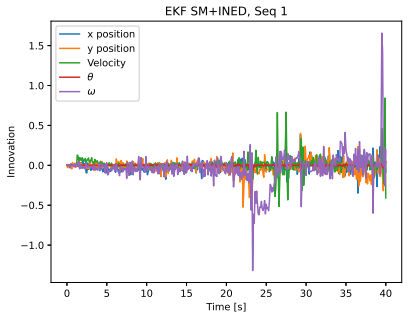
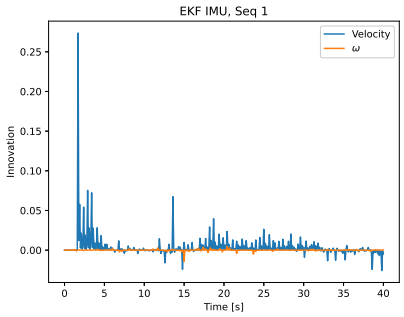


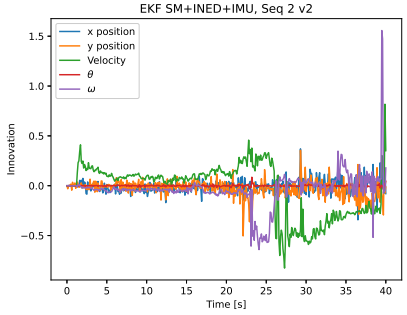
Figure 4.20: All estimated trajectories of sequence 1



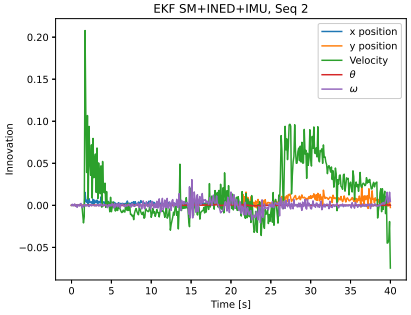
(a) SM + INED



(b) IMU



(c) SM+INED+IMU 1



(d) SM+INED+IMU 2

Figure 4.21: Innovation plots sequence 1 for SM+INED, IMU and the combination

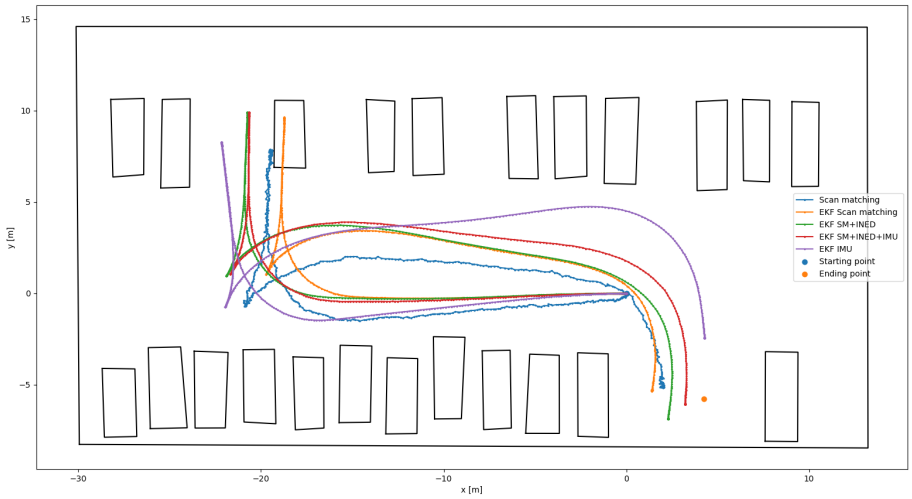
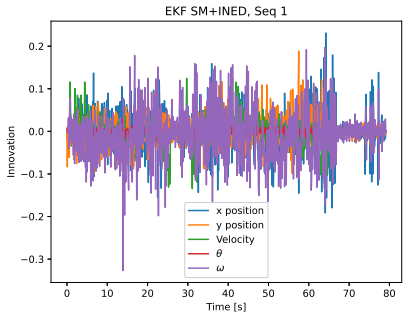
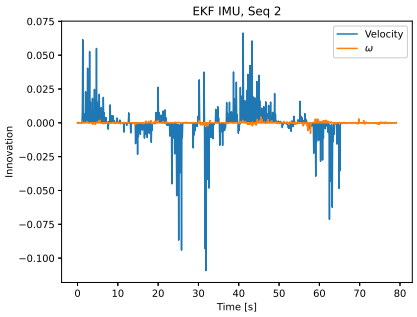


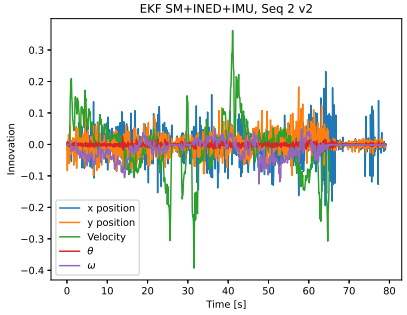
Figure 4.22: All estimated trajectories of sequence 2



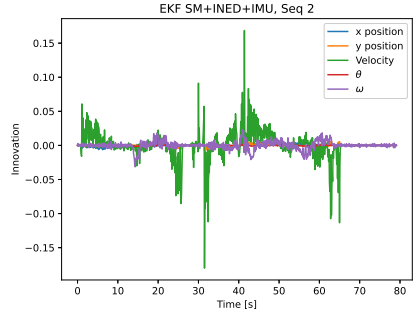
(a) SM + INED



(b) IMU



(c) SM+INED+IMU 1



(d) SM+INED+IMU 2

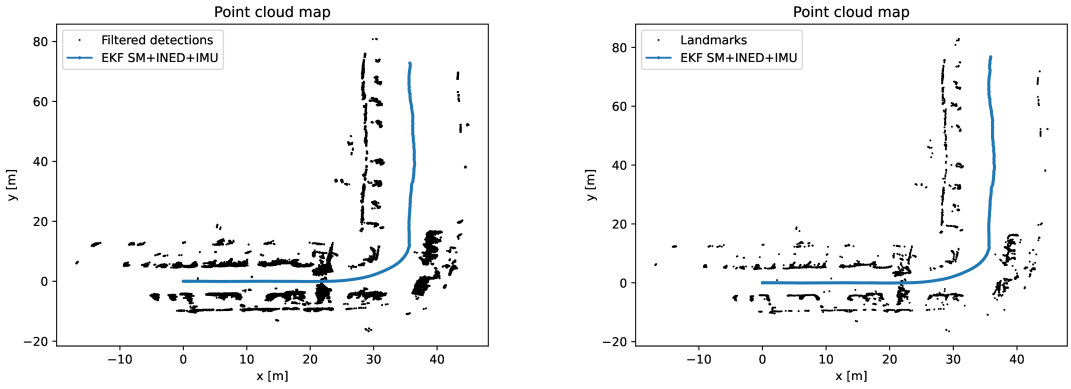
Figure 4.23: Innovation plots sequence 2 for SM+INED, IMU and the combination

Table 4.3: Table over the distance error in the final step of the estimated trajectories

Method	Error Sequence 1 [m]	Error Sequence 2 [m]
Scan matching	11.31	2.34
EKF scan matching	11.32	2.88
EKF SM + INED	3.70	2.25
EKF IMU	7.66	3.33
EKF SM + INED + IMU	1.37	1.08

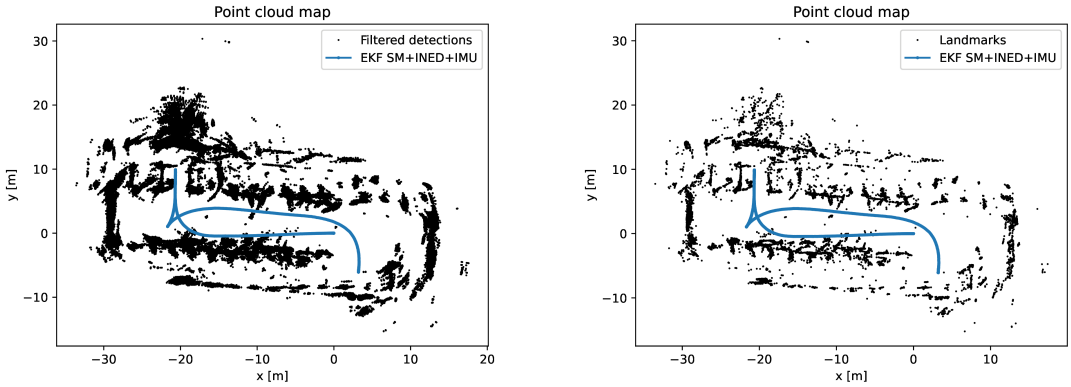
4.9 Point cloud map

With the optimal trajectory, using all the available information, the filtered detections are transformed to the estimated position and plotted in Figure 4.24a and 4.25a. Using all landmarks instead of the filtered detections results in the plot in Figure 4.24b and 4.25b. As expected, using landmarks results in a more sparse representation. Both versions of the first sequence are a good visual representation of the actual surroundings. Individual cars and objects can be observed, but the wall of the big building is not 100 % straight. The second sequence is not as good, as some of the surroundings are observed twice, both in the beginning and at the end of the sequence. Here the individual cars are not distinguishable, but the outlines of the parking lot are similar to the true surroundings.



(a) Using filtered detections **(b)** Using landmarks

Figure 4.24: Trajectory estimation and point cloud representation of sequence 1



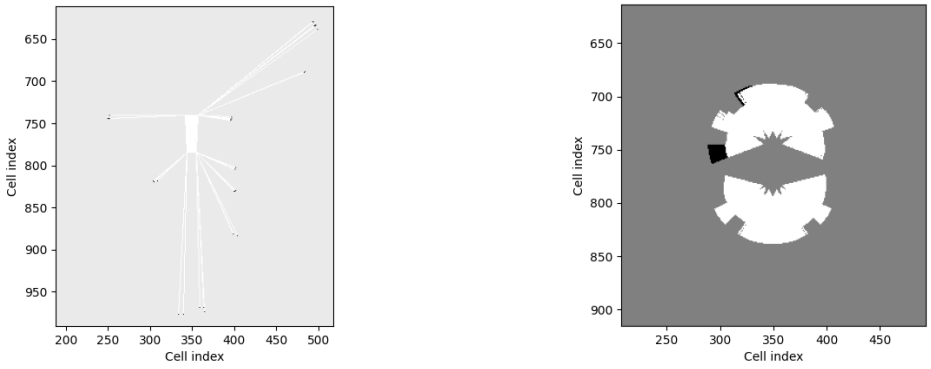
(a) Using filtered detections **(b)** Using landmarks

Figure 4.25: Trajectory estimation and point cloud representation of sequence 2

4.10 Occupancy grid

An occupancy grid is created for both the static detections and the dynamic detections. These binary grids, which represent the objects, are merged to create a map representing the whole environment. All estimations in this section are done using sequence 1. Different cell sizes were investigated and a final cell size of 0.01 m^2 is used. This requires some computational power but results in a detailed view.

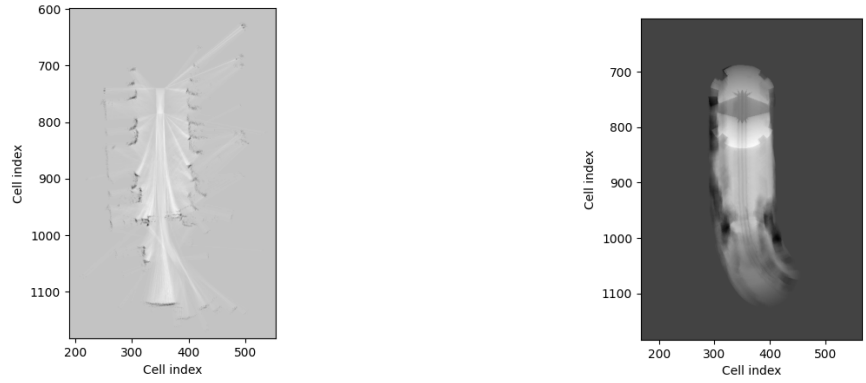
The occupancy grid was created to map the environment. This section describes what the grid looks in the different steps. This is done for both the radars and the ultrasonic sensors. From each time step, a log-odds probability grid is created, which describes how the environment looks, using only one scan. Figure 4.26 shows an example of a log-odds probability grid from one scan for both the radars and the ultrasonic sensors. The black area has a high log-odds value and is more likely to be occupied. The white cells have lower log-odds values and are more likely to be free space. The gray area has a log-odds value of 0, which means that it has the same probability of being free as being occupied. Since there are few detections per scan, this representation is very sparse.



(a) Radar **(b)** Ultrasonic sensors

Figure 4.26: An example of the log-odds probability grid from a single scan using radar and USS. The white area is assumed to be free space, the black is assumed to be occupied and the gray has not been seen.

By using the trajectory, the grid is updated with more scans and which means a better representation of the environment. The trajectory that is used is the estimation where IMU and radar data were fused from Section 4.8. The log-odds probability grid from each time step was added on top of each other to get the total log-odds probability grid. In Figure 4.27, the total grid is visualized, which is more detailed and with more shades of gray.

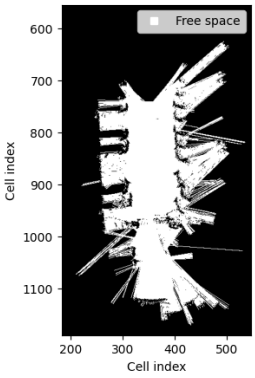


(a) Radar **(b)** Ultrasonic sensors

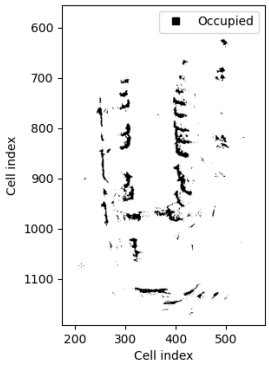
Figure 4.27: Log-odds probability grid from multiple scans in sequence 1 for both radar and USS. The darker it is, the higher probability it is that it is occupied.

From the total log-odds probability grid in Figure 4.27, the binary grids are created. By selecting the cells with log-odds values under 0, the grid representing the free space is made. This binary grid is shown in Figure 4.28a for the radars. Since the radar signal can penetrate objects, this free space is estimated to be larger than it really is. For example, when a signal goes through a car and reflects on the other side. Then this area is classified as free space when it should be occupied. A binary

grid containing the occupied cells is created by selecting the values over 0, which is shown in Figure 4.28b.



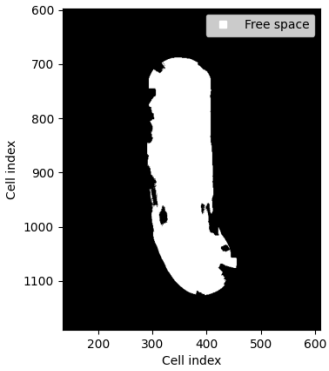
(a) Binary occupancy grid representing free space



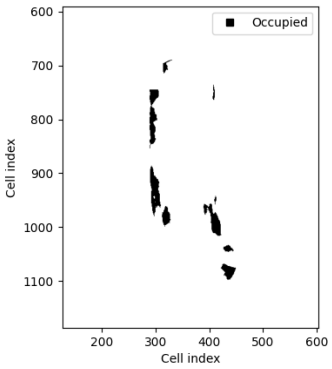
(b) Binary occupancy grid representing objects

Figure 4.28: Binary occupancy grid using radar in sequence 1

The binary grid for the ultrasonic sensors is presented in Figure 4.29. Since the range of the ultrasonic sensors is shorter than the radars, the number of detected objects and free space is much smaller.



(a) Binary occupancy grid representing free space



(b) Binary occupancy grid representing objects

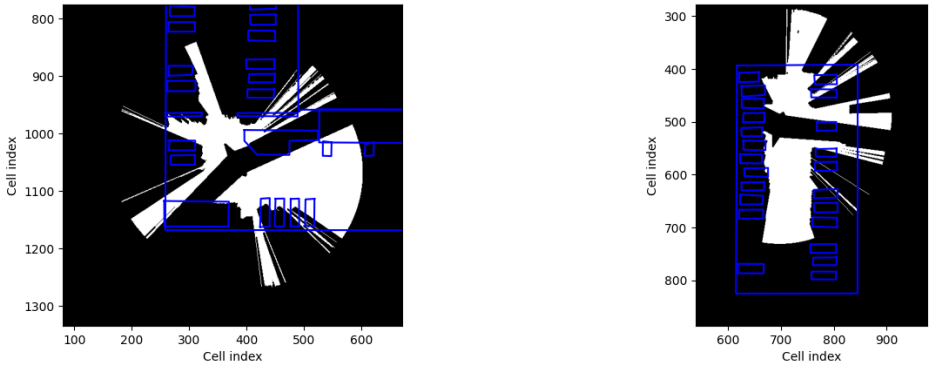
Figure 4.29: Occupancy grid using ultrasonic sensors in sequence 1

Subfigures 4.28a and 4.29a are both representing the free space of the whole sequence. Subfigures 4.28b and 4.29b show where objects are located. When using radars, more information about the environment is found. Since the radars have a longer range, it collects information from objects further away. It also gives more details about the environment, since the exact position of the detections is known. The benefit of USS is that it estimates the objects to be denser. The binary grid representing the objects in Figure 4.29b, is more compact and does not contain as much details as for the radars in Figure 4.28b.

4.11 Free space classification

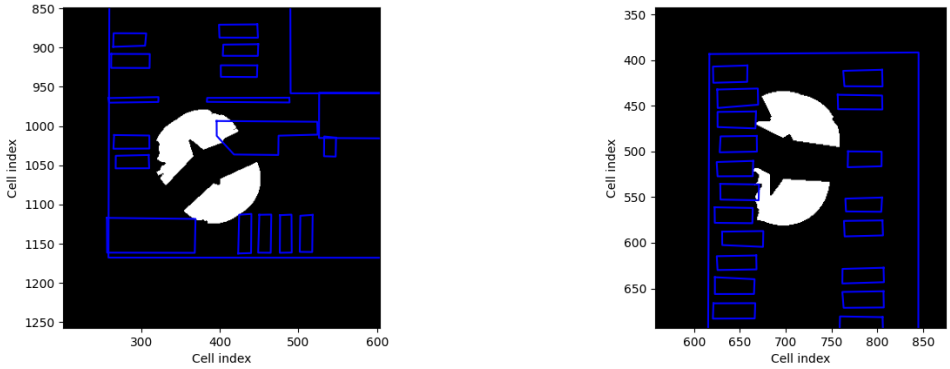
To estimate the final free space, the binary occupancy grid that represents objects is used. To make the analysis fairer, the radars are evaluated within both the full range of 20 meters, as well as 5 meters, which is the same range as the USS.

In figures 4.30, 4.31 and 4.32, two time-steps of free space estimations from sequence 1 and 2 are visualized in white, together with the map from Google maps in blue. The radar with 20 meters range classifies a much larger free space than the USS and the radar with 5 meters range. The estimation from the radars gives a more detailed representation of the environment. However, the free space from the radar with 20 meters range is less accurate than the USS and radar with 5 meters range.



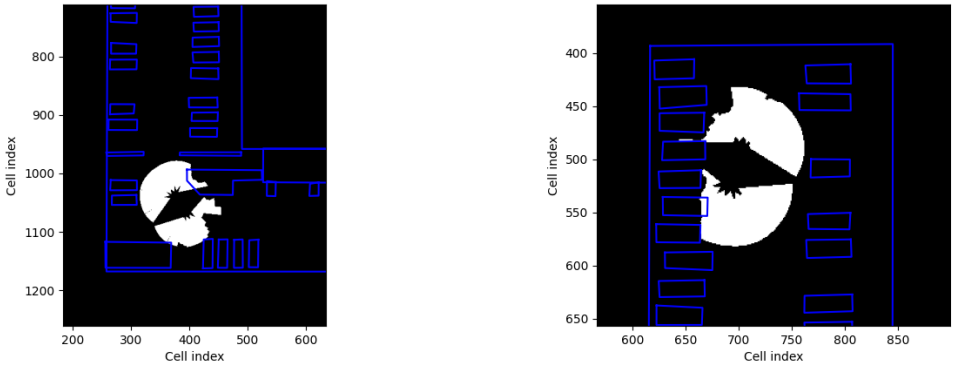
(a) Sequence 1 **(b)** Sequence 2

Figure 4.30: A visualization of the free space estimation using radars with 20 meters range.



(a) Sequence 1 **(b)** Sequence 2

Figure 4.31: A visualization of the free space estimation using radars with 5 meters range.



(a) Sequence 1 **(b)** Sequence 2

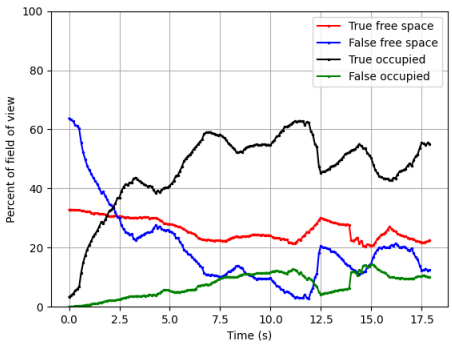
Figure 4.32: A visualization of the free space estimation using USS

To analyze the performance of the free space from the different sensors, the numerical method from Section 3.3.2 is implemented. The values that are analyzed, are the true free space, false free space, true occupied area, false occupied area, total correctly classified area, and the total free space area. One indicator of the performance is the total classified free space area. In Table 4.4, the average classified free space area of sequences 1 and 2 is presented. When the radar's full range is used, it gets a considerably larger free space area than the other two, which is expected. The USS and the radar with a 5-meter range get a similar result, with the USS managing to classify a bit larger area. This could be because of the number of sensors and their positions.

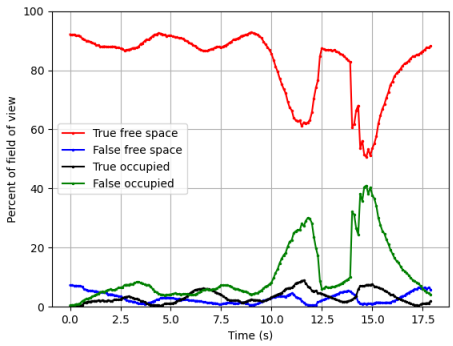
Table 4.4: Average classified free space area for sequence 1 and 2

Sequence	Area classified with radars, 20 meters range [m^2]	Area classified with radars, 5 meters range [m^2]	Area classified with USS, 5 meters range [m^2]
Sequence 1	**607.72**	**86.06**	**99.10**
Sequence 2	**677.12**	**76.17**	**84.44**

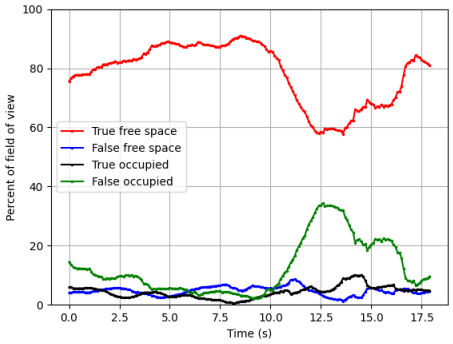
In Figure 4.33, the percentages of each estimation for sequence 1 are plotted over time. All percentages that are presented are dependent on the size of the field of view. For the first 2.5 seconds, the free space for radar with full range classifies more wrong than it classifies correctly. The false free space is high in the beginning but decreases in the first 7 seconds. The reason why this happens is because of the few detections at the beginning of the sequence. In Table 4.5 and 4.6, the average percentages for radars and USS for sequence 1 and 2, are presented. The total correctly classified area, which is also presented in Table 4.5 and 4.6, is the sum of true free space and true occupied area. The radar with a 5-meter range has the highest percent of correctly classified cells, followed by the USS and then the radar with full range.



(a) Analysis of estimated free space for radar within 20 meters



(b) Analysis of estimated free space for radar within 5 meters



(c) Analysis of estimated free space for USS for sequence 1

Figure 4.33: Analysis of the correctly classified cells of the estimated free space for USS for sequence 1

Table 4.5: Average percent correctly and incorrectly classified cells, for each sensor and range, for sequence 1

	Radar with 20 meters range	**Radar with 5 meters range**	**USS with 5 meters range**
True free space	**26.11 %**	**82.77 %**	**78.41 %**
False free space	**19.79 %**	**2.93 %**	**4.73 %**
True occupied	**46.61 %**	**3.28 %**	**4.34 %**
False occupied	**7.49 %**	**11.02 %**	**12.51 %**
Total correctly classified	**72.72 %**	**86.05 %**	**82.75 %**

Table 4.6: Average percent correctly and incorrectly classified cells, for each sensor and range, for sequence 2

	Radar with 20 meters range	**Radar with 5 meters range**	**USS with 5 meters range**
True free space	**26.77 %**	**77.89 %**	**69.09 %**
False free space	**24.40 %**	**2.81 %**	**1.75 %**
True occupied	**45.16 %**	**13.12 %**	**16.86 %**
False occupied	**3.67 %**	**6.18 %**	**12.30 %**
Total correctly classified	**71.93 %**	**91.01 %**	**85.95 %**

These two final tables can be seen as the major indicator of the performance of the free space estimation. Using radar with 5 meters range results in a higher percentage of total correctly classified cells, for both the tested sequences.

5
Discussion

5.1 Analysis of results

The major factor influencing our result of the radar is the inconsistency in the number of detections, presented in Section 4.2. Sometimes less than 50 combined detections were obtained from the 4 radars, in a single time step, which is very low. The circumstances are ideal, driving in a parking lot at a relatively low speed with lots of parked cars. This is a typical setting when free space estimation is needed. When driving at the same location with the same environment multiple times, it is expected that the radars would get approximately the same number of detections, but this is not the case. The inconsistency, together with the sometimes low number of detections, makes the radar unreliable. The vehicle's velocity over time was compared to the number of detections over time to see if there existed a correlation, but none could be found.

The dynamic filtering works well. When standing still and the dynamic detections are less than 50%, all the detections with a Doppler velocity of zero will be correctly classified as static. The threshold ϵ does not play an important role when the car stands still. However, when driving, ϵ has a greater impact on the performance. With a threshold of 0.3 m/s, the result is reasonable with a high number of inliers to estimate the velocities and angular velocity, as well as correctly classifying obvious dynamic detections. As seen in Section 4.6 the INED estimations get worse with a low number of detections. The sequences used for testing the dynamic filter are quite basic and more extensive testing would be needed for a more thorough analysis, but the result is promising.

The scan matching with landmarks and outlier rejection produces good results. Once again, it needs a minimum number of detections to work. One of the tuning parameters with a large impact on the result is the outlier rejection threshold. With a low threshold, the scan matching performs better. However, it gets more vulnerable to the number of detections. Since the vehicle is driving at low speed, the distance changed between two time-steps is short and the landmarks only move a short distance between scans. Our method overcomes the problem that radar data is noisy and contains outliers. The fact that the method does not work when the number of detections is low, is however concerning.

When it comes to the extended Kalman filter, looking at the innovation plots in Fig-

ure 4.21 and 4.23, we see that the errors are centered around zero, and the filter is working. Combining the scan matching and INED results gives a good performance compared to the IMU. This is impressive considering the 10 times faster sampling rate of the IMU. Combining all available data improves the result, reducing the drift from the IMU. One thing to note is that an optimal Kalman filter assumes white Gaussian noise, and the x and y positions estimated from the scan matching are dependent on the previous positions. This means that the noise is not independent of the previous state estimate, and the filter might not be optimal. However, by looking at the innovation once again, it can be confirmed that the error is minimized which means that the filter is working.

Our method of evaluating the free space estimations shows promising results. Checking the true and false free cells as well as the true and false occupied cells gives a concrete and measurable result. Table 4.5 and 4.6 show that evaluation within a range of 5 meters instead of the full 20 meters range for the radars gets a higher percent correctly classified free cells, while the USS has a higher percent correctly classified occupied cells. Using the estimations for the radar's full range cannot compete with the other option when it comes to percentage but classifies a lot more area. Since the radars also distinguish more details about the environment, we believe it has more potential for improvement than the USS.

5.2 Comparing different sensors

Comparing different sensors with different specifications is difficult. A radar detection contains a lot of information, while a USS only outputs a range. Both have advantages and disadvantages. Radar data is enough to estimate the trajectory of the vehicle. With radar data, dynamic objects can be detected and filtered out. With USS data, the free space can be estimated well in the first time step.

In sequences 1 and 2, which are used to test the free space estimation, the environment is static. Having a dynamic setting would not lead to a valid result for the USS, since there is no method of filtering dynamic objects. The grid map over time would be distorted and could not be used with good results. This is a disadvantage and using the USS in a realistic scenario would require a different method. Using the static setting makes a valid comparison between the two sensors but does not put a lot of pressure on the dynamic radar filter.

One choice that had to be made was how to interpret the radar data. All detections are classified as objects, but it is not 100% obvious how to handle areas with no detections. It could be classified as free space, or it could be classified as occupied since it is unknown. If there is a visible object in that area, we would get detection of the object, and the choice was made to classify an undetected area as free space up to the maximal range of the radar. Doing the opposite would result in an often tiny free space area, given the occasionally low number of radar detections. However, objects that are hard to detect in the field of view are now incorrectly classified as free space and can be a source of error.

The free space estimations are made in 2D. This is because of the low dimensional data from the USS. Compressing the radar detections between a certain interval in the z-axis, to go from 3D to 2D, is not optimal but has been done to get a comparable result. The parking lot used for the data collection is quite flat and works to compress into the 2D plane. However, if we were to drive in for example a multi-level parking garage, our method will face problems and needs to be extended.

5.3 The uncertainty of the ground truth

Our method of creating the occupancy grid from Google maps, used as the ground truth, works well. However, there are a lot of uncertainties. The parking lot used for the data collection is an ordinary parking lot. The lot contains curbs, sidewalks, grass, small bushes, etc., and the entire lot is surrounded by a chain-link fence. The fence is see-through meaning that the radar will receive detections from the fence as well as objects on the other side. The grass and sidewalk are classified as occupied in our map. With the definition that free space is the same as drivable space, this is reasonable. But since it is only a small curb separating the grass or sidewalk from the road, it is very difficult to use radar or USS data and see these areas as separate.

To be able to connect the Google maps grid with our estimations of trajectory and free space, the starting position and heading had to be estimated. Camera images were collected from the data collection of the different sequences, and these were used to estimate where the vehicle started and at which angle. Trying to pinpoint the position of the origin (the center of the rear axle) and add this to a satellite image of the parking lot, will not be perfect. Therefore, the result of the trajectory and free space should not be interpreted as the exact number of how well it performs. Instead, what we get is a fair comparison between the different methods with the same preconditions.

5.4 Further development

The different methods used in this thesis have been evaluated separately and on a limited amount of data. To properly evaluate and compare the methods, more profound testing must be done. Different weather conditions and settings should be used to show the robustness of our methods.

The algorithms are at the moment not fast enough to be performed in real-time. The highest computational complexity is the RANSAC algorithm, connecting the correspondences in the scan matching, and generating the occupancy grid with small cell sizes. Using landmarks for the scan matching lowers the computational time considerably, but additional analysis of the performance needs to be done for the algorithm to run in real-time.

A lot of parking lots are flat and suitable for our solution. However, parking lots

with large height differences are not suitable for how we estimate the free space. This needs to be investigated further before commercial use.

6
Conclusion

This thesis investigates the performance of a new 4D short-range radar to see if it could be used instead of USS to estimate the surrounding free space in low-speed scenarios. Knowing the elevation of radar detections as well as the range, azimuth, and Doppler velocity opens up new potentials, including an increased ability to estimate the trajectory. However, for the solution to be more robust, the number of detections for each scan has to be more consistent. The field of use for radar data greatly exceeds the field for USS data, but both have advantages and disadvantages. The ability to filter out dynamic objects and update a static map over time makes the radar method more agile. The amount of information from the radar also enables trajectory estimations with results comparable to the IMU sensor. Analyzing the free space estimations, it is clear that the performance, using the same maximum range, is similar between the two sensors, but with a small advantage for the radar. There is a short period at the beginning of each sequence when the radars estimate the free space worse than the USS. This can be explained by the low number of detections at the beginning, which is not enough to create an accurate representation of the environment. However, the USS has good estimations from the very first time step.

Since no clear ground truth existed, a method for generating a ground truth was created to evaluate the estimations. Even though the ground truth may not be 100% correct it allows for a fair comparison. A metric to analyze free space and compare estimations from different sensors with different specifications was also produced, using a percentage of true and false classified free cells as well as true and false classified occupied cells.

With the methods used, we conclude that the radar outperforms the USS. Table 4.5 and 4.6 clearly show that the radar has a better performance with a higher percentage of correct classified cells when using the same range as the USS. The fact that the radar is still under development, makes the possible potential even higher, and should the problem with the low number of detections be solved, the performance of the radar could be increased additionally. Further development and more testing are needed before commercial use.

Bibliography

[1] H. Durrant-Whyte and T. Bailey, "Simultaneous localization and mapping: part I," *IEEE Robotics & Automation Magazine*, vol. 13, pp. 99–110, Jun. 2006.

[2] T. Bailey and H. Durrant-Whyte, "Simultaneous localization and mapping (SLAM): part II," *IEEE Robotics & Automation Magazine*, vol. 13, pp. 108–117, Sep. 2006.

[3] C. Cadena, L. Carlone, H. Carrillo, Y. Latif, D. Scaramuzza, J. Neira, I. Reid, and J. J. Leonard, "Past, Present, and Future of Simultaneous Localization and Mapping: Toward the Robust-Perception Age," *IEEE Transactions on Robotics*, vol. 32, pp. 1309–1332, Dec. 2016.

[4] M. Schoen, M. Horn, M. Hahn, and J. Dickmann, "Real-time radar SLAM," in *Workshop Fahrerassistenzsysteme und automatisiertes Fahren*, pp. 1–11, Mar. 2017.

[5] F. Schuster, C. G. Keller, M. Rapp, M. Haueis, and C. Curio, "Landmark based radar SLAM using graph optimization," in *2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC)*, (Rio de Janeiro, Brazil), pp. 2559–2564, IEEE, Nov. 2016.

[6] M. Holder, S. Hellwig, and H. Winner, "Real-Time Pose Graph SLAM based on Radar," in *2019 IEEE Intelligent Vehicles Symposium (IV)*, (Paris, France), pp. 1145–1151, IEEE, Jun. 2019.

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

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

[9] S. H. Cen and P. Newman, "Radar-only ego-motion estimation in difficult settings via graph matching," in *2019 International Conference on Robotics and Automation (ICRA)*, pp. 298–304, IEEE, Apr. 2019.

[10] P. Yang, "Efficient particle filter algorithm for ultrasonic sensor-based 2D range-only simultaneous localisation and mapping application," *Wireless Sensor Systems, IET*, vol. 2, pp. 394–401, Dec. 2012.

[11] P. Słowak and P. Kaniewski, "LIDAR-based SLAM implementation using Kalman filter," in *Radioelectronic Systems Conference 2019*, vol. 11442, pp. 198–207, SPIE, Feb. 2020.

[12] M. Li, Z. Feng, M. Stolz, M. Kunert, R. Henze, and F. Küçükay, "High Resolution Radar-based Occupancy Grid Mapping and Free Space Detection," in *VEHITS*, pp. 70–81, Mar. 2018.

[13] M. Schreier, V. Willert, and J. Adamy, "From grid maps to Parametric Free Space maps; A highly compact, generic environment representation for ADAS," in *2013 IEEE Intelligent Vehicles Symposium (IV)*, (Gold Coast City, Australia), pp. 938–944, IEEE, Jun. 2013.

[14] R. Kümmerle, B. Steder, C. Dornhege, M. Ruhnke, G. Grisetti, C. Stachniss, and A. Kleiner, "On measuring the accuracy of SLAM algorithms," *Autonomous Robots*, vol. 27, pp. 387–407, Nov. 2009.

[15] R. Grewe, M. Komar, A. Hohm, S. Lueke, and H. Winner, "Evaluation method and results for the accuracy of an automotive occupancy grid," in *2012 IEEE International Conference on Vehicular Electronics and Safety (ICVES 2012)*, pp. 19–24, Sep. 2012.

[16] S. O. Hansson, M.-A. Belin, and B. Lundgren, "Self-driving vehicles—an ethical overview," *Philosophy and Technology*, vol. 34, Dec. 2021.

[17] M. I. Skolnik, *Radar Handbook, Third Edition.* USA: McGraw-Hill Education, Feb. 2008.

[18] C. Iovescu and S. Rao, "The fundamentals of millimeter wave sensors," *Dallas: Texas Instruments*, pp. 1–8, May 2017.

[19] J. Hasch, E. Topak, R. Schnabel, T. Zwick, R. Weigel, and C. Waldschmidt, "Millimeter-Wave Technology for Automotive Radar Sensors in the 77 GHz Frequency Band," *IEEE Transactions on Microwave Theory and Techniques*, vol. 60, pp. 845–860, Jan. 2012.

[20] B. Siciliano and O. Khatib, *Springer Handbook of Robotics.* Berlin: Springer, May 2008.

[21] A. Carullo and M. Parvis, "An ultrasonic sensor for distance measurement in automotive applications," *IEEE Sensors journal*, vol. 1, p. 143, Aug. 2001.

[22] D. Bank, "A novel ultrasonic sensing system for autonomous mobile systems," *IEEE Sensors Journal*, vol. 2, pp. 597–606, Nov. 2002.

[23] N. Ahmad, R. A. R. Ghazilla, N. M. Khairi, and V. Kasi, "Reviews on various inertial measurement unit (IMU) sensor applications," *International Journal of Signal Processing Systems*, vol. 1, pp. 256–262, Jan. 2013.

[24] T. Basar, "A New Approach to Linear Filtering and Prediction Problems," *Control Theory: Twenty-Five Seminal Papers (1932-1981)*, pp. 167–179, Jul. 2001.

[25] S. J. Julier and J. K. Uhlmann, "New extension of the Kalman filter to nonlinear systems," in *Signal Processing, Sensor Fusion, and Target Recognition VI* (I. Kadar, ed.), vol. 3068, pp. 182 – 193, International Society for Optics and Photonics, SPIE, Jul. 1997.

[26] S. Thrun, W. Burgard, and D. Fox, *Probabilistic robotics.* Cambridge, Mass: MIT Press, Aug. 2005.

[27] J. E. Bresenham, "Ambiguities in Incremental Line Rastering," *IEEE Computer Graphics and Applications*, vol. 7, pp. 31–43, May 1987.

[28] K. G. Derpanis, "Overview of the RANSAC Algorithm," *Image Rochester NY*, vol. 4, pp. 2–3, May 2010.

[29] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, "A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise," in *The Second International Conference on Knowledge Discovery and Data Mining*, vol. 96, pp. 226–231, Aug. 1996.

[30] E. Schubert, J. Sander, M. Ester, H. P. Kriegel, and X. Xu, "DBSCAN revisited, revisited: why and how you should (still) use DBSCAN," *ACM Transactions on Database Systems (TODS)*, vol. 42, pp. 1–21, Jul. 2017.

[31] W. H. Lai, S. L. Kek, and K. G. Tay, "Solving nonlinear least squares problem using Gauss-Newton method," *International Journal of Innovative Science, Engineering & Technology*, vol. 4, pp. 258–262, Jan. 2017.

[32] Q. Gan and C. Harris, "Comparison of two measurement fusion methods for Kalman-filter-based multisensor data fusion," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 37, pp. 273–279, Jan. 2001.

[33] M. Li, Z. Feng, M. Stolz, M. Kunert, R. Henze, and F. Küçükay, "High Resolution Radar-based Occupancy Grid Mapping and FreeSpace Detection:," in *Proceedings of the 4th International Conference on Vehicle Technology and Intelligent Transport Systems*, (Funchal, Madeira, Portugal), pp. 70–81, SCITEPRESS - Science and Technology Publications, Mar. 2018.

DEPARTMENT OF ELECTRICAL ENGINEERING
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden
www.chalmers.se

CHALMERS
UNIVERSITY OF TECHNOLOGY